

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проекту (роботи)

бакалавр

(ступінь вищої освіти)

на тему ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ДЛЯ НАЙМУ
РОБІТНИКІВ ДЛЯ ВИКОНАННЯ РЕМОНТУ БУДИНКУ
SOFTWARE FOR HIRING WORKERS FOR
HOME RENOVATION

Виконав(ла): студент(ка) 4 курсу, групи КНТ-141

Спеціальності 121 Інженерія програмного

(код і найменування спеціальності)

забезпечення

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

ТРОЙКА Д.О.

(ПРИЗВИЩЕ та ініціали)

Керівник ЛЬОВКІН В.М.

(ПРИЗВИЩЕ та ініціали)

Рецензент ЗЕЛІК О.В.

(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
 (повне найменування закладу вищої освіти)

Факультет КНТ
 Кафедра програмних засобів
 Ступінь вищої освіти бакалавр
 Спеціальність 121 Інженерія програмного забезпечення
 (КОД І НАЙМЕНУВАННЯ)
 Освітня програма (спеціалізація) Інженерія програмного забезпечення
 (назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
Сергій СУББОТІН
 “ ” 2025 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

ТРОЙКИ Дениса Олександровича
 (ПРІЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Програмне забезпечення для найму робітників для виконання ремонту будинку. Software for Hiring Workers for Home Renovation

керівник проєкту (роботи) к.т.н., доцент ЛЬОВКІН Валерій Миколайович,
 (науковий ступінь, вчене звання, ПРІЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від “ 28 ” квітня 2025 року № 209

2. Строк подання студентом проєкту (роботи) 02 червня 2025 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Проєктування програми. 3. Розробка програми. 4. Експлуатація та тестування програми.

5. Перелік графічного матеріалу (з з точним зазначенням обов'язкових креслень, кількість слайдів, плакатів)

Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	ПРИЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	ЛЬОВКІН В.М., доцент		
Нормоконтроль	БСЛОВА А.В., асистент		

7. Дата видачі завдання « 14 » квітня 2025 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	1 тиждень	Розділ 1
3	Вибір мови програмування та інших технологій розробки.	2 тиждень	Розділ 2
4	Розробка архітектури програми.	2 тиждень	Розділ 3
5	Розробка програми.	3-4 тижні	Розділ 3, 4
6	Тестування та експериментальне дослідження програмного забезпечення.	5 тиждень	Розділ 4
7	Оформлення пояснювальної записки та документів до неї.	6 тиждень	Додатки
8	Нормоконтроль та рецензування.	7 тиждень	
9	Захист роботи.	8 тиждень	

Студент(ка)

_____ Денис ТРОЙКА
(підпис) (Ім'я ПРИЗВИЩЕ)

Керівник проєкту (роботи)

_____ Валерій ЛЬОВКІН
(підпис) (Ім'я ПРИЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 102 с., 3 табл., 26 рис., 3 дод., 6 джерел.

РЕМОНТ БУДИНКІВ, КОМПАНІЯ, БРИГАДА, ОЦІНЮВАННЯ, ВЕБЗАСТОСУНОК, БАЗА ДАНИХ.

Об'єкт роботи – процес найму робітників для виконання ремонту будинку. Предмет роботи – програмне забезпечення найму робітників для виконання ремонту будинку. Мета роботи – розробити програму найму робітників, направленою на виконання ремонту будинку, для забезпечення принципу двосторонньої пропозиції.

Матеріали, методи та технічні засоби: мова Python для реалізації програми разом з фреймворком Django, система MySQL для роботи з базою даних.

Результати. Програма призначена для пошуку робітників для виконання ремонту будинку з об'єднанням таких робітників у бригади, компанії з ремонту будинків, пошуку оголошень про необхідний ремонт будинку, забезпечення заявок для реалізації ремонту будинків, управління цими заявками.

Висновки. Реалізація принципу двосторонньої пропозиції дозволяє кожній зі сторін, власникам будинків та робітникам, бригадам і компаніям з ремонту будинків виставляти власні позиції, пропонуючи задоволення таких пропозицій в основі реалізації подальшої взаємодії. Це було забезпечено в розробленій програмі, що дозволяє підтримувати реалізацію принципу двосторонньої пропозиції під час найму робітників для виконання ремонту будинку.

Галузь використання. Ремонт будинків окремими робітниками, бригадами робітників та компаніями з ремонту будинків.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 102 pages, 3 tables, 26 figures, 3 appendixes, 6 sources.

HOME RENOVATION, COMPANY, CREW, EVALUATION, WEB APPLICATION, DATABASE.

The object of the work is the process of hiring workers to perform home renovation. The subject of the work is software for hiring workers to perform home renovation. The purpose of the work is to develop a program for hiring workers to perform home renovation, to ensure the principle of two-way offer.

Materials, methods and technical tools: Python language for implementing the program together with the Django framework, MySQL system for working with the database.

Results. The program is designed to search for workers to perform home renovation with the unification of such workers into crews, house repair companies, search for announcements about necessary house repairs, provide applications for the implementation of home renovation, manage these applications.

Conclusions. The implementation of the principle of two-way offer allows each of the parties, home owners and workers, crews and house repair companies, to present their own positions, offering to satisfy such proposals as the basis for the implementation of further interaction. It was ensured in the developed program, which allows supporting the implementation of the principle of two-way offer when hiring workers to perform home renovation.

Field of use. Home renovations by individual workers, teams of workers and home repair companies.

ЗМІСТ

	С.
Перелік скорочень та умовних познач.....	8
Вступ.....	9
1 Аналіз предметної області.....	10
1.1 Огляд програм найму робітників для виконання ремонту будинку	10
1.2 Порівняння програм найму робітників для виконання ремонту будинку	13
1.3 Висновки за розділом 1	14
2 Проєктування програми.....	15
2.1 Вибір засобів проєктування і розробки програми найму робітників для виконання ремонту будинку.....	15
2.2 Моделювання роботи користувачів з програмним забезпеченням найму робітників для виконання ремонту будинку.....	16
2.3 Визначення структури бази даних програми найму робітників для виконання ремонту будинку.....	23
2.4 Висновки за розділом 2	35
3 Розробка програми	36
3.1 Визначення послідовності роботи з програмою найму робітників для виконання ремонту будинку.....	36
3.2 Особливості реалізації програми найму робітників для виконання ремонту будинку.....	37
3.3 Висновки за розділом 3	55
4 Експлуатація та тестування програми	56
4.1 Призначення програми найму робітників для виконання ремонту будинку	56
4.2 Умови виконання програми найму робітників для виконання ремонту будинку	56
4.3 Експлуатація програми найму робітників для виконання ремонту будинку	57

4.4 Тестування програми найму робітників для виконання ремонту будинку	60
4.5 Висновки за розділом 4	65
Висновки	66
Перелік джерел посилання	67
Додаток А Технічне завдання	68
Додаток Б Текст програми	74
Додаток В Слайди презентації	95

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

БД – база даних;
ВЗ – вебзастосунок.

ВСТУП

Ремонт будинку – це завжди складний процес, який потребує багато уваги, часу, фінансів, а також ще й нерідко відзначається стресом. Від виконавців ремонту залежить вплив кожного з цих факторів. Тому пошук робітників для виконання ремонту будинку є вкрай важливою основою для даного процесу. Від результатів його виконання залежать всі наступні етапи.

Виконувати пошук на сайтах окремих компаній з ремонту будинків фактично неможливо для якісного вирішення цієї проблеми. Це пояснюється тим, що в такому випадку неможливо орієнтуватися на відгуки замовників, тому що на вебсайті кожної окремої компанії рідко приводяться правдиві відгуки, але окрім того точно можна сказати, що такі відгуки є правдивими точно не на кожному сайті.

Тому потрібен засіб, в якому б різні працівники, можливо бригади з ремонту будинків та компанії могли пропонувати свої послуги, а власники будинків шукати тих, хто надасть їм такі послуги. Важливо детально визначити необхідні можливості для такої програми, щоб повноцінно охоплювати можливості найму робітників для виконання ремонту будинку через програму.

Для того щоб отримувати доступ до такої програми простіше, не вимагати особливих типів пристроїв, її варто розробити як вебзастосунок (ВЗ).

Мета роботи полягала в тому, щоб розробити програму найму робітників, направленою на виконання ремонту будинку, для забезпечення принципу двосторонньої пропозиції.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд програм найму робітників для виконання ремонту будинку

На даний момент вже існують деякі програмні рішення, які забезпечують роботу в сфері найму робітників для виконання ремонту будинку.

До таких програм зокрема відноситься Rabotniki.ua (рис. 1.1) [1].

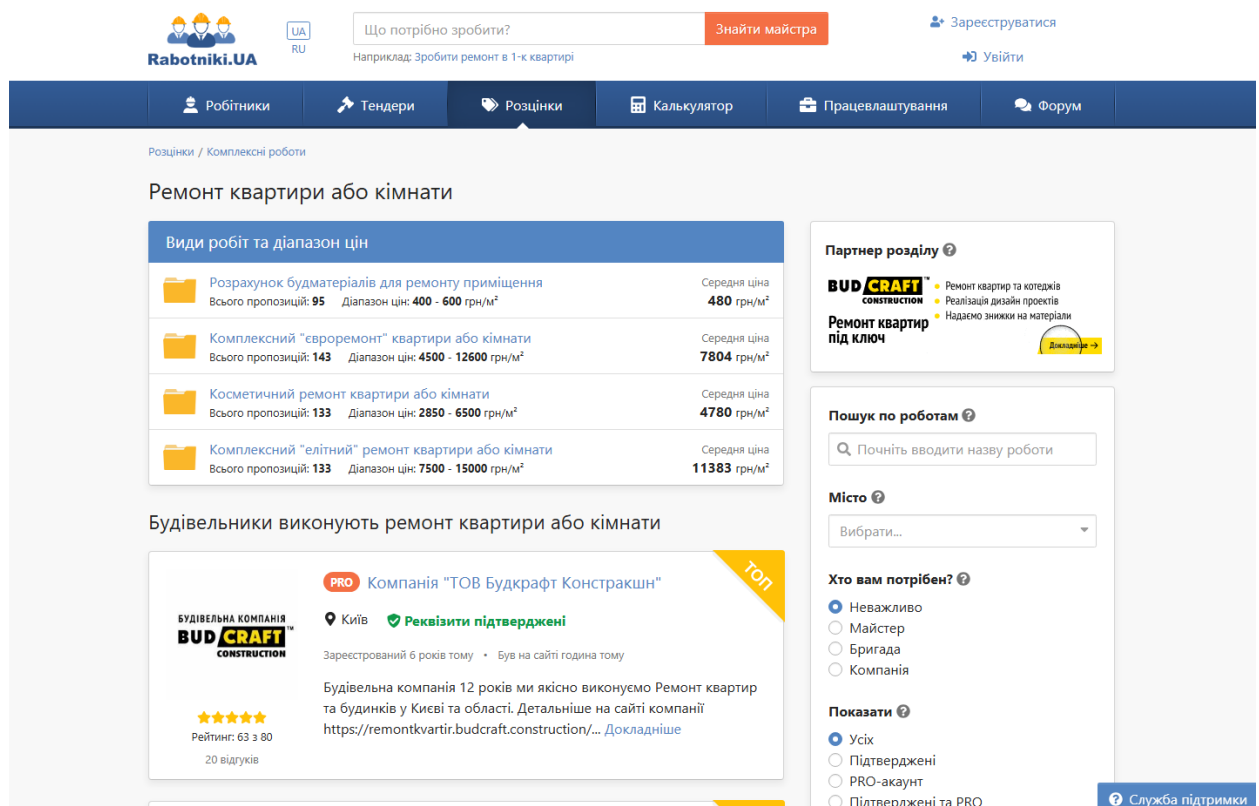


Рисунок 1.1 – Інтерфейс ВЗ Rabotniki.ua

Rabotniki.ua дозволяє виконувати пошук робітників для виконання ремонтів у квартирах або кімнатах. Це можуть бути окремі робітники (рис. 1.2), які позиціонують себе окремо, а можуть бути компанії з ремонту. За робітниками та компаніями доступні відгуки про них, можна надсилати повідомлення, але не можна формувати заявки на конкретні роботи на

конкретний час. Тобто подальша взаємодія реалізується як довільне спілкування.

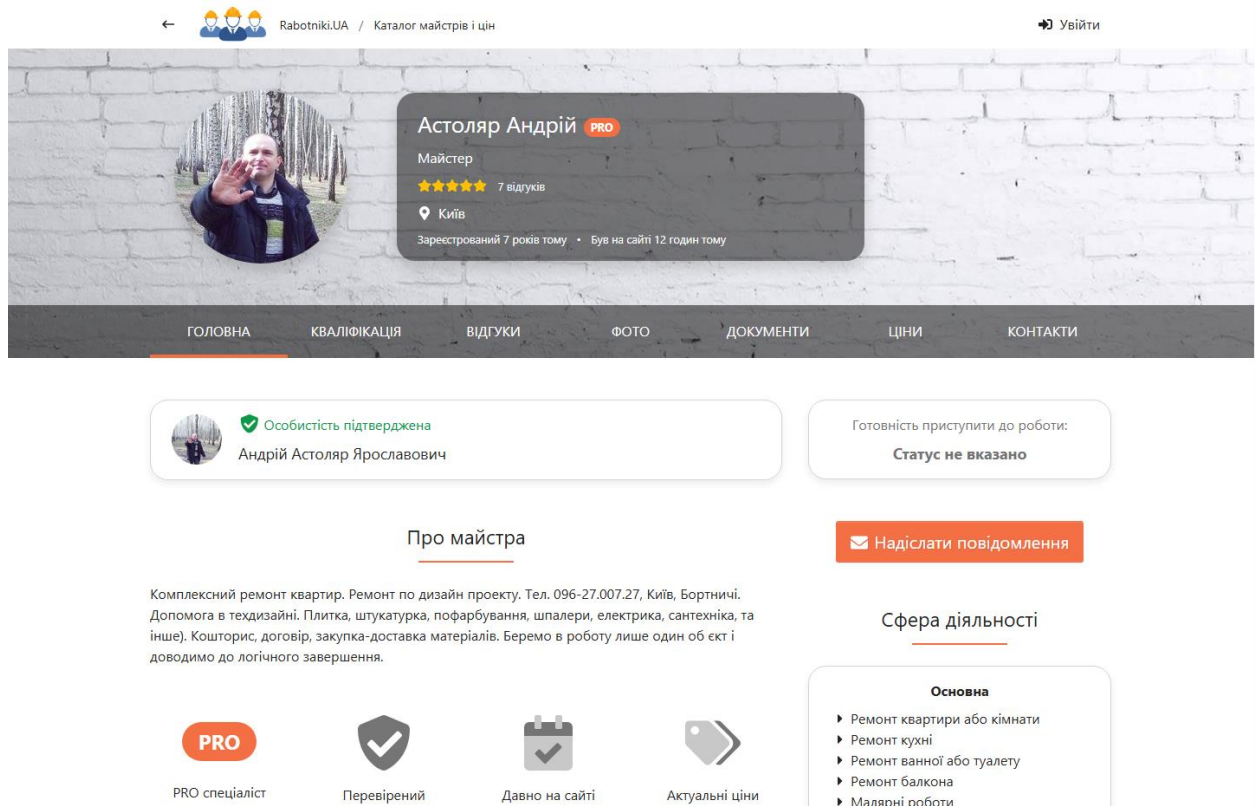


Рисунок 1.2 – Сторінка робітника, який здатен виконувати ремонт, у ВЗ Rabotniki.ua

На сторінках робітників або компаній доступна інформація про кваліфікацію, наявні документи, фотографії, вартість послуг. Вартість встановлюється чітко без вказання діапазонів.

ВЗ oosh.com.ua (рис. 1.3), на відміну від попереднього, дозволяє формувати конкретне завдання робітнику [2]. Проте не для всіх робітників такі завдання проставлені, тому фактично конкретика, а тим більше дати виконання робіт, не визначаються. Окрім переліку робіт за робітником доступні також відгуки та портфоліо. Наводиться також статистика виконаних завдань, активних та сумарна. Наявний пошук.

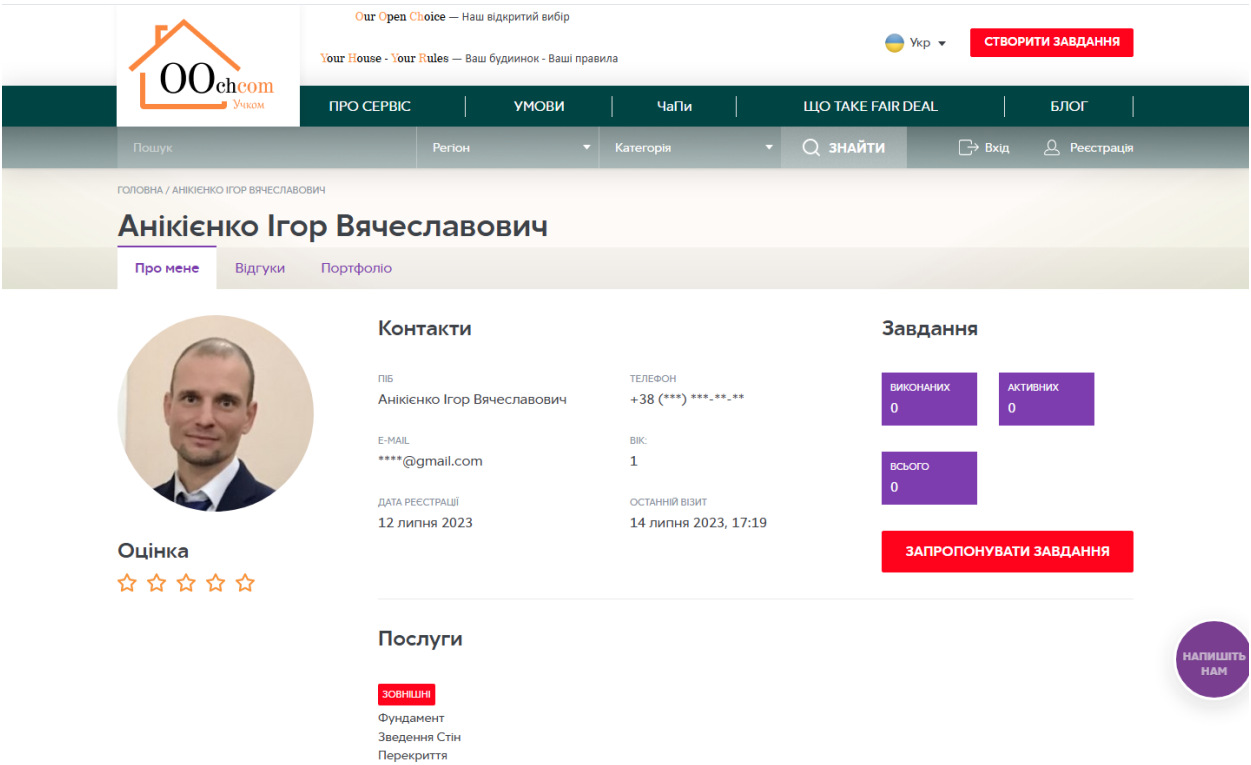


Рисунок 1.3 – Інтерфейс ВЗ Ooch.com.ua

Також подібні засоби доступні і на платформі OLX (рис. 1.4) [3]. У даному випадку робітники виставляють оголошення про те, які ремонтні роботи вони здатні виконувати.

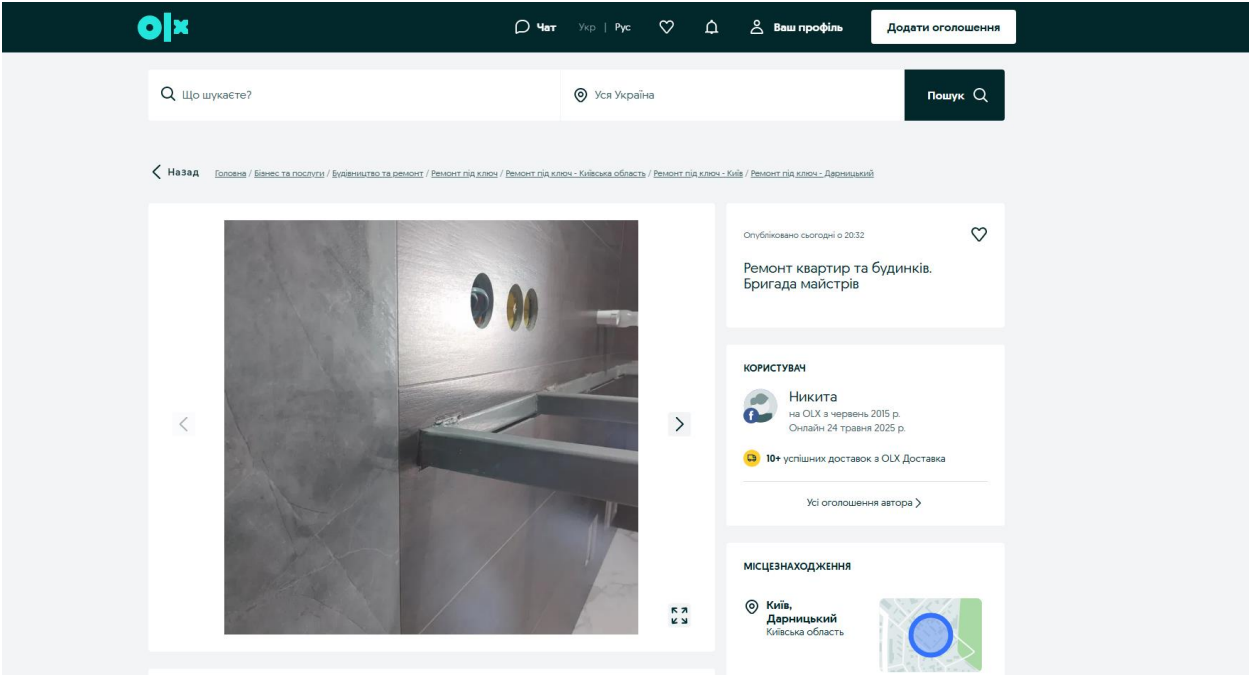


Рисунок 1.4 – Оголошення про ремонтні роботи бригади у ВЗ OLX

У даному випадку може бути в оголошенні приведено будь-яка інформація, але вона не має стандартної структуризації. В результаті, наприклад, інформація про вартість робіт найчастіше не приводиться. Виконати замовлення на ремонт за певним планом неможливо.

1.2 Порівняння програм найму робітників для виконання ремонту будинку

Підсумкове порівняння існуючих програм найму робітників для виконання ремонту будинку приведено до Rabotniki.ua, Ooch.com.ua та розробки, що виконується в роботі (табл. 1.1).

Таблиця 1.1 – Порівняння програм найму робітників для виконання ремонту будинку

Характеристика	Rabotniki.ua	Ooch.com.ua	Розробка
Оголошення про необхідність ремонту будинків від власників	—	—	+
Оголошення про ремонтні послуги робітників	+	+	+
Визначення пропонованого плану робіт ремонту	—	—+	+
Замовлення ремонту з параметрами	—	—	+
Перегляд доступності бригади або робітника для виконання ремонту	—	—	+
Визначення портфоліо робіт	+	+	+

У процесі огляду було виявлено, що існуючі програми не дозволяють створювати власні оголошення власникам житла (будинків). Тобто вони не

дозволяють ставати власника будинків активними учасниками взаємодії. Це має бути усунуто в роботі на основі створення двосторонньої пропозиції.

Окрім того було виявлено, що існуючі програми не дозволяють формувати конкретне замовлення ремонту, визначаючи параметри такого ремонту, а з доступністю за часом робітника або бригади ознайомитись неможливо. Тобто фактично вони працюють виключно на основі відправлення повідомлення від заявника робітнику, що обмежує їх інформативність та використання. Подальша ж робота має відбуватися в інших програмах, тобто реалізовуватися у месенджерах, які в такому випадку будуть схильні використовувати користувачі. Цей недолік вирішено владнати у цій розробці.

Визначення пропонованого плану робіт ремонту як таке відсутнє в Oosh.com.ua, проте можливо визначити з переліку потрібне завдання для роботи. Це одне загальне завдання тільки, окрім того фактично такий перелік часто для робітників взагалі не заданий.

1.3 Висновки за розділом 1

У цьому розділі огляд програм найму робітників для виконання ремонту будинку та виконане порівняння дозволили встановити, що існуючі програми не дозволяють формувати конкретне замовлення ремонту, визначаючи параметри такого ремонту, а з доступністю за часом робітника або бригади ознайомитись неможливо. Ці можливості разом з підтримкою можливостей двосторонньої пропозиції вирішено реалізувати в розробці.

Завдання роботи:

- аналіз проблеми розробки програми найму робітників для виконання ремонту будинку;
- проектування програми найму робітників для виконання ремонту будинку;
- розробка програми найму робітників для виконання ремонту будинку.

2 ПРОЄКТУВАННЯ ПРОГРАМИ

2.1 Вибір засобів проєктування і розробки програми найму робітників для виконання ремонту будинку

Python-розробка дозволяє охопити велику кількість доступних засобів у процесі створення застосунку, що сприяє подальшому розвитку програми. Зважаючи на те, що веброзробка є одним з таких напрямків, то створювати ВЗ вирішено мовою Python, а вибір засобів розробки зведено до фреймворків Pyramid [4] і Django [5] (табл. 2.1). Pyramid може дозволяти розширювати власні засоби, проте тоді витрачається додатковий час на вибір, підключення, налаштування, тестування створеного рішення. Такий час не вимагається у Django. Тому саме Django як результат визначено фреймворком для програми робітників для виконання ремонту будинку.

Таблиця 2.1 – Вибір засобів розробки програми найму робітників для виконання ремонту будинку

Характеристика	Pyramid	Django
Мова програмування	Python	Python
Відносний час розробки	Більший	Менший
Підтримка роботи з користувачами для програми	Базово відсутня	Забезпечується авторизація всіх користувачів, підтримка адміністраторів
Взаємодія з базою даних	Необхідно налаштувати	Встановлено

2.2 Моделювання роботи користувачів з програмним забезпеченням найму робітників для виконання ремонту будинку

Основний потік сценаріїв користувача до авторизації в програмі найму робітників для виконання ремонту будинку:

- авторизація;
- перегляд переліку ремонтних робіт, які здатна виконувати компанія;
- перегляд даних власника будинку;
- перегляд даних компанії з ремонту будинків;
- перегляд переліку бригад з ремонту будинків компанії;
- перегляд даних бригади з ремонту будинків;
- перегляд доступних для роботи бригад у компанії з ремонту будинків на заданий час;

- перегляд переліку необхідних ремонтів будинків власника;
- перегляд оголошення про необхідність ремонту будинку;
- перегляд оцінювань власника будинку;
- перегляд оцінювань компанії з ремонту будинків;
- пошук оголошень про необхідність ремонту будинку;
- пошук компанії з ремонту будинку.

Альтернативні сценарії користувача до авторизації в програмі найму робітників для виконання ремонту будинку:

- реєстрація облікового запису як власника будинку для авторизації, коли потрібен обліковий запис власника будинку, а його ще немає;
- реєстрація облікового запису як робітника з ремонту будинків, коли потрібен обліковий запис робітника з ремонту будинків, а його ще немає (рис. 2.1).



Рисунок 2.1 – Діаграма прецедентів програми найму робітників для виконання ремонту будинку для користувача до авторизації

Основний потік сценаріїв власника будинку в програмі найму робітників для виконання ремонту будинку:

- визначення даних власника будинку;
- створення оголошення про необхідний ремонт будинку;
- додавання кімнати до даних ремонту будинку;
- додавання роботи до плану ремонту будинку;
- подання заявки на виконання ремонту будинку;
- управління станом заявки на виконання ремонту будинку;
- перегляд заявки на виконання ремонту будинку;
- перегляд заявок на виконання ремонту за будинком;

- перегляд власних поданих заявок на виконання ремонту будинку;
- перегляд станів, які визначались за заявкою на виконання ремонту будинку;
- оцінювання компанії з ремонту будинків;
- перегляд оцінених компаній з ремонту будинків власником будинку;
- створення закладки за компанією з ремонту будинку;
- перегляд переліку компаній з ремонту будинку в закладках.

Альтернативні сценарії власника будинку в програмі найму робітників для виконання ремонту будинку:

- оновлення оголошення про необхідний ремонт будинку для створення оголошення про необхідний ремонт будинку, коли оголошення про необхідний ремонт будинку було створено, але потрібно його оновити;
- вилучення оголошення про необхідний ремонт будинку для створення оголошення про необхідний ремонт будинку, коли оголошення про необхідний ремонт будинку було створено, але воно більше не потрібне;
- вилучення закладки за компанією з ремонту будинку для створення закладки за компанією з ремонту будинку, коли закладка за компанією з ремонту будинку саме цим власником будинку була створена, але вона йому не потрібна;
- вилучення роботи з плану ремонту будинку для додавання роботи до плану ремонту будинку, коли ця робота до плану ремонту будинку була додана, але більше не потрібна в плані;
- оновлення роботи з плану ремонту будинку для додавання роботи до плану ремонту будинку, коли ця робота до плану ремонту будинку була додана, але має бути змінено деякі її параметри;
- оновлення кімнати з даних ремонту будинку для додавання кімнати до даних ремонту будинку, коли кімната була додана до даних ремонту будинку, але параметри потрібно оновити;

– вилучення кімнати з даних ремонту будинку для додавання кімнати до даних ремонту будинку, коли кімната була додана до даних ремонту будинку, але ці дані більше не потрібні (рис. 2.2).

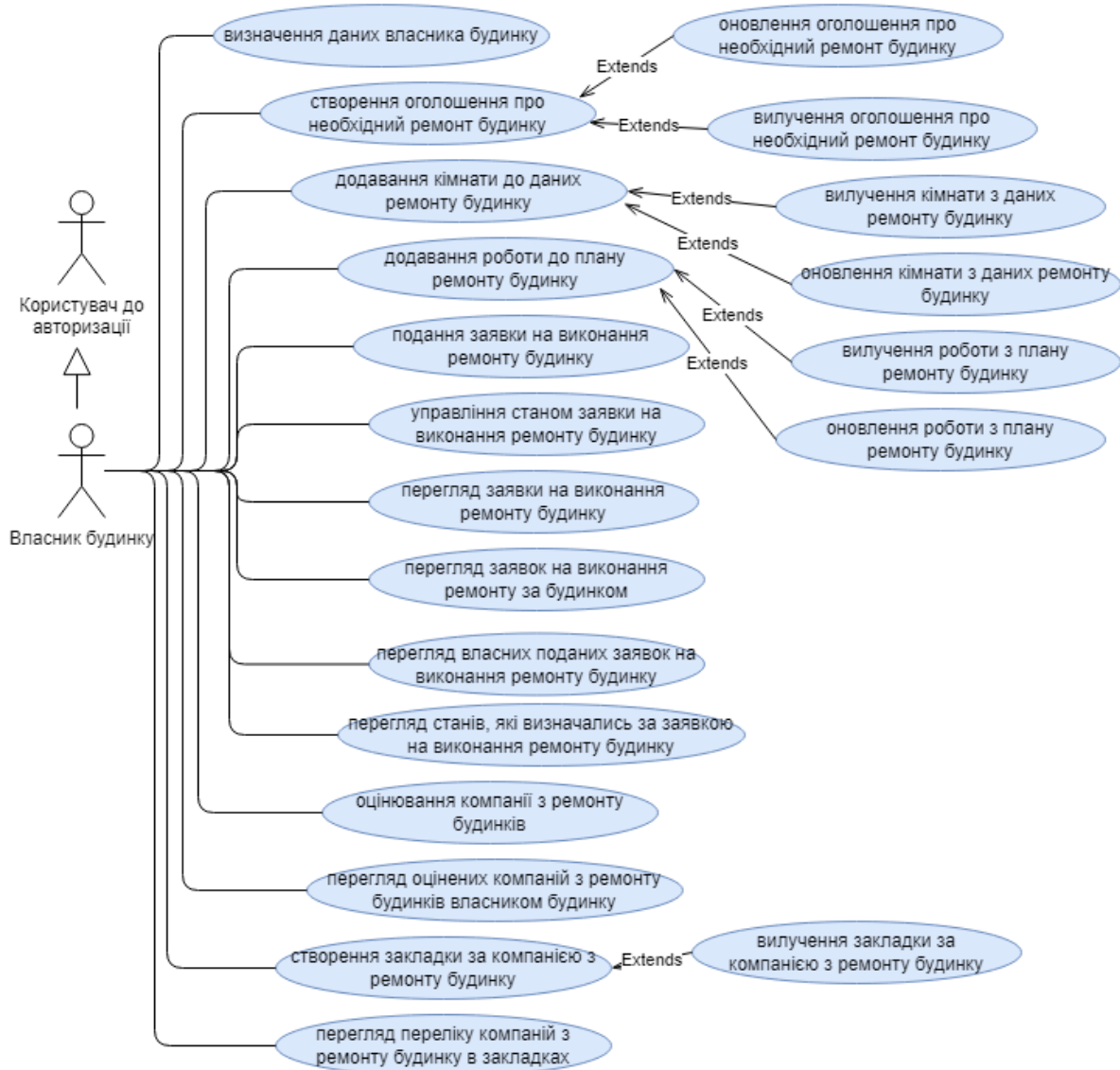


Рисунок 2.2 – Діаграма прецедентів програми найму робітників для виконання ремонту будинку для власника будинку

Основний потік сценаріїв робітника з ремонту будинків у програмі найму робітників для виконання ремонту будинку:

- визначення власної компанії з ремонту будинків;
- додавання ремонтної роботи для компанії з ремонту будинків;
- додавання бригади з ремонту будинків компанії;

- оновлення даних робітника з ремонту будинків;
- перегляд даних робітника з ремонту будинків;
- перегляд складу робітників бригади з ремонту будинків;
- додавання часового лоту для роботи бригади з ремонту будинків;
- додавання ремонтної роботи до можливостей бригади;
- подання заявки на виконання ремонту будинку;
- управління станом заявки на виконання ремонту будинку;
- перегляд заявки на виконання ремонту будинку;
- перегляд заявок на виконання ремонту будинку;
- перегляд власних поданих заявок на виконання ремонту будинку;
- додавання роботи до плану ремонту будинку за заявкою;
- перегляд станів, які визначались за заявкою на виконання ремонту будинку;
- оцінювання власника будинку;
- перегляд оцінених власників будинків бригадою з ремонту будинків;
- створення закладки за ремонтом будинку;
- перегляд переліку ремонтів будинків у закладках.

Альтернативні сценарії робітника з ремонту будинків у програмі найму робітників для виконання ремонту будинку:

- оновлення даних компанії з ремонту будинків для визначення власної компанії з ремонту будинків, коли власна компанія з ремонту будинків вже визначена, проте в параметрах цієї компанії потрібні зміни;
- вилучення закладки за ремонтом будинку для створення закладки за ремонтом будинку, коли закладка цим робітником з ремонту будинків за цим ремонтом будинку в оголошенні була створена, але вона йому не потрібна;
- вилучення роботи з плану ремонту будинку за заявкою для додавання роботи до плану ремонту будинку за заявкою, коли ця ремонтна робота в план ремонту цього будинку з заявкою вже була внесена, але більше не потрібна в ньому;

– оновлення роботи з плану ремонту будинку за заявкою для додавання роботи до плану ремонту будинку за заявкою, коли ця ремонтна робота в план ремонту цього будинку з заявкою вже була внесена, але в параметрах цієї роботи потрібні зміни;

– вилучення ремонтної роботи з можливостей бригади для додавання ремонтної роботи до можливостей бригади, коли ця ремонтна робота вже не відповідає можливостям цієї бригади;

– оновлення часового лоту для роботи бригади з ремонту будинків для додавання часового лоту для роботи бригади з ремонту будинків, коли цей часовий лот для роботи бригади з ремонту будинків був створений, але повинен бути уточнений;

– вилучення часового лоту для роботи бригади з ремонту будинків для додавання часового лоту для роботи бригади з ремонту будинків, коли цей часовий лот для роботи бригади з ремонту будинків був створений, але він більше не доступний з тих чи інших причина самого виконавця;

– оновлення ремонтної роботи для компанії з ремонту будинків для додавання ремонтної роботи для компанії з ремонту будинків, коли ця ремонтна робота для компанії з ремонту будинків вже була додана, але потребує оновлення;

– вилучення ремонтної роботи для компанії з ремонту будинків для додавання ремонтної роботи для компанії з ремонту будинків, коли ця ремонтна робота для компанії з ремонту будинків вже була додана, але більше не забезпечується цією компанією з ремонту будинків;

– оновлення даних бригади з ремонту будинків компанії для додавання бригади з ремонту будинків компанії, коли дані бригади з ремонту будинків потребують оновлення;

– вилучення бригади з ремонту будинків компанії для додавання бригади з ремонту будинків компанії, коли ця бригада з ремонту будинків компанії вже не функціонує (рис. 2.3).

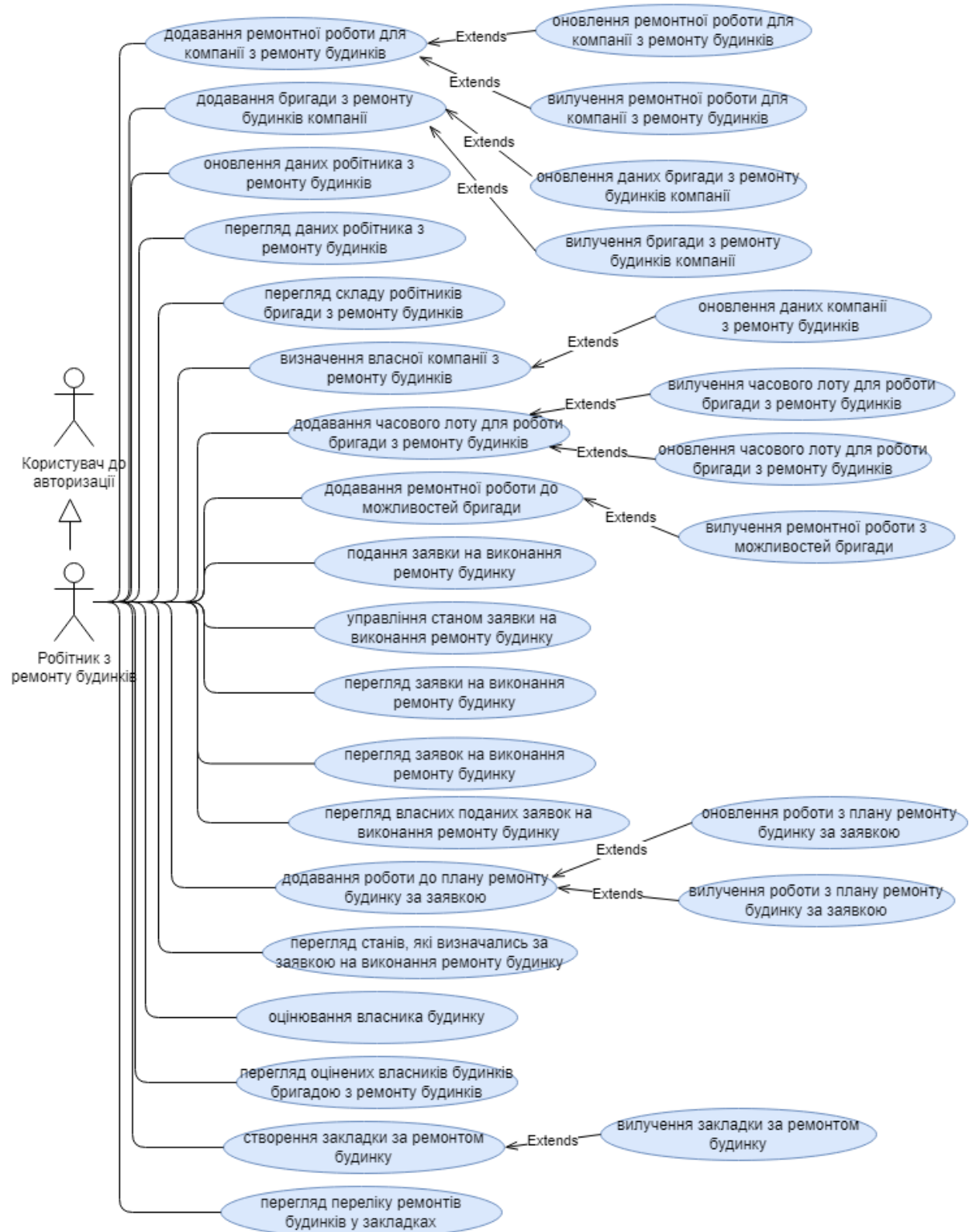


Рисунок 2.3 – Діаграма прецедентів програми найму робітників для виконання ремонту будинку для робітника з ремонту будинків

2.3 Визначення структури бази даних програми найму робітників для виконання ремонту будинку

База даних (БД) для програми найму робітників для виконання ремонту будинку розроблена з деталізацією всіх необхідних сутностей, а її структура на рівні проєкту визначена через створення програмного коду моделей і з застосуванням системи MySQL.

Власники будинків, які є користувачами програми, визначені інформацією про себе та пов'язаними з цим обліковими записами з утворенням таблиці Homeowner зі стовпцями:

- id – ідентифікатор власника будинку;
- homeownerindjango_id – обліковий запис власника будинку, яким його ідентифіковано в Django (зовнішній ключ, таблиця User);
- homeownerphone – номер телефону власника будинку;
- homeownerphoto – фотографія власника будинку;
- homeownerratingav – середнє значення рейтингу власника будинку.

Міста, збереження яких потрібне для визначення розташування будинків, роботи робітників, визначені таблицею БД Renovationcity зі стовпцями:

- id – ідентифікатор міста;
- renovationcityname – назва міста.

Для визначення структури представлення робітників, які займаються ремонтом будинків, було вирішено виділити окрім самого окремого робітника, який у програм як користувач може бути і не представлений, також бригади таких робітників для ремонту будинків, а також компанії, що займаються ремонтом будинків. Компанії необхідні для укрупнення бригад та фактично визначають ті випадки, коли декілька бригад виконують ремонт будинків у межах однієї організації. Для того щоб не визначити такі бригади з ремонту будинків окремо, не ускладнювати доступ, а синхронізувати роботу, вирішено виділити компанію з ремонту будинків, яка може включати одну або декілька

бригад для ремонту будинків, а кожна така бригада в свою чергу може складатися з певної кількості робітників, які безпосередньо сам ремонт будинків виконують. Якщо бригада одна, то вона все одно має узагальнення у вигляді компанії для ремонту будинків. Так само якщо навіть один окремих робітників має бажання працювати з програмою, то він визначає про себе відповідні дані, а далі вже визначається як єдиний робітників у бригаду з ремонту будинків, а бригада є єдиною бригадою компанії з ремонту будинків.

Компанії, що займаються ремонтом будинків, визначені таблицею Renovator зі стовпцями:

- id – ідентифікатор компанії, що займається ремонтом будинків;
- renovatorcompanyname – назва компанії, що займається ремонтом будинків;
- renovatorcompanylogo – логотип або інше зображення компанії, що займається ремонтом будинків;
- renovatorratingav – середнє значення рейтингу компанії, що займається ремонтом будинків;
- renovatorcompanyphone – номер телефону компанії, що займається ремонтом будинків;
- renovatorcompanydescription – опис компанії, що займається ремонтом будинків;
- renovatorcreatetime – час створення запису про компанію, що займається ремонтом будинків;
- renovationcity_id – місто, де знаходиться і діє ця компанія, що займається ремонтом будинків (зовнішній ключ, таблиця Renovationcity).

Документи на діяльність компаній, що займаються ремонтом будинків, визначені таблицею Renovatordocument зі стовпцями:

- id – ідентифікатор документу на діяльність компанії, що займається ремонтом будинків;
- renovator_id – компанія, що займається ремонтом будинків та якій видано цей документ на діяльність (зовнішній ключ, таблиця Renovator);

– documenttype – тип документу на діяльність компанії, що займається ремонтом будинків;

– documentfile – сам файл документу на діяльність компанії, що займається ремонтом будинків;

– documentdate – дата видачі документу на діяльність компанії, що займається ремонтом будинків;

– documentcreatetime – час збереження документу на діяльність компанії, що займається ремонтом будинків у програмі.

Фотографії результатів роботи компаній, що займаються ремонтом будинків, тобто фактично портфолію, визначені таблицею Renovatorphoto зі стовпцями:

– id – ідентифікатор фотографії результатів роботи компанії, що займається ремонтом будинків;

– renovator_id – компанія, що займається ремонтом будинків та результати роботи якої представлені на фотографії (зовнішній ключ, таблиця Renovator);

– renovatorphotofile – сама фотографія результатів роботи компанії, що займається ремонтом будинків.

Роботи з ремонту будинків, які загалом здатна виконувати компанія з ремонту будинків визначені таблицею БД Renovatorwork зі стовпцями:

– id – ідентифікатор роботи з ремонту будинків, яку загалом здатна виконувати компанія з ремонту будинків;

– renovator_id – компанія з ремонту будинків, в якій можливо реалізувати цю роботу для ремонту (зовнішній ключ, таблиця Renovator);

– renovationwork_id – ремонтна робота, яку здатна реалізувати ця компанія (зовнішній ключ, таблиця Renovationwork).

Тут важливо зазначити, що перелік робіт, які виконує загалом компанія та окрема бригада з ремонту будинків, може відрізнятися. При цьому вартість таких робіт є однаковою для будь-якої бригади в одній і тій самій компанії.

Вартість виконання ремонтних робіт в межах компанії з ремонту будинків визначена таблицею Renovatorworkprice зі стовпцями:

- id – ідентифікатор запису про вартість виконання ремонтних робіт в межах компанії з ремонту будинків;
- renovatorwork_id – робота з ремонту будинків, яку здатна виконувати ця компанія з ремонту будинків (зовнішній ключ, таблиця Renovatorwork);
- complexitylevel – рівень складності, за якого за цією ремонтною роботою вимагається така плата;
- renovationbaseprice – плата за виконання одиниці цієї ремонтної роботи бригадами цієї компанії з ремонту будинків.

Робітники, які виконують ремонт будинків, є користувачами програми та входять до складу бригад з ремонту будинків, визначені таблицею Renovatorcrewaccount зі стовпцями:

- id – ідентифікатор робітника, який виконує ремонт будинків, є користувачем програми та входить до складу бригад з ремонту будинків;
- renovatorcrewaccountindjango_id – обліковий запис робітника, який виконує ремонт будинків, є користувачем програми та входить до складу бригад з ремонту будинків, у Django (зовнішній ключ, таблиця User);
- renovatorcrew_id – бригада з ремонту будинку, в якій працює цей робітник (зовнішній ключ, таблиця Renovatorcrew);
- renovatorcrewaccountphone – номер телефону робітника, який виконує ремонт будинків, є користувачем програми та входить до складу бригад з ремонту будинків;
- renovatorcrewaccountphoto – фотографія робітника, який виконує ремонт будинків, є користувачем програми та входить до складу бригад з ремонту будинків.

Бригади з ремонту будинків визначені таблицею Renovatorcrew зі стовпцями:

- id – ідентифікатор бригади з ремонту будинків;

- renovator_id – ідентифікатор компанії з ремонту будинків, яку представляє ця бригада (зовнішній ключ, таблиця Renovator);
- renovatorcrewname – назва бригади з ремонту будинків;
- renovatorcrewphoto – фотографія бригади з ремонту будинків;
- renovatorcrewparticipants – кількість робітників у цій бригаді з ремонту будинків.

Доступний час роботи бригади з ремонту будинків визначений таблицею Renovatorcrewtime зі стовпцями:

- id – ідентифікатор доступного часу роботи бригади з ремонту будинків;
- renovatorcrew_id – бригада з ремонту будинків, у якої доступний цей час для виконання робіт (зовнішній ключ, таблиця Renovatorcrew);
- renovationstarttime – можливий час початку роботи бригади з ремонту будинків;
- renovationendtime – можливий час закінчення роботи бригади з ремонту будинків.

Роботи з ремонту будинків, які здатна виконувати конкретна бригада, а не компанія загалом, визначені таблицею Renovatorcrewwork зі стовпцями:

- id – ідентифікатор роботи з ремонту будинків, які здатна виконувати конкретна бригада;
- renovatorwork_id – ремонтна робота, яку здатні виконувати робітники цієї компанії з ремонту будинків (зовнішній ключ, таблиця Renovatorwork);
- renovatorcrew_id – бригада з ремонту будинків, яка здатна цю роботу виконувати (зовнішній ключ, таблиця Renovatorcrew).

Групи робіт з ремонту будинків визначені таблицею Renovationworkgroup зі стовпцями:

- id – ідентифікатор групи робіт з ремонту будинків;
- renovationworkgroupname – назва групи робіт з ремонту будинків.

Роботи з ремонту будинків визначені таблицею Renovationwork зі стовпцями:

- id – ідентифікатор роботи з ремонту будинків;
- renovationworkgroup_id – група робіт з ремонту будинків, до якої належить ця робота (зовнішній ключ, таблиця Renovationworkgroup);
- renovationworkname – назва роботи з ремонту будинків;
- renovationbaseunit – одиниця виміру робіт з ремонту будинків (стандартно метр квадратний, може бути просто одиниця).

Дизайнерські стилі ремонту будинків визначені таблицею Homerovationstyle зі стовпцями:

- id – ідентифікатор дизайнерського стилю ремонту будинку;
- homerovationstylename – назва дизайнерського стилю ремонту будинку.

Необхідні власникам будинків ремонти визначені таблицею Homerovation зі стовпцями:

- id – ідентифікатор необхідного власнику будинку ремонту;
- homeowner_id – власник будинку, якому необхідно виконати цей ремонт будинку (зовнішній ключ, таблиця Homeowner);
- homerovationminprice – мінімальна вартість, яку готовий сплатити за виконання цього ремонту свого будинку власник;
- homerovationmaxprice – максимальна допустима вартість, яку готовий сплатити за виконання цього ремонту свого будинку власник;
- homerovationstartdate – необхідна найраніша дата початку виконання ремонту будинку власника;
- homerovationfinishdate – необхідна найпізніша дата завершення виконання ремонту будинку власника;
- homerovationdescription – опис особливостей виконання ремонту цього будинку власника;
- renovationcity_id – місто, де знаходиться цей будинок власника, ремонт якого має бути виконаний (зовнішній ключ, таблиця Renovationcity);
- homerovationcreatetime – час збереження у програмі цього оголошення про необхідний власнику будинку ремонт;

– `homerenovationaddress` – адреса цього будинку власника, ремонт якого має бути виконаний;

– `homerenovationstyle_id` – дизайнерський стиль, у якому потрібно виконати ремонт цього будинку (зовнішній ключ, таблиця `Homerenovationstyle`).

Кімнати будинку, в яких потрібно виконати ремонт, визначені таблицею `Homerenovationroom` зі стовпцями:

– `id` – ідентифікатор кімнати будинку, в якій потрібно виконати ремонт;

– `homerenovation_id` – будинок, ремонт якого потрібно виконати для власника та в якому знаходиться ця кімната (зовнішній ключ, таблиця `Homerenovation`);

– `homeroomtype` – тип кімнати будинку, в якій потрібно виконати ремонт;

– `homeroomdescription` – опис особливостей кімнати будинку, в якій потрібно виконати ремонт, та ремонту в цій кімнаті;

– `homerenovationstartdate` – необхідна найраніша дата початку ремонту в цій кімнаті;

– `homerenovationfinishdate` – необхідна найпізніша дата завершення ремонту в цій кімнаті.

Фотографії кімнат будинку, в яких потрібно виконати ремонт, визначені таблицею `Homerenovationroomphoto` зі стовпцями:

– `id` – ідентифікатор фотографії кімнати будинку, в якій потрібно виконати ремонт;

– `homerenovationroom_id` – кімната будинку, в якій потрібно виконати ремонт та яка зображена на фотографії (зовнішній ключ, таблиця `Homerenovationroom`);

– `homeroomphoto` – сама фотографія кімнати будинку, в якій потрібно виконати ремонт;

– `homeroomphotodescription` – опис особливостей, зображених на фотографії кімнати будинку, в якій потрібно виконати ремонт.

Фотографії будинку, в якому власнику потрібно виконати ремонт, визначені таблицею `Homerenovationphoto` зі стовпцями:

- `id` – ідентифікатор фотографії будинку, в якому власнику потрібно виконати ремонт;
- `homerenovation_id` – оголошення про необхідність ремонту будинку, який зображений на фотографії (зовнішній ключ, таблиця `Homerenovation`);
- `homephoto` – сама фотографія будинку, в якому потрібно виконати ремонт;
- `homephotodescription` – опис особливостей, зображених на фотографії будинку, в якому потрібно виконати ремонт.

Роботи з ремонту, які за думкою власника необхідно виконати для його будинку, визначені таблицею `Homerenovationwork` зі стовпцями:

- `id` – ідентифікатор роботи з ремонту, яку за думкою власника необхідно виконати для його будинку;
- `homerenovation_id` – необхідний власнику ремонт будинку, до якого відноситься ця робота (зовнішній ключ, таблиця `Homerenovation`);
- `renovationwork_id` – робота з ремонту, яку необхідно виконати (на думку власника) для будинку (зовнішній ключ, таблиця `Renovationwork`);
- `renovationunitnumber` – кількість базових одиниць, в яку оцінюється масштаб виконання цієї роботи з ремонту для всього будинку;
- `renovationcomment` – опис особливостей виконання цієї ремонтної роботи для будинку.

Заявки на виконання ремонтних робіт за будинком визначені таблицею `Homerenovationapplication` зі стовпцями:

- `id` – ідентифікатор заявки на виконання ремонтних робіт за будинком;
- `homerenovation_id` – необхідний власнику ремонт будинку, якого стосується заявка (зовнішній ключ, таблиця `Homerenovation`);
- `renovatorcrew_id` – бригада з ремонту будинку, яка має виконати ремонт за цією заявкою (зовнішній ключ, таблиця `Renovatorcrew`);

– `applicationsourcetype` – тип подачі заявки на виконання ремонтних робіт за будинком: це може бути заявка, ініційована самим власником будинку без визначення конкретної бригади, це може бути заявка, ініційована самим власником будинку з визначенням конкретної бригади, це може бути заявка, ініційована самою ремонтною бригадою;

– `renovationstarttime` – запланований або фактичний час початку ремонту за цим будинком власника;

– `renovationendtime` – запланований або фактичний час завершення ремонту за цим будинком власника;

– `renovationproposedprice` – пропонована вартість ремонту за цим будинком власника;

– `renovationprice` – фактична вартість ремонту за цим будинком власника;

– `renovationcomment` – коментар до виконання ремонту за цим будинком власника.

Ремонтні роботи за такими заявками з таблиці `Homerenovationapplication`, які визначені вже самою бригадою з ремонту будинків, а не власником (фактично кошторис робіт), визначені таблицею `Homerenovationapplicationwork` зі стовпцями:

– `id` – ідентифікатор ремонтної роботи за заявкою на виконання ремонтних робіт за будинком;

– `homerenovationapplication_id` – заявка на виконання ремонтних робіт за будинком, якої стосується ця робота (зовнішній ключ, таблиця `Homerenovationapplication`);

– `renovatorcrewwork_id` – ремонтна робота, яка визначена бригадою для виконання за цією заявкою (зовнішній ключ, таблиця `Renovatorcrewwork`);

– `renovationunitnumber` – кількість базових одиниць виконання цієї роботи з ремонту, яку необхідно застосувати під час ремонту всього будинку;

– `renovationcomment` – коментар до виконання цієї ремонтної роботи за заявкою стосовно будинку.

Стани, в яких знаходилась заявка на виконання ремонтних робіт за будинком, визначені таблицею `Homerenovationapplicationstatus` зі стовпцями:

- `id` – ідентифікатор стану, в якому знаходилась заявка на виконання ремонтних робіт за будинком;
- `homerenovationapplication_id` – заявка на виконання ремонтних робіт, яка знаходилась в цьому стані (зовнішній ключ, таблиця `Homerenovationapplication`);
- `renovationstatus` – стан, який під час цього збереження було встановлено заявці на виконання ремонтних робіт;
- `applicationstatussettype` – тип встановлення стану: власником будинку або бригадою з ремонту будинків.

Виконані оцінювання компаній з ремонту будинків визначені таблицею `Renovatorrating` зі стовпцями:

- `id` – ідентифікатор виконаного оцінювання компанії з ремонту будинків;
- `homerenovationapplication_id` – заявка на ремонт будинку, за якою було виконано це оцінювання (зовнішній ключ, таблиця `Homerenovationapplication`);
- `plusside` – позитивні сторони, виявлені власником будинку стосовно компанії з ремонту будинків, бригади, яка її представляла, виконаних робіт;
- `minusside` – негативні сторони, виявлені власником будинку стосовно компанії з ремонту будинків, бригади, яка її представляла, виконаних робіт;
- `ratingcreatetime` – час оцінювання компанії з ремонту будинків власником будинку;
- `renovoverallpoint` – загальна оцінка компанії з ремонту будинків, виставлена власником будинку;
- `renovqualitypoint` – оцінка якості виконаних ремонтних робіт компанії, виставлена власником будинку;
- `renovdurationpoint` – оцінка дотриманню строків при виконанні ремонтних робіт компанією, виставлена власником будинку;

– renovattitudepoint – оцінка відношенню при виконанні ремонтних робіт компанією, виставлена власником будинку;

– renovpricerpoint – оцінка відповідності вартості виконаних ремонтних робіт якості, виставлена власником будинку;

– renovphoto – фотографія результатів виконання ремонтних робіт.

Виконані оцінювання власників будинків компаніями з ремонту визначені таблицею БД Homeownerrating зі стовпцями:

– id – ідентифікатор виконаного оцінювання власника будинку компанією з ремонту будинків;

– homerenovationapplication_id – заявка на ремонт будинку, за якою було виконано це оцінювання (зовнішній ключ, таблиця Homerenovationapplication);

– plusside – позитивні сторони, виявлені компанією з ремонту будинків стосовно власника будинку;

– minusside – негативні сторони, виявлені компанією з ремонту будинків стосовно власника будинку;

– ratingcreatetime – час оцінювання власника будинку компанією з ремонту будинків власником будинку;

– renovoverallpoint – оцінка власника будинку компанією з ремонту будинків.

Закладки на необхідних власникам будинків ремонтах визначені таблицею Homerenovationbookmark зі стовпцями:

– id – ідентифікатор закладки на необхідному власнику будинку ремонті;

– homerenovation_id – ремонт, необхідний власнику будинку, на якому створено закладку (зовнішній ключ, таблиця Homerenovation);

– renovatorcrewaccount_id – користувач, що є робітником, який представляє бригаду з ремонту будинків і який створив цю закладку (зовнішній ключ, таблиця Renovatorcrewaccount);

– renovbookmarktime – час створення закладки на необхідному власнику будинку ремонті.

Закладки на компаніях з ремонту будинків визначені таблицею Renovatorbookmark зі стовпцями:

- id – ідентифікатор закладки на компанії з ремонту будинків;
- renovator_id – компанія з ремонту будинків, на якій було створено закладку власником будинку (зовнішній ключ, таблиця Renovator);
- homeowner_id – власник будинку, який створив закладку на компанії з ремонту будинку (зовнішній ключ, таблиця Homeowner);
- renovbookmarktime – час створення закладки на компанії з ремонту будинку (рис. 2.4).

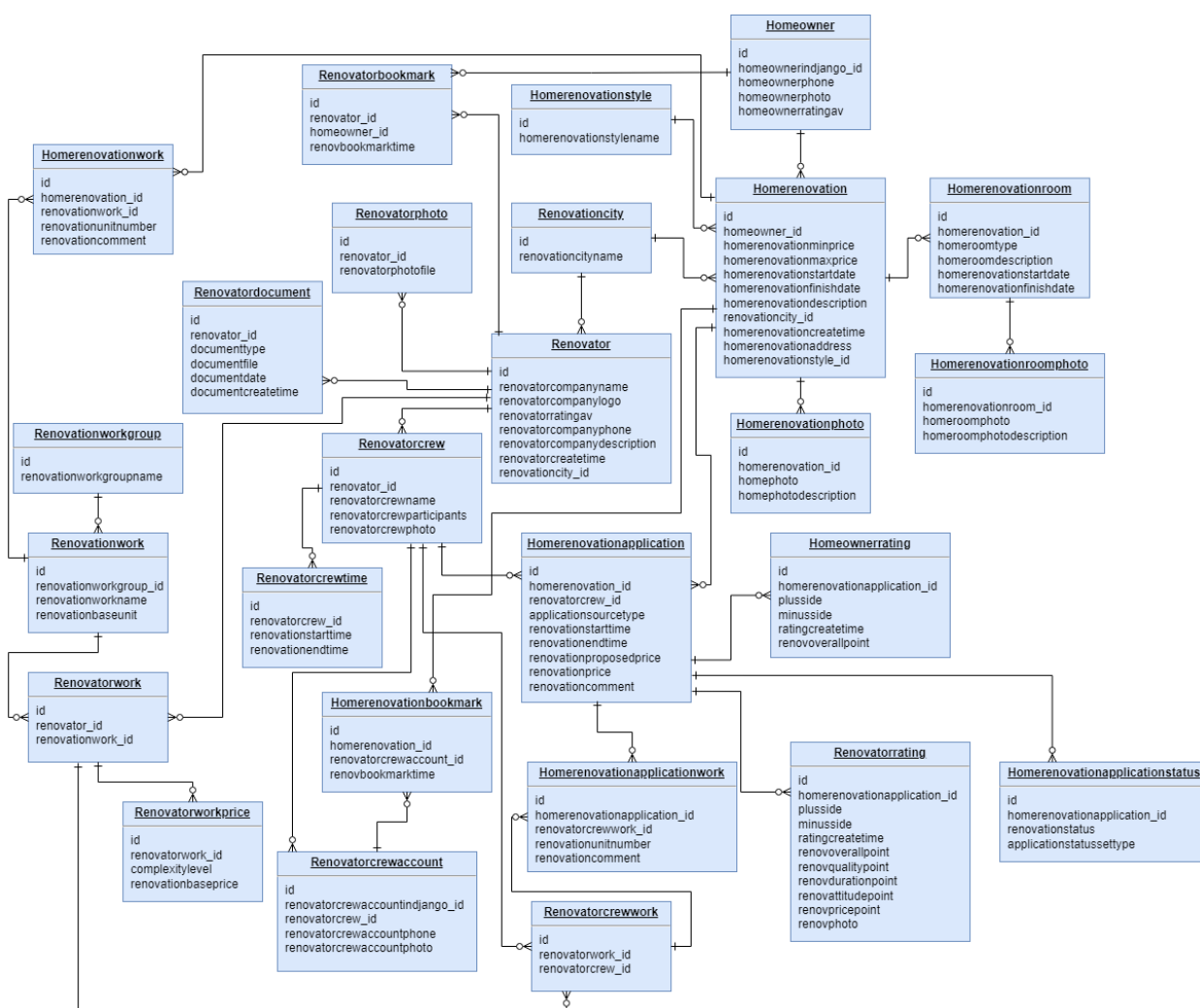


Рисунок 2.4 – Схема БД програми найму робітників для виконання ремонту будинку

2.4 Висновки за розділом 2

У цьому розділі як результат проєктування програми найму робітників для виконання ремонту будинку промодельовано вимоги до програми зі сторони користувача до авторизації, власника будинку та робітника з ремонту будинку, визначено структуру БД, обрано засоби розробки.

3 РОЗРОБКА ПРОГРАМИ

3.1 Визначення послідовності роботи з програмою найму робітників для виконання ремонту будинку

Для забезпечення послідовної роботи користувачів у всіх ролях в програмі найму робітників для виконання ремонту будинку зі всіма наявними функціональними можливостями було створено відповідні варіанти меню, що узагальнено на функціональній схемі (рис. 3.1).

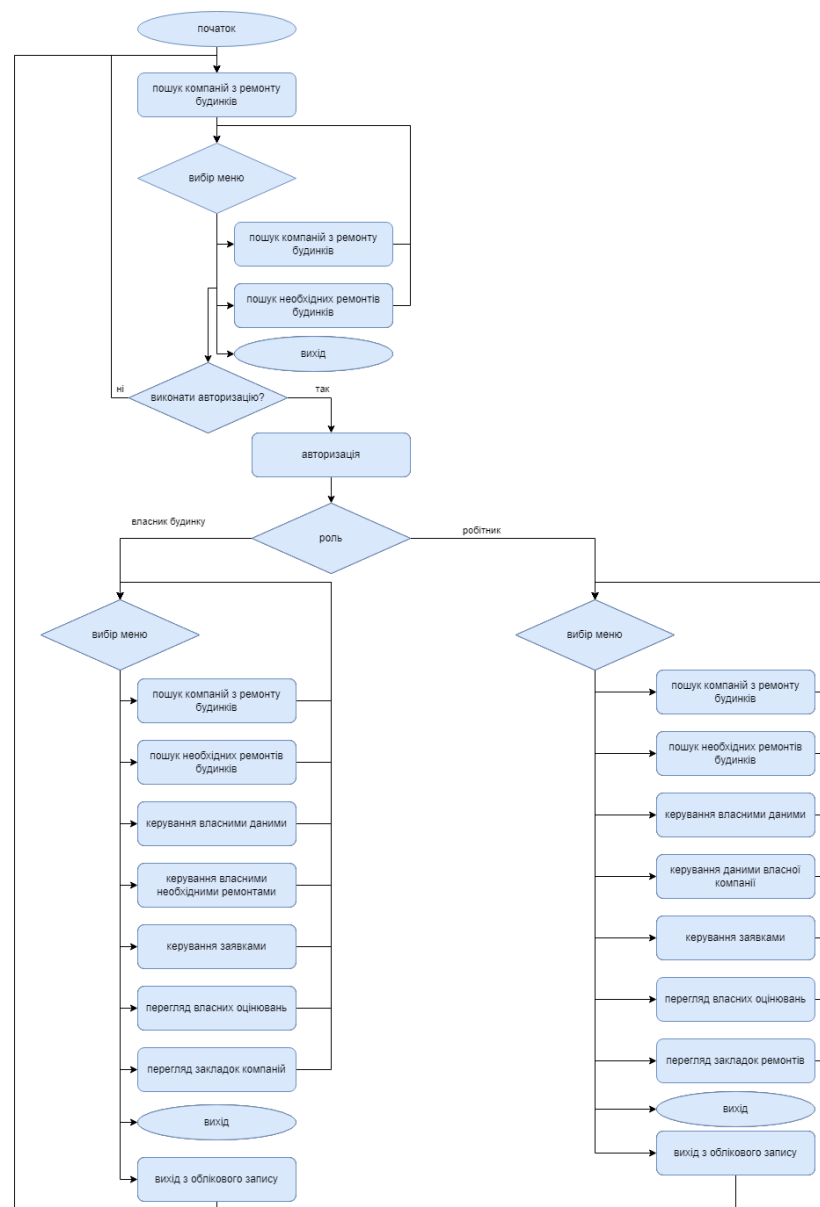


Рисунок 3.1 – Функціональна схема програми найму робітників для виконання ремонту будинку

3.2 Особливості реалізації програми найму робітників для виконання ремонту будинку

ВЗ найму робітників для виконання ремонту будинку реалізовано через фреймворк Django, як і було обрано і заплановано раніше.

Перегляд переліку необхідних ремонтів будинків власника реалізовано функцією представлення `homeowner_renovations_view` з параметром `homeowner_umat`, який задає логін власника будинків (або авторизаційні дані, якщо не задано).

Усі представлення мають відповідні шаблони вебсторінок, а також шаблон сторінки помилки, що відображає інформацію про помилку, яку було виявлено під час роботи з коректною обробкою такої ситуації. Усі шаблони вебсторінок створені з використанням вільного наблону [6].

Перегляд оголошення про необхідність ремонту будинку реалізовано функцією представлення `homeowner_renovation_view` з параметром `homerovation_id`, який задає ідентифікатор оголошення про необхідність ремонту будинку.

Перегляд оцінювань власника будинку реалізовано функцією представлення `homeowner_ratings_view` з параметром `homeowner_umat`, який задає логін власника будинку (або авторизаційні дані, якщо логін не задано).

Перегляд оцінювань компанії з ремонту будинків реалізовано функцією представлення `renovator_ratings_view` з параметром `renovator_id`, який задає ідентифікатор компанії з ремонту будинків (або компанію з авторизаційних даних робітника з ремонту будинків, якщо не задано).

Додавання кімнати до даних ремонту будинку реалізовано функцією представлення `homerovationroom_add_view` з параметром `homerovation_id`, який задає ідентифікатор самого необхідного ремонту будинку, працюючи на основі класу форми `HomerovationroomForm`.

Оновлення кімнати з даних ремонту будинку реалізовано функцією представлення `homerenovationroom_change_view` з параметром `homerenovationroom_id`, що задає ідентифікатор кімнати з плану будинку, працюючи на основі класу форми `HomerenovationroomForm`.

Додавання роботи до плану ремонту будинку реалізовано функцією представлення `homerenovationwork_add_view` з параметром `homerenovation_id`, який задає ідентифікатор самого необхідного ремонту будинку, працюючи на основі класу форми `HomerenovationworkForm`.

Оновлення роботи з плану ремонту будинку реалізовано функцією представлення `homerenovationwork_change_view` з параметром `homerenovationwork_id`, який задає ідентифікатор самої роботи з плану ремонту будинку, працюючи на основі класу форми `HomerenovationworkForm`.

Додавання роботи до плану ремонту будинку за заявкою реалізовано функцією представлення `homerenovationapplicationwork_add_view` з параметром `homerenovationapplication_id`, який задає ідентифікатор заявки на виконання ремонту будинку, працюючи на основі класу форми `HomerenovationapplicationworkForm`.

Оновлення роботи з плану ремонту будинку за заявкою реалізовано функцією представлення `homerenovationapplicationwork_change_view` з параметром `homerenovationapplicationwork_id`, який задає ідентифікатор роботи з плану ремонту будинку за заявкою, працюючи на основі класу форми `HomerenovationapplicationworkForm`.

Вилучення роботи з плану ремонту будинку за заявкою реалізовано функцією представлення `homerenovationapplicationwork_delete_view` з параметром `homerenovationapplicationwork_id`, який задає ідентифікатор роботи з плану ремонту будинку за заявкою

Додавання часового лоту для роботи бригади з ремонту будинків реалізовано функцією представлення `renovatorcrewtime_add_view`, працюючи на основі класу форми `RenovatorcrewtimeForm`.

Оновлення часового лоту для роботи бригади з ремонту будинків реалізовано функцією представлення `renovatorcrewtime_change_view` з параметром `renovatorcrewtime_id`, який задає ідентифікатор часового лоту для роботи бригади з ремонту будинків, працюючи на основі класу форми `RenovatorcrewtimeForm`.

Вилучення кімнати з даних ремонту будинку реалізовано функцією представлення `homerenovationroom_delete_view` з параметром `homerenovationroom_id`, що задає ідентифікатор кімнати з плану будинку.

Оцінювання компанії з ремонту будинків реалізовано функцією представлення `renovatorrating_add_view` з параметром `homerenovationapplication_id`, який задає ідентифікатор заявки на ремонт будинку, на основі якої відбувається оцінювання.

Оцінювання власника будинку реалізовано функцією представлення `homeownerrating_add_view` з параметром `homerenovationapplication_id`, який задає ідентифікатор заявки на ремонт будинку, на основі якої відбувається оцінювання.

Перегляд переліку бригад з ремонту будинків компанії реалізовано функцією представлення `renovator_crews_view` з параметром `renovator_id`, який задає ідентифікатор компанії з ремонту будинків.

Перегляд даних бригади з ремонту будинків реалізовано функцією представлення `renovator_crew_view` з параметром `renovatorcrew_id`, який задає ідентифікатор бригади з ремонту будинків.

Перегляд переліку ремонтних робіт, які здатна виконувати компанія, реалізовано функцією представлення `renovator_works_view` з параметром `renovator_id`, який задає ідентифікатор компанії з ремонту будинків.

Перегляд даних власника будинку реалізовано функцією представлення `homeowner_info_view` з параметром `homeowner_uname`, який задає логін власника будинку.

Перегляд даних компанії з ремонту будинків реалізовано функцією представлення `renovator_info_view` з параметром `renovator_id`, який задає ідентифікатор компанії з ремонту будинків.

Перегляд доступних для роботи бригад у компанії з ремонту будинків на заданий час реалізовано функцією представлення `renovator_crews_access_view` з параметрами `renovator_id`, який задає ідентифікатор компанії з ремонту будинків, та `renovationdate`, що задає дату виконання ремонту.

Визначення даних власника будинку реалізовано функцією представлення `homeowner_set_view`, працюючи на основі класу форми `HomeownerForm`.

Створення оголошення про необхідний ремонт будинку реалізовано функцією представлення `homerenovation_add_view`, працюючи на основі класу форми `HomerenovationForm`.

Оновлення оголошення про необхідний ремонт будинку реалізовано функцією представлення `homerenovation_change_view` з параметром `homerenovation_id`, що задає ідентифікатор оголошення про необхідний ремонт будинку, працюючи на основі класу форми `HomerenovationForm`.

Вилучення оголошення про необхідний ремонт будинку реалізовано функцією представлення `homerenovation_delete_view` з параметром `homerenovation_id`, що задає ідентифікатор оголошення про необхідний ремонт будинку.

Вилучення роботи з плану ремонту будинку реалізовано функцією представлення `homerenovationwork_delete_view` з параметром `homerenovationwork_id`, який задає ідентифікатор самої роботи з плану ремонту будинку.

Управління станом заявки на виконання ремонту будинку реалізовано функцією представлення `homerenovationapplication_change_view` з параметром `homerenovationapplication_id`, що задає заявку на виконання ремонту будинку.

Подання заявки на виконання ремонту будинку реалізовано функцією представлення `homerenovationapplication_add_view` з параметром `homerenovation_id`, який задає ідентифікатор ремонту будинку, за яким створюється заявка.

Перегляд заявки на виконання ремонту будинку реалізовано функцією представлення `homerenovationapplication_info_view` з параметром `homerenovationapplication_id`, що задає заявку на виконання ремонту будинку.

Перегляд заявок на виконання ремонту за будинком реалізовано функцією представлення `homerenovation_applications_view` з параметром `renovation_id`, що задає ідентифікатор ремонту за будинком.

Перегляд власних поданих заявок на виконання ремонту будинку реалізовано функцією представлення `own_homerenovationapplications_view`.

Створення закладки за компанією з ремонту будинку та її вилучення в залежності від наявності такої закладки реалізовано функцією представлення `homerenovationbookmark_add_view` з параметром `homerenovation_id`, який задає ідентифікатор необхідного ремонту будинку.

Перегляд станів, які визначались за заявкою на виконання ремонту будинку, реалізовано функцією представлення `homerenovationapplication_statuses_view` з параметром `homerenovationapplication_id`, який задає ідентифікатор заявки на виконання ремонту будинку.

Перегляд оцінених компаній з ремонту будинків власником будинку реалізовано функцією представлення `my_renovatorratings_view`.

Перегляд переліку компаній з ремонту будинку в закладках реалізовано функцією представлення `my_renovatorbookmarks_view`.

Пошук оголошень про необхідність ремонту будинку реалізовано функцією `homerenovation_search_view` з параметрами, заданими ключами GET-запиту, включаючи `homerenovationprice`, що задає мінімальну прийнятну вартість ремонту будинку, `homerenovationstartdate` і `homerenovationenddate`, що

задають проміжок дат виконання ремонту, `renovationstyle`, що задає дизайнерський стиль.

Визначення власної компанії з ремонту будинків реалізовано функцією представлення `renovator_set_view` з параметром `renovator_id`, який задає ідентифікатор компанії з ремонту будинків.

Оновлення даних компанії з ремонту будинків реалізовано функцією представлення `renovator_change_view` з параметром `renovator_id`, який задає ідентифікатор компанії з ремонту будинків.

Оновлення даних робітника з ремонту будинків реалізовано функцією представлення `renovatorcrewaccount_set_view`.

Вилучення часового лоту для роботи бригади з ремонту будинків реалізовано функцією представлення `renovatorcrewtime_delete_view` з параметром `renovatorcrewtime_id`, який задає ідентифікатор часового лоту для роботи бригади з ремонту будинків

Додавання ремонтної роботи до можливостей бригади реалізовано функцією представлення `add_renovatorcrew_work_view` з параметром `renovatorwork_id`, що задає ідентифікатор ремонтної роботи.

Вилучення ремонтної роботи з можливостей бригади реалізовано функцією представлення `delete_renovatorcrew_work_view` з параметром `renovatorcrewwork_id`, що задає ідентифікатор ремонтної роботи у можливостях бригади.

Перегляд даних робітника з ремонту будинків реалізовано функцією представлення `renovatorcrewaccount_info_view`.

Перегляд складу робітників бригади з ремонту будинків реалізовано функцією представлення `renovatorcrew_accounts_view`.

Перегляд заявок на виконання ремонту будинку реалізовано функцією представлення `homerenovation_application_view` з параметром `homerenovation_id`, який задає ідентифікатор ремонту будинку.

Перегляд переліку ремонтів будинків у закладках реалізовано функцією представлення `my_homerenovationbookmarks_view`.

Перегляд оцінених власників будинків бригадою з ремонту будинків реалізовано функцією представлення `homeownerratings_view`.

Створення закладки за ремонтом будинку та її вилучення в залежності від наявності такої закладки реалізовано функцією представлення `renovatorbookmark_add_view` з параметром `renovator_id`, який задає ідентифікатор компанії з ремонту будинку.

Модель власників будинків, які є користувачами програми, реалізована класом `Homeowner`.

Поле `homeownerindjango_id` у класі моделі `Homeowner` представляє обліковий запис власника будинку, яким його ідентифіковано в Django, реалізоване типом `OneToOneField` зі взаємодією зі стандартною моделлю Django User.

Поле `homeownerphone` у класі моделі `Homeowner` представляє номер телефону власника будинку, реалізований типом `CharField`.

Поле `homeownerphoto` у класі моделі `Homeowner` представляє фотографію власника будинку, реалізовану типом `ImageField`.

Поле `homeownerratingav` у класі моделі `Homeowner` представляє середнє значення рейтингу власника будинку, реалізоване типом `FloatField`.

Модель міст, збереження яких потрібне для визначення розташування будинків, роботи робітників, реалізована класом `Renovationcity`.

Поле `renovationcityname` у класі моделі `Renovationcity` представляє назву міста.

Модель компанії, що займається ремонтом будинків, реалізована класом `Renovator`.

Поле `renovatorcompanyname` у класі моделі `Renovator` представляє назву компанії, що займається ремонтом будинків, реалізовану типом `CharField`.

Поле `renovatorcompanylogo` у класі моделі `Renovator` представляє логотип або інше зображення компанії, що займається ремонтом будинків, реалізовану типом `ImageField`.

Поле `renovatorratingav` у класі моделі `Renovator` представляє середнє значення рейтингу компанії, що займається ремонтом будинків, реалізовану типом `FloatField`.

Поле `renovatorcompanyphone` у класі моделі `Renovator` представляє номер телефону компанії, що займається ремонтом будинків, реалізований типом `CharField`.

Поле `renovatorcompanydescription` у класі моделі `Renovator` представляє опис компанії, що займається ремонтом будинків, реалізований типом `CharField`.

Поле `renovatorcreatetime` у класі моделі `Renovator` представляє час створення запису про компанію, що займається ремонтом будинків, реалізований типом `DateTimeField`.

Поле `renovationcity_id` у класі моделі `Renovator` представляє місто, де знаходиться і діє ця компанія, що займається ремонтом будинків, реалізоване типом `ForeignKey` зі взаємодією з моделлю `Renovationcity`.

Модель документів на діяльність компаній, що займаються ремонтом будинків, реалізована класом `Renovatordocument`.

Поле `renovator_id` у класі моделі `Renovatordocument` представляє компанію, що займається ремонтом будинків та якій видано цей документ на діяльність, реалізовану типом `ForeignKey` зі взаємодією з моделлю `Renovator`.

Поле `documenttype` у класі моделі `Renovatordocument` представляє тип документу на діяльність компанії, що займається ремонтом будинків, реалізований типом `PositiveSmallIntegerField`.

Поле `documentfile` у класі моделі `Renovatordocument` представляє сам файл документу на діяльність компанії, що займається ремонтом будинків, реалізований типом `FileField`.

Поле `documentdate` у класі моделі `Renovatordocument` представляє дату видачі документу на діяльність компанії, що займається ремонтом будинків, реалізовану типом `DateField`.

Поле `documentcreatetime` у класі моделі `Renovatordocument` представляє час збереження документу на діяльність компанії, що займається ремонтом будинків у програмі, реалізований типом `DateTimeField`.

Модель фотографії результатів роботи компаній, що займаються ремонтом будинків, реалізована класом `Renovatorphoto`.

Поле `renovator_id` у класі моделі `Renovatordocument` представляє компанію, що займається ремонтом будинків та результати роботи якої представлені на фотографії, реалізовану типом `ForeignKey` зі зв'язком з моделлю `Renovator`.

Поле `renovatorphotofile` у класі моделі `Renovatordocument` представляє саму фотографію результатів роботи компанії, що займається ремонтом будинків, реалізовану типом `ImageField`.

Модель робіт з ремонту будинків, які загалом здатна виконувати компанія з ремонту будинків, реалізована класом `Renovatorwork`.

Поле `renovator_id` у класі моделі `Renovatorwork` представляє компанію з ремонту будинків, в якій можливо реалізувати цю роботу для ремонту, реалізовану типом `ForeignKey` зі зв'язком з моделлю `Renovator`.

Поле `renovationwork_id` у класі моделі `Renovatorwork` представляє ремонтну роботу, яку здатна реалізувати ця компанія, реалізовану типом `ForeignKey` зі зв'язком з моделлю `Renovationwork`.

Модель вартості виконання ремонтних робіт в межах компанії з ремонту будинків реалізована класом `Renovatorworkprice`.

Поле `renovatorwork_id` у класі моделі `Renovatorworkprice` представляє роботу з ремонту будинків, яку здатна виконувати ця компанія з ремонту будинків, реалізовану типом `ForeignKey` зі зв'язком з моделлю `Renovatorwork`.

Поле `complexitylevel` у класі моделі `Renovatorworkprice` представляє рівень складності, за якого за цією ремонтною роботою вимагається така плата, реалізований типом `PositiveSmallIntegerField`.

Поле `renovationbaseprice` у класі моделі `Renovatorworkprice` представляє плату за виконання одиниці цієї ремонтної роботи бригадами цієї компанії з ремонту будинків, реалізовану типом `FloatField`.

Модель робітників, які виконують ремонт будинків, є користувачами програми та входять до складу бригад з ремонту будинків, реалізована класом `Renovatorcrewaccount`.

Поле `renovatorcrewaccountindjango_id` у класі моделі `Renovatorcrewaccount` представляє обліковий запис робітника, який виконує ремонт будинків, є користувачем програми та входить до складу бригад з ремонту будинків, у Django, реалізований типом `OneToOneField` зі зв'язком з моделлю `User`.

Поле `renovatorcrew_id` у класі моделі `Renovatorcrewaccount` представляє бригаду з ремонту будинку, в якій працює цей робітник реалізований типом `ForeignKey` зі зв'язком з моделлю `Renovatorcrew`.

Поле `renovatorcrewaccountphone` у класі моделі `Renovatorcrewaccount` представляє номер телефону робітника, який виконує ремонт будинків, є користувачем програми та входить до складу бригад з ремонту будинків, реалізований типом `CharField`.

Поле `renovatorcrewaccountphoto` у класі моделі `Renovatorcrewaccount` представляє фотографію робітника, який виконує ремонт будинків, є користувачем програми та входить до складу бригад з ремонту будинків, реалізовану типом `ImageField`.

Модель бригади з ремонту будинків реалізована класом `Renovatorcrew`.

Поле `renovator_id` у класі моделі `Renovatorcrew` представляє ідентифікатор компанії з ремонту будинків, реалізований типом `ForeignKey` зі зв'язком з моделлю `Renovator`.

Поле `renovatorcrewname` у класі моделі `Renovatorcrew` представляє назву бригади з ремонту будинків, реалізовану типом `CharField`.

Поле `renovatorcrewparticipants` у класі моделі `Renovatorcrew` представляє кількість робітників у цій бригаді з ремонту будинків, реалізовану типом `PositiveIntegerField`.

Поле `renovatorcrewphoto` у класі моделі `Renovatorcrew` представляє фотографію бригади з ремонту будинків, реалізований типом `ImageField`.

Поле `renovatorcrewworks` у класі моделі `Renovatorcrew` представляє роботи з ремонту будинків, які здатна виконувати ця конкретно бригада, реалізовані типом `ManyToManyField` зі зв'язком до моделі `Renovatorwork`.

Модель доступного часу роботи бригади з ремонту будинків реалізована класом `Renovatorcrewtime`.

Поле `renovatorcrew_id` у класі моделі `Renovatorcrewtime` представляє бригаду з ремонту будинків, у якій доступний цей час для виконання робіт, реалізовану типом `ForeignKey` зі зв'язком з моделлю `Renovatorcrew`.

Поле `renovationstarttime` у класі моделі `Renovatorcrewtime` представляє можливий час початку роботи бригади з ремонту будинків, реалізований типом `DateTimeField`.

Поле `renovationendtime` у класі моделі `Renovatorcrewtime` представляє можливий час закінчення роботи бригади з ремонту будинків, реалізований типом `DateTimeField`.

Модель груп робіт з ремонту будинків реалізована класом `Renovationworkgroup`.

Поле `renovationworkgroupname` у класі моделі `Renovationworkgroup` представляє назву групи робіт з ремонту будинків, реалізовану типом `CharField`.

Модель робіт з ремонту будинків реалізована класом `Renovationwork`.

Поле `renovationworkgroup_id` у класі моделі `Renovationwork` представляє групу робіт з ремонту будинків, до якої належить ця робота, реалізовану типом `ForeignKey` зі зв'язком з моделлю `Renovationworkgroup`.

Поле `renovationworkname` у класі моделі `Renovationwork` представляє назву роботи з ремонту будинків, реалізовану типом `CharField`.

Поле `renovationbaseunit` у класі моделі `Renovationwork` представляє одиницю виміру робіт з ремонту будинків, реалізовану типом `CharField`.

Модель дизайнерських стилів ремонту будинків реалізована класом `Homerenovationstyle`.

Поле `homerenovationstylename` у класі моделі `Renovationwork` представляє назву дизайнерського стилю ремонту будинку, реалізовану типом `CharField`.

Модель необхідних власникам будинків ремонтів реалізована класом `Homerenovation`.

Поле `homeowner_id` у класі моделі `Homerenovation` представляє власника будинку, якому необхідно виконати цей ремонт будинку, реалізованого типом `ForeignKey` зі зв'язком з моделлю `Homeowner`.

Поле `homerenovationminprice` у класі моделі `Homerenovation` представляє мінімальну вартість, яку готовий сплатити за виконання цього ремонту свого будинку власник, реалізовану типом `FloatField`.

Поле `homerenovationmaxprice` у класі моделі `Homerenovation` представляє максимальну допустиму вартість, яку готовий сплатити за виконання цього ремонту свого будинку власник, реалізовану типом `FloatField`.

Поле `homerenovationstartdate` у класі моделі `Homerenovation` представляє необхідну найранішу дату початку виконання ремонту будинку власника, реалізовану типом `DateField`.

Поле `homerenovationfinishdate` у класі моделі `Homerenovation` представляє необхідну найпізнішу дату завершення виконання ремонту будинку власника, реалізовану типом `DateField`.

Поле `homerenovationdescription` у класі моделі `Homerenovation` представляє опис особливостей виконання ремонту цього будинку власника, реалізований типом `CharField`.

Поле `renovationcity_id` у класі моделі `Homerenovation` представляє місто, де знаходиться цей будинок власника, ремонт якого має бути

виконаний, реалізоване типом `ForeignKey` зі зв'язком з моделлю `Renovationcity`.

Поле `homerenovationcreatetime` у класі моделі `Homerenovation` представляє час збереження у програмі цього оголошення про необхідний власнику будинку ремонт, реалізований типом `DateTimeField`.

Поле `homerenovationaddress` у класі моделі `Homerenovation` представляє адресу цього будинку власника, ремонт якого має бути виконаний, реалізовану типом `CharField`.

Поле `homerenovationstyle_id` у класі моделі `Homerenovation` представляє дизайнерський стиль, у якому потрібно виконати ремонт цього будинку, реалізований типом `ForeignKey` зі зв'язком з моделлю `Homerenovationstyle`.

Модель кімнат будинку, в яких потрібно виконати ремонт, реалізована класом `Homerenovationroom`.

Поле `homerenovation_id` у класі моделі `Homerenovationroom` представляє будинок, ремонт якого потрібно виконати для власника та в якому знаходиться ця кімната, реалізований типом `ForeignKey` зі зв'язком з моделлю `Homerenovation`.

Поле `homeroomtype` у класі моделі `Homerenovationroom` представляє тип кімнати будинку, в якій потрібно виконати ремонт, реалізований типом `CharField`.

Поле `homeroomdescription` у класі моделі `Homerenovationroom` представляє опис особливостей кімнати будинку, в якій потрібно виконати ремонт, та ремонту в цій кімнаті, реалізований типом `CharField`.

Поле `homerenovationstartdate` у класі моделі `Homerenovationroom` представляє необхідну найранішу дату початку ремонту в цій кімнаті, реалізовану типом `DateField`.

Поле `homerenovationfinishdate` у класі моделі `Homerenovationroom` представляє необхідну найпізнішу дату завершення ремонту в цій кімнаті, реалізовану типом `DateField`.

Модель фотографії кімнат будинку, в яких потрібно виконати ремонт, реалізована класом `Homerenovationroomphoto`.

Поле `homerenovationroom_id` у класі моделі `Homerenovationroomphoto` представляє кімнату будинку, в якій потрібно виконати ремонт та яка зображена на фотографії, реалізовану типом `ForeignKey` зі зв'язком з моделлю `Homerenovationroom`.

Поле `homeroomphoto` у класі моделі `Homerenovationroomphoto` представляє саму фотографію кімнати будинку, в якій потрібно виконати ремонт, реалізовану типом `ImageField`.

Поле `homeroomphotodescription` у класі моделі `Homerenovationroomphoto` представляє опис особливостей, зображених на фотографії кімнати будинку, в якій потрібно виконати ремонт, реалізований типом `CharField`.

Модель фотографії будинку, в якому власнику потрібно виконати ремонт, реалізована класом `Homerenovationphoto`.

Поле `homerenovation_id` у класі моделі `Homerenovationphoto` представляє оголошення про необхідність ремонту будинку, який зображений на фотографії, реалізоване типом `ForeignKey` зі зв'язком з моделлю `Homerenovation`.

Поле `homephoto` у класі моделі `Homerenovationphoto` представляє саму фотографію будинку, в якому потрібно виконати ремонт, реалізовану типом `ImageField`.

Поле `homephotodescription` у класі моделі `Homerenovationphoto` представляє опис особливостей, зображених на фотографії будинку, в якому потрібно виконати ремонт, реалізований типом `CharField`.

Модель робіт з ремонту, які за думкою власника необхідно виконати для його будинку, реалізована класом `Homerenovationwork`.

Поле `homerenovation_id` у класі моделі `Homerenovationwork` представляє необхідний власнику ремонт будинку, до якого відноситься ця робота, реалізований типом `ForeignKey` зі зв'язком з моделлю `Homerenovation`.

Поле `renovationwork_id` у класі моделі `Homerenovationwork` представляє роботу з ремонту, яку необхідно виконати (на думку власника) для будинку, реалізовану типом `ForeignKey` зі зв'язком з моделлю `Renovationwork`.

Поле `renovationunitnumber` у класі моделі `Homerenovationwork` представляє кількість базових одиниць, в яку оцінюється масштаб виконання цієї роботи з ремонту для всього будинку, реалізовану типом `FloatField`.

Поле `renovationcomment` у класі моделі `Homerenovationwork` представляє опис особливостей виконання цієї ремонтної роботи для будинку, реалізований типом `CharField`.

Модель заявки на виконання ремонтних робіт за будинком реалізована класом `Homerenovationapplication`.

Поле `homerenovation_id` у класі моделі `Homerenovationapplication` представляє необхідний власнику ремонт будинку, якого стосується заявка, реалізований типом `ForeignKey` зі зв'язком з моделлю `Homerenovation`.

Поле `renovatorcrew_id` у класі моделі `Homerenovationapplication` представляє бригаду з ремонту будинку, яка має виконати ремонт за цією заявкою, реалізовану типом `ForeignKey` зі зв'язком з моделлю `Renovatorcrew`.

Поле `applicationsourcetype` у класі моделі `Homerenovationapplication` представляє тип подачі заявки на виконання ремонтних робіт за будинком, реалізований типом `CharField`.

Поле `renovationstarttime` у класі моделі `Homerenovationapplication` представляє запланований або фактичний час початку ремонту за цим будинком власника, реалізований типом `DateTimeField`.

Поле `renovationendtime` у класі моделі `Homerenovationapplication` представляє запланований або фактичний час завершення ремонту за цим будинком власника, реалізований типом `DateTimeField`.

Поле `renovationproposedprice` у класі моделі `Homerenovationapplication` представляє пропонувану вартість ремонту за цим будинком власника, реалізовану типом `FloatField`.

Поле `renovationprice` у класі моделі `Homerenovationapplication` представляє фактичну вартість ремонту за цим будинком власника, реалізовану типом `FloatField`.

Поле `renovationcomment` у класі моделі `Homerenovationapplication` представляє коментар до виконання ремонту за цим будинком власника, реалізований типом `CharField`.

Модель ремонтних робіт за такими заявками реалізована класом `Homerenovationapplicationwork`.

Поле `homerenovationapplication_id` у класі моделі `Homerenovationapplicationwork` представляє заявку на виконання ремонтних робіт за будинком, якої стосується ця робота, реалізовану типом `ForeignKey` зі зв'язком з моделлю `Homerenovationapplication`.

Поле `renovatorcrewwork_id` у класі моделі `Homerenovationapplicationwork` представляє ремонтну роботу, яка визначена бригадою для виконання за цією заявкою, реалізовану типом `ForeignKey` зі зв'язком з моделлю `Renovatorcrewwork`.

Поле `renovationunitnumber` у класі моделі `Homerenovationapplicationwork` представляє кількість базових одиниць виконання цієї роботи з ремонту, яку необхідно застосувати під час ремонту всього будинку, реалізовану типом `FloatField`.

Поле `renovationcomment` у класі моделі `Homerenovationapplicationwork` представляє коментар до виконання цієї ремонтної роботи за заявкою стосовно будинку, реалізований типом `CharField`.

Модель станів, в яких знаходилась заявка на виконання ремонтних робіт за будинком, реалізована класом `Homerenovationapplicationstatus`.

Поле `homerenovationapplication_id` у класі моделі `Homerenovationapplicationstatus` представляє заявке на виконання ремонтних робіт, яка знаходилась в цьому стані, реалізовану типом `ForeignKey` зі зв'язком з моделлю `Homerenovationapplication`.

Поле `renovationstatus` у класі моделі `Homerenovationapplicationstatus` представляє стан, який під час цього збереження було встановлено заявці на виконання ремонтних робіт, реалізований типом `PositiveSmallIntegerField`.

Поле `applicationstatussettype` у класі моделі `Homerenovationapplicationstatus` представляє тип встановлення стану: власником будинку або бригадою з ремонту будинків, реалізований типом `BooleanField`.

Модель виконаних оцінювання компаній з ремонту будинків реалізована класом `Renovatorrating`.

Поле `homerenovationapplication_id` у класі моделі `Renovatorrating` представляє заявку на ремонт будинку, за якою було виконано це оцінювання, реалізовану типом `ForeignKey` зі зв'язком з моделлю `Homerenovationapplication`.

Поле `plusside` у класі моделі `Renovatorrating` представляє позитивні сторони, виявлені власником будинку стосовно компанії з ремонту будинків, бригади, яка її представляла, виконаних робіт, реалізовані типом `CharField`.

Поле `minusside` у класі моделі `Renovatorrating` представляє негативні сторони, виявлені власником будинку стосовно компанії з ремонту будинків, бригади, яка її представляла, виконаних робіт, реалізовані типом `CharField`.

Поле `ratingcreatetime` у класі моделі `Renovatorrating` представляє час оцінювання компанії з ремонту будинків власником будинку, реалізований типом `DateTimeField`.

Поле `renovoverallpoint` у класі моделі `Renovatorrating` представляє загальну оцінку компанії з ремонту будинків, виставлену власником будинку, реалізовану типом `PositiveSmallIntegerField`.

Поле `renovqualitypoint` у класі моделі `Renovatorrating` представляє оцінку якості виконаних ремонтних робіт компанії, виставлену власником будинку, реалізовану типом `PositiveSmallIntegerField`.

Поле `renovdurationpoint` у класі моделі `Renovatorrating` представляє оцінку дотриманню строків при виконанні ремонтних робіт компанією, виставлену власником будинку, реалізовану типом `PositiveSmallIntegerField`.

Поле `renovattitudepoint` у класі моделі `Renovatorrating` представляє оцінку відношенню при виконанні ремонтних робіт компанією, виставлену власником будинку, реалізовану типом `PositiveSmallIntegerField`.

Поле `renovpricerpoint` у класі моделі `Renovatorrating` представляє оцінку відповідності вартості виконаних ремонтних робіт якості, виставлену власником будинку, реалізовану типом `PositiveSmallIntegerField`.

Поле `renovphoto` у класі моделі `Renovatorrating` представляє фотографію результатів виконання ремонтних робіт, реалізовану типом `ImageField`.

Модель виконаних оцінювань власників будинків компаніями з ремонту реалізована класом `Homeownerrating`.

Поле `homerenovationapplication_id` у класі моделі `Homeownerrating` представляє заявку на ремонт будинку, за якою було виконано це оцінювання, реалізовану типом `ForeignKey` зі зв'язком з моделлю `Homerenovationapplication`.

Поле `plusside` у класі моделі `Homeownerrating` представляє позитивні сторони, виявлені компанією з ремонту будинків стосовно власника будинку, реалізовані типом `CharField`.

Поле `minusside` у класі моделі `Homeownerrating` представляє негативні сторони, виявлені компанією з ремонту будинків стосовно власника будинку, реалізовані типом `CharField`.

Поле `ratingcreatetime` у класі моделі `Homeownerrating` представляє час оцінювання власника будинку компанією з ремонту будинків власником будинку, реалізований типом `DateTimeField`.

Поле `renovoverallpoint` у класі моделі `Homeownerrating` представляє оцінку власника будинку компанією з ремонту будинків, реалізовану типом `PositiveSmallIntegerField`.

Модель закладок на необхідних власникам будинків ремонтах реалізована класом `Homerenovationbookmark`.

Поле `homerenovation_id` у класі моделі `Homeownerrating` представляє ремонт, необхідний власнику будинку, на якому створено закладку, реалізований типом `ForeignKey` зі зв'язком з моделлю `Homerenovation`.

Поле `renovatorcrewaccount_id` у класі моделі `Homeownerrating` представляє користувача, що є робітником, який представляє бригаду з ремонту будинків і який створив цю закладку, реалізованого типом `ForeignKey` зі зв'язком з моделлю `Renovatorcrewaccount`.

Поле `renovbookmarktime` у класі моделі `Homeownerrating` представляє час створення закладки на необхідному власнику будинку ремонті, реалізований типом `DateTimeField`.

Модель закладок на компаніях з ремонту будинків реалізована класом `Renovatorbookmark`.

Поле `renovator_id` у класі моделі `Homeownerrating` представляє компанію з ремонту будинків, на якій було створено закладку власником будинку, реалізовану типом `ForeignKey` зі зв'язком з моделлю `Renovator`.

Поле `homeowner_id` у класі моделі `Homeownerrating` представляє власника будинку, який створив закладку на компанії з ремонту будинку, реалізованого типом `ForeignKey` зі зв'язком з моделлю `Homeowner`.

Поле `renovbookmarktime` у класі моделі `Homeownerrating` представляє час створення закладки на компанії з ремонту будинку.

3.3 Висновки за розділом 3

У цьому розділі програма найму робітників для виконання ремонту будинку реалізована. Описано створені класи, функції, можливості роботи з цими засобами.

4 ЕКСПЛУАТАЦІЯ ТА ТЕСТУВАННЯ ПРОГРАМИ

4.1 Призначення програми найму робітників для виконання ремонту будинку

Програма призначена для пошуку робітників для виконання ремонту будинку з об'єднанням таких робітників у бригад, компанії з ремонту будинків, пошуку оголошень про необхідний ремонт будинку, забезпечення заявок для реалізації ремонту будинків, управління цими заявками.

4.2 Умови виконання програми найму робітників для виконання ремонту будинку

Сервер на технічному рівні повинен щонайменше мати 2 Гб оперативної пам'яті, два ядра процесора, 100 Гб для організації сховища, що має надати можливість виконання сумарно 25000 запитів на місяць до вебзастосунку, але в подальшому кількість запитів та інші варіанти навантажень мають бути оцінені додатково, що може призвести до розширення мінімальних вимог, окрім того відсоток вільного простору для організації сховища та темпи його змінювання мають оцінюватися та призводити до прийняття рішень щодо розширення доступного простору за необхідності.

Експлуатаційні вимоги до вебзастосунку найму робітників для виконання ремонту будинку включають вимоги до спеціаліста-адміністратора, який має забезпечити хостинг, розгортання вебзастосунку на сервері з застосуванням Django, підтримку роботи бази даних на основі MySQL.

Відповідно експлуатація такого вебзастосунку найму робітників для виконання ремонту будинку має бути забезпечена можливістю доступу до технологій, на основі яких реалізовано вебзастосунок та які відповідно були обрані в роботі як підсумок.

4.3 Експлуатація програми найму робітників для виконання ремонту будинку

Експлуатація програми найму робітників для виконання ремонту будинку можлива користувачем ще до авторизації. Для цього йому доступно два розділи меню для пошуку серед компаній з ремонту будинків та серед оголошень про необхідність ремонту будинків, створених їх власниками.

Вже з результатів пошуку можна переходити на сторінки компаній, далі бригад (рис. 4.1).

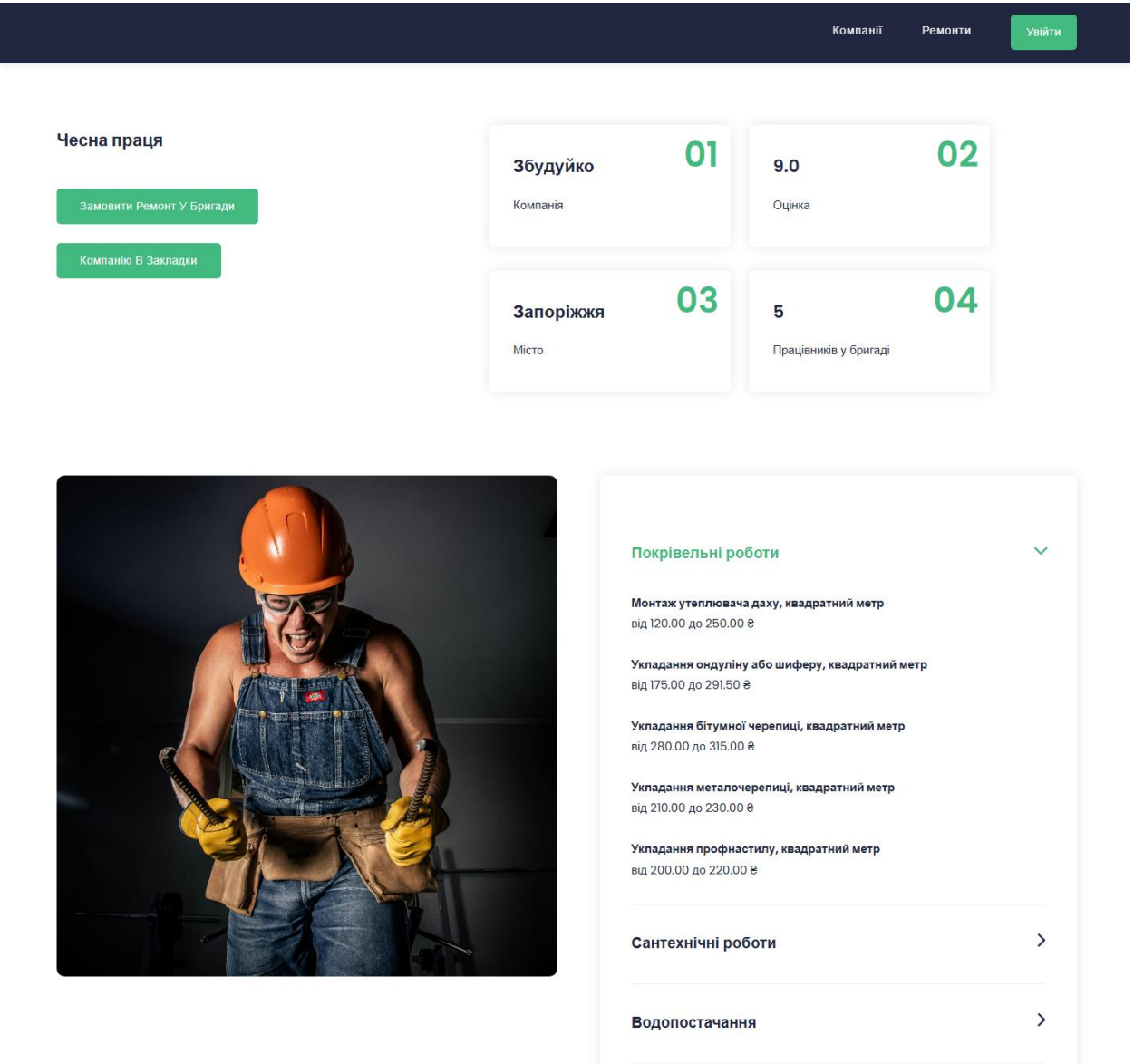


Рисунок 4.1 – Сторінка бригади з ремонту будинку

Сторінка бригади з ремонту будинку містить перелік груп робіт, самі роботи та проміжок вартості таких робіт (конкретна градація за складністю приводиться на сторінці компанії). Також надається фотографія бригади з ремонту будинку, можливість додати в закладки компанію з ремонту будинку, в якій діє ця бригада, посилання на створення заявки для робіт у цієї бригади, інформація про компанію та кількість робітників у бригаді.

Власники будинків мають у меню окрім розділів для пошуку також розділ власних ремонтів, звідки можна створити нове оголошення, переглянути власні оголошення. Також доступний розділ керування заявками. З оголошення про ремонт будинку можна перейти до сторінки з заявками на ремонт цього будинку (рис. 4.2), звідки власник будинку може керувати ними в залежності від стану.

Компанії Ремонти Мої Ремонти Заявки Мої Дані Закладки Оцінювання Вийти			
ЗАПОРІЖЖЯ, ВУЛ. НИЖНЯ, 10			
Заявки, Подані За Вашим Будинком			
Номер та виконавець	Строки та вартість	Стан	Керування
#19 Бригада Детально	20.06.2025 9:00 - 24.06.2025 17:00 23000€	Створено бригадою Історія станів	Підтвердити Скасувати
#18 Бригада Детально	19.06.2025 9:00 - 23.06.2025 18:00 22000€	Створено бригадою Історія станів	Підтвердити Скасувати
#17 Бригада Детально	20.06.2025 9:00 - 23.06.2025 15:00 14000€	Створено бригадою Історія станів	Підтвердити Скасувати
#16 Бригада Детально	20.06.2025 9:00 - 21.06.2025 14:00 12000€	Створено бригадою Історія станів	Підтвердити Скасувати

Рисунок 4.2 – Перегляд та керування заявками, поданими на ремонт будинку власника

Коли заявки на ремонт тільки створені бригадами, то доступні тільки засоби підтвердження та скасування. Далі ж можна визначити початок, завершення робіт. З цієї сторінки можна також перейти на сторінку бригади з ремонту будинку, яка створила заявку, сторінку детальних параметрів заявки, сторінку зміни станів заявки.

Власник будинку може також з окремого розділу керувати власними даними, переглядати вже виконані цим оцінювання компанії з ремонту будинків, переглядати компанії з ремонту будинків, на яких він створив закладки.

Робітник з ремонту будинків окрім засобів пошуку має також засоби керування заявками, які були бригаді, в якій він працює, направлені, або нею створені. Окрім того йому доступні засоби керування власними даними, даними компанії з ремонту будинків, в якій він працює, а звідти можна перейти і до даних бригади. Стосовно бригади, то такий робітник вже може управляти доступним для неї переліком робіт з ремонту будинків, обираючи їх з переліку ремонтних робіт компанії. Для компанії з ремонту будинку можна відповідно додавати, оновлювати, вилучати окремі роботи, визначати їх вартість. Також він може переглядати закладки на оголошеннях про ремонт будинків, виконані ним оцінювання власників будинків.

Оцінювання власників будинків відбуваються тільки тоді, коли було раніше створено заявку стосовно будинку такого власника цією бригадою (або навпаки власником будинку до бригади). Під час оцінювання необхідно задати не тільки загальну оцінку, але і опис відгуку, розділений на поля для негативних сторін та позитивних сторін (рис. 4.3).

В залежності від поточної ролі також наявна кнопка, яка дозволяє або вийти з облікового запису, якщо авторизація була виконана, або перейти до неї. При переході до авторизації можна також виконати реєстрацію облікового запису.

Компанії Ремонти Заявки Мої Дані Моя Компанія Закладки Оцінювання Вийти

ЗАЯВКА #38

Залиште Ваш Відгук Про Власника Будинку

Оцінка 10

Плюси

Мінуси

Зберегти

Рисунок 4.3 – Оцінювання власника будинку, для якого виконувався ремонт, робітником

4.4 Тестування програми найму робітників для виконання ремонту будинку

Тестування програми найму робітників для виконання ремонту будинку було виконано у повній мірі за переліком функцій, які були попередньо розподілені за ролями користувачів. Результати тестування були внесені в чек-лист (табл. 4.1). Для випадків, коли функція виконувалась під час тестування успішно для цієї ролі користувача, виставлялась позначка «+». У протилежному випадку проставлялась позначка «-». Якщо така функція не мала бути доступна для такої ролі користувача, то позначка не проставлялась. Оскільки результатом авторизації є вже відповідно авторизований обліковий запис у конкретній ролі, то авторизація для користувача до авторизації не проставлялась. Усі функції призвели до успіху під час тестування.

Таблиця 4.1 – Результати тестування програми найму робітників для виконання ремонту будинку

Функція	Користувач до авторизації	Власник будинку	Робітник з ремонту будинків
1	2	3	4
Перегляд переліку ремонтних робіт, які здатна виконувати компанія	+	+	+
Перегляд даних власника будинку	+	+	+
Перегляд даних компанії з ремонту будинків	+	+	+
Авторизація		+	+
Перегляд переліку бригад з ремонту будинків компанії	+	+	+
Перегляд даних бригади з ремонту будинків	+	+	+
Перегляд доступних для роботи бригад у компанії з ремонту будинків на заданий час	+	+	+
Перегляд переліку необхідних ремонтів будинків власника	+	+	+
Перегляд оголошення про необхідність ремонту будинку	+	+	+
Перегляд оцінювань власника будинку	+	+	+
Перегляд оцінювань компанії з ремонту будинків	+	+	+
Пошук оголошень про необхідність ремонту будинку	+	+	+

Продовження таблиці 4.1

1	2	3	4
Пошук компанії з ремонту будинку	+	+	+
Реєстрація облікового запису як власника будинку		+	
Реєстрація облікового запису як робітника з ремонту будинків			+
Визначення даних власника будинку		+	
Створення оголошення про необхідний ремонт будинку		+	
Оновлення оголошення про необхідний ремонт будинку		+	
Вилучення оголошення про необхідний ремонт будинку		+	
Додавання кімнати до даних ремонту будинку		+	
Оновлення кімнати з даних ремонту будинку		+	
Вилучення кімнати з даних ремонту будинку		+	
Додавання роботи до плану ремонту будинку		+	
Вилучення роботи з плану ремонту будинку		+	
Оновлення роботи з плану ремонту будинку		+	
Подання заявки на виконання ремонту будинку		+	+

Продовження таблиці 4.1

1	2	3	4
Управління станом заявки на виконання ремонту будинку		+	+
Перегляд заявки на виконання ремонту будинку		+	+
Перегляд заявок на виконання ремонту за будинком		+	
Перегляд власних поданих заявок на виконання ремонту будинку		+	+
Перегляд станів, які визначались за заявкою на виконання ремонту будинку		+	+
Оцінювання компанії з ремонту будинків		+	
Перегляд оцінених компаній з ремонту будинків власником будинку		+	
Створення закладки за компанією з ремонту будинку		+	
Вилучення закладки за компанією з ремонту будинку		+	
Перегляд переліку компаній з ремонту будинку в закладках		+	
Визначення власної компанії з ремонту будинків			+
Оновлення даних компанії з ремонту будинків			+
Додавання ремонтної роботи для компанії з ремонту будинків			+
Оновлення ремонтної роботи для компанії з ремонту будинків			+

Продовження таблиці 4.1

1	2	3	4
Вилучення ремонтної роботи для компанії з ремонту будинків			+
Додавання бригади з ремонту будинків компанії			+
Оновлення даних бригади з ремонту будинків компанії			+
Вилучення бригади з ремонту будинків компанії			+
Оновлення даних робітника з ремонту будинків			+
Перегляд даних робітника з ремонту будинків			+
Перегляд складу робітників бригади з ремонту будинків			+
Додавання часового лоту для роботи бригади з ремонту будинків			+
Оновлення часового лоту для роботи бригади з ремонту будинків			+
Вилучення часового лоту для роботи бригади з ремонту будинків			+
Додавання ремонтної роботи до можливостей бригади			+
Вилучення ремонтної роботи з можливостей бригади			+
Перегляд заявок на виконання ремонту будинку			+

Кінець таблиці 4.1

1	2	3	4
Додавання роботи до плану ремонту будинку за заявкою			+
Вилучення роботи з плану ремонту будинку за заявкою			+
Оновлення роботи з плану ремонту будинку за заявкою			+
Оцінювання власника будинку			+
Перегляд оцінених власників будинків бригадою з ремонту будинків			+
Створення закладки за ремонтом будинку			+
Вилучення закладки за ремонтом будинку			+
Перегляд переліку ремонтів будинків у закладках			+

4.5 Висновки за розділом 4

У розділі 4, забезпечуючи тестування програми найму робітників для виконання ремонту будинку, було описано принципи експлуатації програми, перевірено успішну реалізацію всіх функцій програми, що свідчить про виконання завдання розробки програми найму робітників для виконання ремонту будинку та її тестування.

ВИСНОВКИ

У роботі було розроблено програму найму робітників, направлено на виконання ремонту будинку, для забезпечення принципу двосторонньої пропозиції. Виконання роботи базувалось на аналізі сучасного стану розробки програм, які можуть застосовуватися для найму робітників для ремонту будинків, виділяючи від них основні позитивні сторони та беручи до разунки можливості подолання негативних сторін.

Програму найму робітників для ремонту будинків розроблено мовою Python з застосуванням фреймворк Django та системи MySQL для роботи з базою даних.

Розроблена програма призначена для пошуку робітників для виконання ремонту будинку з об'єднанням таких робітників у бригади, компанії з ремонту будинків, пошуку оголошень про необхідний ремонт будинку, забезпечення заявок для реалізації ремонту будинків, управління цими заявками.

Реалізація принципу двосторонньої пропозиції дозволяє кожній зі сторін, власникам будинків та робітникам, бригадам і компаніям з ремонту будинків виставляти власні позиції, пропонуючи задоволення таких пропозицій в основі реалізації подальшої взаємодії. Це було забезпечено в розробленій програмі, що дозволяє підтримувати реалізацію принципу двосторонньої пропозиції під час найму робітників для виконання ремонту будинку.

Усі прийняті рішення, основні досягнуті результати під час виконання роботи загалом та розробки програми найму робітників для ремонту будинків описані. Тестування програми, що створено в результаті, відбулось успішно.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Допоможемо знайти бригаду майстрів, будівельників на ремонт квартири або будівництво будинку в Києві. Robotniki.ua [Електрон. ресурс]. – Режим доступу : <https://www.rabotniki.ua/uk/>.
2. Oochcom – Сервіс з пошуку будівельних та ремонтних робіт [Електрон. ресурс]. – Режим доступу : <https://ooch.com.ua/>.
3. Ремонт квартир та будинків під ключ в Україні на OLX.ua [Електрон. ресурс]. – Режим доступу : <https://www.olx.ua/uk/uslugi/stroitelstvo-otdelka-remont/remont-pod-klyuch/?currency=UAH>.
4. Documentation Pyramid [Electronic resource]. – Access mode : <https://trypyramid.com/documentation.html>.
5. Django documentation [Electronic resource]. – Access mode : <https://docs.djangoproject.com/en/5.2/>.
6. Free Template 574 Mexant [Electronic resource]. – Access mode : <https://templatemo.com/tm-574-mexant>.

ДОДАТОК А
Технічне завдання

Вступ

Ремонт будинку – це завжди складний процес, який потребує багато уваги, часу, фінансів, а також ще й нерідко відзначається стресом. Від виконавців ремонту залежить вплив кожного з цих факторів. Тому пошук робітників для виконання ремонту будинку є вкрай важливою основою для даного процесу. Від результатів його виконання залежать всі наступні етапи.

Виконувати пошук на сайтах окремих компаній з ремонту будинків фактично неможливо для якісного вирішення цієї проблеми. Це пояснюється тим, що в такому випадку неможливо орієнтуватися на відгуки замовників, тому що на вебсайті кожної окремої компанії рідко приводяться правдиві відгуки, але окрім того точно можна сказати, що такі відгуки є правдивими точно не на кожному сайті.

Тому потрібен засіб, в якому б різні працівники, можливо бригади з ремонту будинків та компанії могли пропонувати свої послуги, а власники будинків шукати тих, хто надасть їм такі послуги. Важливо детально визначити необхідні можливості для такої програми, щоб повноцінно охоплювати можливості найму робітників для виконання ремонту будинку через програму.

A.1 Підстави для розробки

Основна підстава для виконання роботи сформульована у відповідності з Наказом № 209 від 28 квітня 2025 року, виданим у Національному університеті «Запорізька політехніка», де для виконання даної дипломної кваліфікаційної роботи бакалавра визначено «Програмне забезпечення для найму робітників для виконання ремонту будинку» як тему роботи.

A.2 Призначення розробки

Програма призначена для пошуку робітників для виконання ремонту будинку з об'єднанням таких робітників у бригад, компанії з ремонту будинків, пошуку оголошень про необхідний ремонт будинку, забезпечення заявок для реалізації ремонту будинків, управління цими заявками.

A.3 Основні вимоги до програми

A.3.1 Вимоги до функціональних характеристик

Вимоги до програми найму робітників для виконання ремонту будинку за набором функцій включають:

- визначення даних власника будинку;
- реєстрація облікового запису як власника будинку або робітника з ремонту будинків;
- авторизація;
- визначення власної компанії з ремонту будинків;
- оновлення даних компанії з ремонту будинків;
- додавання ремонтної роботи для компанії з ремонту будинків;
- оновлення ремонтної роботи для компанії з ремонту будинків;
- вилучення ремонтної роботи для компанії з ремонту будинків;
- перегляд переліку ремонтних робіт, які здатна виконувати компанія;
- перегляд даних власника будинку;
- перегляд даних компанії з ремонту будинків;
- додавання бригади з ремонту будинків компанії;
- оновлення даних бригади з ремонту будинків компанії;
- вилучення бригади з ремонту будинків компанії;
- перегляд переліку бригад з ремонту будинків компанії;
- оновлення даних робітника з ремонту будинків;
- перегляд даних робітника з ремонту будинків;

- перегляд складу робітників бригади з ремонту будинків;
- перегляд даних бригади з ремонту будинків;
- додавання часового лоту для роботи бригади з ремонту будинків;
- оновлення часового лоту для роботи бригади з ремонту будинків;
- вилучення часового лоту для роботи бригади з ремонту будинків;
- перегляд доступних для роботи бригад у компанії з ремонту будинків

на заданий час;

- додавання ремонтної роботи до можливостей бригади;
- вилучення ремонтної роботи з можливостей бригади;
- створення оголошення про необхідний ремонт будинку власником;
- оновлення оголошення про необхідний ремонт будинку власником;
- вилучення оголошення про необхідний ремонт будинку власником;
- перегляд переліку необхідних ремонтів будинків власника;
- додавання кімнати до даних ремонту будинку;
- оновлення кімнати з даних ремонту будинку;
- вилучення кімнати з даних ремонту будинку;
- додавання роботи до плану ремонту будинку;
- вилучення роботи з плану ремонту будинку;
- оновлення роботи з плану ремонту будинку;
- подання заявки на виконання ремонту будинку бригадою або

компанією;

- управління станом заявки на виконання ремонту будинку;
- перегляд заявки на виконання ремонту будинку;
- перегляд заявок на виконання ремонту будинку;
- перегляд власних поданих заявок на виконання ремонту будинку;
- додавання роботи до плану ремонту будинку за заявкою;
- вилучення роботи з плану ремонту будинку за заявкою;
- оновлення роботи з плану ремонту будинку за заявкою;
- перегляд станів, які визначались за заявкою на виконання ремонту

будинку;

- оцінювання компанії з ремонту будинків;
- оцінювання власника будинку;
- перегляд оцінених власників будинків бригадою з ремонту будинків;
- перегляд оцінених компаній з ремонту будинків власниками будинків;
- перегляд оцінювань власника будинку;
- перегляд оцінювань компанії з ремонту будинків;
- створення закладки за компанією з ремонту будинку;
- вилучення закладки за компанією з ремонту будинку;
- створення закладки за ремонтом будинку;
- вилучення закладки за ремонтом будинку;
- перегляд переліку ремонтів будинків у закладках;
- перегляд переліку компаній з ремонту будинку в закладках;
- перегляд оголошення про необхідність ремонту будинку;
- пошук оголошень про необхідність ремонту будинку;
- пошук компанії з ремонту будинку.

А.3.2 Умови експлуатації

Експлуатаційні вимоги до вебзастосунку найму робітників для виконання ремонту будинку включають вимоги до спеціаліста-адміністратора, який має забезпечити хостинг, розгортання вебзастосунку на сервері з застосуванням обраного для реалізації вебфреймворку, підтримку роботи бази даних на основі обраної в роботі системи керування.

Відповідно експлуатація такого вебзастосунку найму робітників для виконання ремонту будинку має бути забезпечена можливістю доступу до технологій, на основі яких реалізовано вебзастосунок та які відповідно були обрані в роботі як підсумок.

А.3.3 Вимоги до складу та параметрів технічних засобів

Сервер на технічному рівні повинен щонайменше мати 2 Гб оперативної пам'яті, два ядра процесора, 100 Гб для організації сховища, що має надати можливість виконання сумарно 25000 запитів на місяць до вебзастосунку, але в подальшому кількість запитів та інші варіанти навантажень мають бути оцінені додатково, що може призвести до розширення мінімальних вимог, окрім того відсоток вільного простору для організації сховища та темпи його змінювання мають оцінюватися та призводити до прийняття рішень щодо розширення доступного простору за необхідності.

А.4 Порядок контролю та приймання

Сумарно 8 тижнів виділено на виконання дипломної кваліфікаційної роботи бакалавра, розподіл між якими має виконуватися за календарним планом, визначеним у завданні на роботі, з забезпеченням результатів, які мають бути досягнуті та які зазначені в календарному плані.

У підсумку робота оцінюється керівником зі створенням відгуку, рецензентом зі створенням рецензії та екзаменаційною комісією за наперед затвердженим графіком.

ДОДАТОК Б
Текст програми

```

from .models import Homeowner
from .models import Renovationcity
from .models import Renovator
from .models import Renovatordocument
from .models import Renovatorphoto
from .models import Renovatorwork
from .models import Renovatorworkprice
from .models import Renovatorcrewaccount
from .models import Renovatorcrew
from .models import Renovatorcrewtime
from .models import Renovatorcrewwork
from .models import Renovationworkgroup
from .models import Renovationwork
from .models import Homerenovationstyle
from .models import Homerenovation
from .models import Homerenovationroom
from .models import Homerenovationroomphoto
from .models import Homerenovationphoto
from .models import Homerenovationwork
from .models import Homerenovationapplication
from .models import Homerenovationapplicationwork
from .models import Homerenovationapplicationstatus
from .models import Renovatorrating
from .models import Homeownerrating
from .models import Homerenovationbookmark
from .models import Renovatorbookmark
from django.shortcuts import render
from django.shortcuts import redirect
from django.core.paginator import Paginator
from .forms import HomerenovationroomForm


def homerenovationroom_add_view(request, homerenovation_id):
    if request.user.is_authenticated:
        homeowner_ready =
Homeowner.objects.select_related("homeownerindjango_id").get(homeowner
indjango_id = request.user)
        if not homeowner_ready:
            return render(request, "renovatorapp/error.html",
{"error": "Для роботи в цьому режимі потрібно бути власником будинку,
що авторизувався."})
        else:

```

```

        return render(request, "renovatorapp/error.html",
{"error": "Для додавання кімнат до плану будинку під час ремонту
потрібно бути власником будинку, що авторизувався."})
        if homeowner_ready:
            try:
                home_renovation =
Homer renovation.objects.select_related("homeowner_id").get(id =
homer renovation_id)
            except Homer renovation.DoesNotExist:
                home_renovation = None
            if not home_renovation:
                return render(request, "renovatorapp/error.html",
{"error": "Такий план ремонту будинку не було знайдено."})
            elif home_renovation.homeowner_id != homeowner_ready:
                return render(request, "renovatorapp/error.html",
{"error": "Для додавання кімнат до плану будинку під час ремонту
потрібно бути його власником."})
            if request.method == "POST":
                home_renovation_form =
Homer renovationroomForm(request.POST)
                if home_renovation_form.is_valid():
                    home_renovation_from_form =
home_renovation_form.save(commit=False)
                    home_renovation_from_form.homer renovation_id =
home_renovation
                    home_renovation_from_form.save()
                    return redirect(request.path_info)
            else:
                home_renovation_form = Homer renovationroomForm ()
                return render(request, "renovatorapp/newroom.html", {"form":
home_renovation_form })

def homer renovationroom_change_view(request,
homer renovationroom_id):
    if request.user.is_authenticated:
        homeowner_ready =
Homeowner.objects.select_related("homeownerindjango_id").get(homeowner
indjango_id = request.user)
        if not homeowner_ready:
            return render(request, "renovatorapp/error.html",
{"error": "Для роботи з оцінюванням в цьому режимі потрібно бути
власником будинку, що авторизувався."})

```

```

        else:
            return render(request, "renovatorapp/error.html",
{"error": "Для роботи в цьому режимі потрібно бути власником будинку,
що авторизувався."})
            if homeowner_ready:
                try:
                    home_renovation_room =
Homer renovationroom.objects.select_related("homer renovation_id",
"homer renovation_id__homeowner_id").get(id = homer renovationroom_id)
                except Homer renovationroom.DoesNotExist:
                    home_renovation_room = None
                if not home_renovation_room:
                    return render(request, "renovatorapp/error.html",
{"error": "Таку кімнату в плані будинку не було знайдено."})
                if home_renovation_room.homer renovation_id.homeowner_id !=
homeowner_ready:
                    return render(request, "renovatorapp/error.html",
{"error": "Для оновлення даних кімнати з плану будинку під час ремонту
потрібно бути його власником."})
                home_renovation_form = Homer renovationroomForm(request.POST
or None, instance = home_renovation_room)
                if request.method == "POST" and
home_renovation_form.is_valid():
                    home_renovation_form.save()
                    return redirect('viewhomer renovation', id =
home_renovation_room.homer renovation_id.id)
                else:
                    home_renovation_form = None
                    return render(request, "renovatorapp/changeroom.html",
{"form": home_renovation_form })

```

```

from .forms import Homer renovationworkForm

```

```

def homer renovationwork_add_view(request, homer renovation_id):
    if request.user.is_authenticated:
        homeowner_ready =
Homeowner.objects.select_related("homeowner indjango_id").get(homeowner
indjango_id = request.user)
        if not homeowner_ready:

```

```

        return render(request, "renovatorapp/error.html",
{"error": "Для роботи в цьому режимі потрібно бути власником будинку,
що авторизувався."})
    else:
        return render(request, "renovatorapp/error.html",
{"error": "Для додавання робіт до плану ремонту будинку потрібно бути
власником будинку, що авторизувався."})
        if homeowner_ready:
            try:
                home_renovation =
Homer renovation.objects.select_related("homeowner_id").get(id =
homer renovation_id)
            except Homer renovation.DoesNotExist:
                home_renovation = None
            if not home_renovation:
                return render(request, "renovatorapp/error.html",
{"error": "Такий план ремонту будинку не було знайдено."})
            elif home_renovation.homeowner_id != homeowner_ready:
                return render(request, "renovatorapp/error.html",
{"error": "Для додавання робіт до плану ремонту будинку потрібно бути
його власником."})
            if request.method == "POST":
                home_renovation_work_form =
Homer renovationworkForm(request.POST)
                if home_renovation_work_form.is_valid():
                    home_renovation_work_from_form =
home_renovation_work_form.save(commit=False)
                    home_renovation_work_from_form.homer renovation_id =
home_renovation
                    home_renovation_work_from_form.save()
                    return redirect(request.path_info)
            else:
                home_renovation_work_form = Homer renovationworkForm()
                return render(request,
"renovatorapp/newrenovaplanwork.html", {"form":
home_renovation_work_form })

def homer renovationwork_change_view(request,
homer renovationwork_id):
    if request.user.is_authenticated:

```

```

        homeowner_ready =
Homeowner.objects.select_related("homeownerindjango_id").get(homeowner
indjango_id = request.user)
        if not homeowner_ready:
            return render(request, "renovatorapp/error.html",
{"error": "Для роботи в цьому режимі потрібно бути власником будинку,
що авторизувався."})
        else:
            return render(request, "renovatorapp/error.html",
{"error": "Для оновлення даних роботи з плану будинку під час ремонту
потрібно бути власником будинку, що авторизувався."})
        if homeowner_ready:
            try:
                home_renovation_work =
Homerenovationwork.objects.select_related("homerenovation_id",
"homerenovation_id__homeowner_id").get(id = homerenovationwork_id)
            except Homerenovationwork.DoesNotExist:
                home_renovation_work = None
            if not home_renovation_work:
                return render(request, "renovatorapp/error.html",
{"error": "Таку роботу в плані будинку не було знайдено."})
            if home_renovation_work.homerenovation_id.homeowner_id !=
homeowner_ready:
                return render(request, "renovatorapp/error.html",
{"error": "Для оновлення даних роботи з плану будинку під час ремонту
потрібно бути його власником."})
            home_renovation_work_form =
HomerenovationworkForm(request.POST or None, instance =
home_renovation_work)
            if request.method == "POST" and
home_renovation_work_form.is_valid():
                home_renovation_work_form.save()
                return redirect('viewhomerenovation', id =
home_renovation_work_form.homerenovation_id.id)
            else:
                home_renovation_work_form = None
                return render(request, "renovatorapp/changeplanwork.html",
{"form": home_renovation_work_form })

```

```

from .forms import HomerenovationapplicationworkForm

```

```

def homerenovationapplicationwork_add_view(request,
homerenovationapplication_id):
    if request.user.is_authenticated:
        homerenovator_ready =
Renovatorcrewaccount.objects.select_related("renovatorcrewaccountindja
ngo_id", "renovatorcrew_id").get(renovatorcrewaccountindjango_id =
request.user)
        if not homerenovator_ready:
            return render(request, "renovatorapp/error.html",
{"error": "Для роботи в цьому режимі потрібно бути робітником, що
авторизувався."})
        else:
            return render(request, "renovatorapp/error.html",
{"error": "Для додавання робіт до заявки на ремонт будинку потрібно
бути робітником, що авторизувався."})
            if homerenovator_ready:
                try:
                    home_renovation_application =
Homerenovationapplication.objects.select_related("renovatorcrew_id").g
et(id = homerenovationapplication_id)
                except Homerenovationapplication.DoesNotExist:
                    home_renovation_application = None
                if not home_renovation_application:
                    return render(request, "renovatorapp/error.html",
{"error": "Таку заявку на ремонт будинку не було знайдено."})
                elif home_renovation_application.renovatorcrew_id !=
homerenovator_ready.renovatorcrew_id:
                    return render(request, "renovatorapp/error.html",
{"error": "Для додавання робіт до заявки на ремонт будинку потрібно
бути представником тієї бригади, якої стосується ця заявка."})
                if request.method == "POST":
                    home_renovation_application_work_form =
HomerenovationapplicationworkForm(request.POST)
                    if home_renovation_application_work_form.is_valid():
                        home_renovation_application_work_from_form =
home_renovation_application_work_form.save(commit=False)
                        home_renovation_application_work_from_form.
homerenovationapplication_id = home_renovation_application
                        home_renovation_application_work_from_form.save()
                        return redirect(request.path_info)
                    else:
                        home_renovation_application_work_form =
HomerenovationapplicationworkForm()

```

```

        return render(request, "renovatorapp/newapplwork.html",
{"form": home_renovation_application_work_form })

    def homer renovationapplicationwork_change_view(request,
homer renovationapplicationwork_id):
        if request.user.is_authenticated:
            homer renovator_ready =
Renovatorcrewaccount.objects.select_related("renovatorcrewaccountindja
ngo_id", "renovatorcrew_id").get(renovatorcrewaccountindjango_id =
request.user)
            if not homer renovator_ready:
                return render(request, "renovatorapp/error.html",
{"error": "Для роботи в цьому режимі потрібно бути робітником, що
авторизувався."})
            else:
                return render(request, "renovatorapp/error.html",
{"error": "Для оновлення даних роботи з плану ремонту за заявкою
потрібно бути робітником, що авторизувався."})
            if homer renovator_ready:
                try:
                    home_renovation_application_work =
Homer renovationapplicationwork.objects.select_related("homer renovationap
plication_id", "renovatorcrewwork_id",
"homer renovationapplication_id__homer renovation_id",
"homer renovationapplication_id__homer renovation_id__homeowner_id").get(i
d = homer renovationapplicationwork_id)
                except Homer renovationapplicationwork.DoesNotExist:
                    home_renovation_application_work = None
                if not home_renovation_application_work:
                    return render(request, "renovatorapp/error.html",
{"error": "Таку роботу в плані заявки на ремонт за будинком не було
знайдено."})
                if
home_renovation_application_work.home_renovation_work.homer renovation_i
d.homeowner_id != homeowner_ready:
                    return render(request, "renovatorapp/error.html",
{"error": "Для оновлення даних роботи з плану заявки ремонту будинку
потрібно представляти бригаду, якої стосується ця заявка."})
                home_renovation_application_work_form =
Homer renovationapplicationworkForm(request.POST or None, instance =
home_renovation_application_work)

```

```

        if request.method == "POST" and
home_renovation_application_work_form.is_valid():
            home_renovation_application_work_form.save()
            return redirect('viewhomerenovationapplication', id =
home_renovation_work_form.homerenovationapplication_id.id)
        else:
            home_renovation_application_work_form = None
            return render(request, "renovatorapp/changeapplwork.html",
{"form": home_renovation_application_work_form })

```

```

from .forms import RenovatorcrewtimeForm

```

```

def renovatorcrewtime_add_view(request):
    if request.user.is_authenticated:
        homerenovator_ready =
Renovatorcrewaccount.objects.select_related("renovatorcrewaccountindja
ngo_id", "renovatorcrew_id").get(renovatorcrewaccountindjango_id =
request.user)
        if not homerenovator_ready:
            return render(request, "renovatorapp/error.html",
{"error": "Для роботи в цьому режимі потрібно бути робітником, що
авторизувався."})
        else:
            return render(request, "renovatorapp/error.html",
{"error": "Для додавання часового лоту потрібно бути робітником, що
авторизувався."})
        if request.method == "POST":
            home_renovator_crew_form =
RenovatorcrewtimeForm(request.POST)
            if home_renovator_crew_form.is_valid():
                home_renovator_crew_from_form =
home_renovator_crew_form.save(commit=False)
                home_renovator_crew_from_form.renovatorcrew_id =
homerenovator_ready.renovatorcrew_id
                renovator_crew_times =
Renovatorcrewtime.objects.filter(renovatorcrew_id =
homerenovator_ready.renovatorcrew_id, renovationstarttime__lte =
home_renovator_crew_from_form.renovationstarttime,
renovationendtime__gt =
home_renovator_crew_from_form.renovationstarttime)
                if not renovator_crew_times:

```

```

        renovator_crew_times =
Renovatorcrewtime.objects.filter(renovatorcrew_id =
homerenovator_ready.renovatorcrew_id, renovationstarttime__gte =
home_renovator_crew_from_form.renovationstarttime,
renovationendtime__gt =
home_renovator_crew_from_form.renovationstarttime,
renovationstarttime__lt =
home_renovator_crew_from_form.renovationendtime)
        if not renovator_crew_times:
            home_renovator_crew_from_form.save()
            return redirect(request.path_info)
        else:
            return render(request,
"renovatorapp/error.html", {"error": "Новий часовий лот для цієї
бригади охоплює існуючий. Таке збереження не може бути виконано.
Змініть спочатку існуючий або цей."})
        else:
            return render(request,
"renovatorapp/error.html", {"error": "Новий часовий лот для цієї
бригади охоплює існуючий. Таке збереження не може бути виконано.
Змініть спочатку існуючий або цей."})
        else:
            home_renovator_crew_form = RenovatorcrewtimeForm()
            return render(request, "renovatorapp/newcrewtime.html",
{"form": home_renovator_crew_form })

def renovatorcrewtime_change_view(request,
renovatorcrewtime_id):
    if request.user.is_authenticated:
        homerenovator_ready =
Renovatorcrewaccount.objects.select_related("renovatorcrewaccountindja
ngo_id", "renovatorcrew_id").get(renovatorcrewaccountindjango_id =
request.user)
        if not homerenovator_ready:
            return render(request, "renovatorapp/error.html",
{"error": "Для роботи в цьому режимі потрібно бути робітником, що
авторизувався."})
        else:
            return render(request, "renovatorapp/error.html",
{"error": "Для оновлення часового лоту потрібно бути робітником, що
авторизувався."})
        if homerenovator_ready:

```

```

        try:
            renovator_crew_time =
Renovatorcrewtime.objects.select_related("renovatorcrew_id").get(id =
renovatorcrewtime_id)
        except Renovatorcrewtime.DoesNotExist:
            renovator_crew_time = None
            if not renovator_crew_time:
                return render(request, "renovatorapp/error.html",
{"error": "Такий часовий лот бригади з ремонту будинків не було
знайдено."})
            if renovator_crew_time.renovatorcrew_id !=
homerenovator_ready.renovatorcrew_id:
                return render(request, "renovatorapp/error.html",
{"error": "Для оновлення даних часового лоту потрібно представляти
бригаду, якої стосується цей лот."})
            renovator_crew_time_form =
RenovatorcrewtimeForm(request.POST or None, instance =
renovator_crew_time)
            if request.method == "POST" and
renovator_crew_time_form.is_valid():
                renovator_crew_times =
Renovatorcrewtime.objects.filter(renovatorcrew_id =
homerenovator_ready.renovatorcrew_id, renovationstarttime__lte =
renovator_crew_time_form.renovationstarttime, renovationendtime__gt =
renovator_crew_time_form.renovationstarttime)
                if not renovator_crew_times:
                    renovator_crew_times =
Renovatorcrewtime.objects.filter(renovatorcrew_id =
homerenovator_ready.renovatorcrew_id, renovationstarttime__gte =
renovator_crew_time_form.renovationstarttime, renovationendtime__gt =
renovator_crew_time_form.renovationstarttime, renovationstarttime__lt
= renovator_crew_time_form.renovationendtime)
                if not renovator_crew_times:
                    renovator_crew_time_form.save()
                    return redirect('viewrenovatorcrewtimes', id
= homerenovator_ready.renovatorcrew_id.id)
            else:
                renovator_crew_time_form = None
                return render(request, "renovatorapp/changecrewtime.html",
{"form": renovator_crew_time_form })

```

```

from .forms import RenovatorratingForm

def renovatorrating_add_view(request,
homerenovationapplication_id):
    if request.user.is_authenticated:
        homeowner_ready =
Homeowner.objects.select_related("homeownerindjango_id").get(homeowner
indjango_id = request.user)
        if not homerenovator_ready:
            return render(request, "renovatorapp/error.html",
{"error": "Для роботи в цьому режимі потрібно бути робітником, що
авторизувався."})
        else:
            return render(request, "renovatorapp/error.html",
{"error": "Для оцінювання компанії з ремонту будинків потрібно бути
власником будинку, що авторизувався."})
            if homeowner_ready:
                try:
                    home_renovation_application =
Homerenovationapplication.objects.select_related("renovatorcrew_id",
"home_renovation_id", "home_renovation_id__homeowner_id").get(id =
homerenovationapplication_id)
                except Homerenovationapplication.DoesNotExist:
                    home_renovation_application = None
                    if not home_renovation_application:
                        return render(request, "renovatorapp/error.html",
{"error": "Таку заявку на ремонт будинку не було знайдено."})
                    if
home_renovation_application.homerenovation_id.homeowner_id !=
homeowner_ready:
                        return render(request, "renovatorapp/error.html",
{"error": "Для оцінювання потрібно, щоб заявка, за якою це оцінювання
виконується, була вашою."})
                    if request.method == "POST":
                        renovator_rating_form =
RenovatorratingForm(request.POST)
                        if renovator_rating_form.is_valid():
                            renovator_rating_from_form =
renovator_rating_form.save(commit=False)

renovator_rating_from_form.homerenovationapplication_id =
home_renovation_application

```

```

        renovator_rating_from_form.save()
        return redirect(request.path_info)
    else:
        renovator_rating_form = RenovatorratingForm()
        return render(request, "renovatorapp/newrenovrate.html",
{"form": renovator_rating_form })

from .forms import HomeownerratingForm

def homeownerrating_add_view(request,
homerenovationapplication_id):
    homerenovator_ready =
Renovatorcrewaccount.objects.select_related("renovatorcrewaccountindja
ngo_id", "renovatorcrew_id").get(renovatorcrewaccountindjango_id =
request.user)
    if not homerenovator_ready:
        return render(request, "renovatorapp/error.html",
{"error": "Для роботи в цьому режимі потрібно бути робітником, що
авторизувався."})
    else:
        return render(request, "renovatorapp/error.html",
{"error": "Для оцінювання власника будинку потрібно бути робітником,
що авторизувався."})
    if homerenovator_ready:
        try:
            home_renovation_application =
Homerenovationapplication.objects.select_related("renovatorcrew_id",
"homerenovation_id", "homerenovation_id__homeowner_id").get(id =
homerenovationapplication_id)
        except Homerenovationapplication.DoesNotExist:
            home_renovation_application = None
        if not home_renovation_application:
            return render(request, "renovatorapp/error.html",
{"error": "Таку заявку на ремонт будинку не було знайдено."})
        if home_renovation_application.renovatorcrew_id !=
homerenovator_ready.renovatorcrew_id:
            return render(request, "renovatorapp/error.html",
{"error": "Для оцінювання потрібно, щоб заявка, за якою це оцінювання
виконується, була вашою."})
        if request.method == "POST":

```

```

        home_owner_rating_form =
HomeownerratingForm(request.POST)
        if home_owner_rating_form.is_valid():
            home_owner_rating_from_form =
home_owner_rating_form.save(commit=False)

home_owner_rating_from_form.homerenovationapplication_id =
home_renovation_application
            home_owner_rating_from_form.save()
            return redirect(request.path_info)
        else:
            home_owner_rating_form = HomeownerratingForm()
            return render(request, "renovatorapp/newownerrate.html",
{"form": home_owner_rating_form })

def homeowner_ratings_view(request, homeowner_uname = None):
    if not homeowner_uname:
        if request.user.is_authenticated:
            homeowner_ready =
Homeowner.objects.select_related("homeownerindjango_id").get(homeowner
indjango_id = request.user)
            if not homeowner_ready:
                return render(request,
"renovatorapp/error.html", {"error": "Для роботи з оцінюванням в цьому
режимі потрібно бути власником будинку, що авторизувався."})
            else:
                return render(request, "renovatorapp/error.html",
{"error": "Для роботи з оцінюванням в цьому режимі потрібно бути
власником будинку, що авторизувався."})
        else:
            try:
                homeowner_ready =
Homeowner.objects.select_related("homeownerindjango_id").get(homeowner
indjango_id__username = homeowner_uname)
            except Homeowner.DoesNotExist:
                homeowner_ready = None
            if not homeowner_ready:
                return render(request, "renovatorapp/error.html",
{"error": "Такого власника будинку не знайдено."})
            homeowner_ratings =
Homeownerrating.objects.select_related("homerenovationapplication_id",
"homerenovationapplication_id__homerenovation_id",

```

```

"homerenovationapplication_id__homerenovation_id__homeowner_id",
"homerenovationapplication_id__homerenovation_id__homeowner_id__homeow
nerindjango_id", "homerenovationapplication_id",
"homerenovationapplication_id__renovatorcrew_id",
"homerenovationapplication_id__renovatorcrew_id__renovator_id").filter
(homerenovationapplication_id__homerenovation_id__homeowner_id =
homeowner_ready)
    homeowner_ratings = homeowner_ratings.order_by("-
ratingcreatetime")
    homeowner_ratings = Paginator(homeowner_ratings, 10)
    homeowner_ratings_page = request.GET.get('page')
    homeowner_ratings =
homeowner_ratings.get_page(homeowner_ratings_page)
    return render(request, "renovatorapp/ownerratings.html",
        {"ratings": homeowner_ratings,
         "homeowner": homeowner_ready })

def renovator_ratings_view(request, renovator_id = None):
    if not renovator_id:
        if request.user.is_authenticated:
            homerenovator_ready =
Renovatorcrewaccount.objects.select_related("renovatorcrewaccountindja
ngo_id", "renovatorcrew_id",
"renovatorcrew_id__renovator_id").get(renovatorcrewaccountindjango_id
= request.user)

            if not homerenovator_ready:
                return render(request,
"renovatorapp/error.html", {"error": "Для роботи в цьому режимі
потрібно бути робітником, що авторизувався."})
            else:
                return render(request, "renovatorapp/error.html",
{"error": "Для роботи з оцінюванням в цьому режимі потрібно бути
власником робітником, що авторизувався."})
            renovator_company =
homerenovator_ready.renovatorcrew_id.renovator_id
        else:
            try:
                renovator_company =
Renovator.objects.select_related("renovationcity_id").get(id =
renovator_id)
            except Renovator.DoesNotExist:
                renovator_company = None
            if not renovator_company:

```

```

        return render(request, "renovatorapp/error.html",
{"error": "Такої компанії з ремонту будинків не знайдено."})
        renovator_ratings =
Renovatorrating.objects.select_related("homerenovationapplication_id",
"homerenovationapplication_id__homerenovation_id",
"homerenovationapplication_id__homerenovation_id__homeowner_id",
"homerenovationapplication_id__homerenovation_id__homeowner_id__homeow
nerindjango_id", "homerenovationapplication_id",
"homerenovationapplication_id__renovatorcrew_id",
"homerenovationapplication_id__renovatorcrew_id__renovator_id").filter
(homerenovationapplication_id__renovatorcrew_id__renovator_id =
renovator_company)
        renovator_ratings = renovator_ratings.order_by("-
ratingcreatetime")
        renovator_ratings = Paginator(renovator_ratings, 10)
        renovator_ratings_page = request.GET.get('page')
        renovator_ratings =
renovator_ratings.get_page(renovator_ratings_page)
        return render(request, "renovatorapp/renovratings.html",
{"ratings": renovator_ratings,
"renovator": renovator_company })

```

```

def homeowner_renovations_view(request, homeowner_urname = None):
    if not homeowner_urname:
        if request.user.is_authenticated:
            homeowner_ready =
Homeowner.objects.select_related("homeownerindjango_id").get(homeowner
indjango_id = request.user)
            if not homeowner_ready:
                return render(request,
"renovatorapp/error.html", {"error": "Для роботи з оцінюванням в цьому
режимі потрібно бути власником будинку, що авторизувався."})
            else:
                return render(request, "renovatorapp/error.html",
{"error": "Для роботи з оцінюванням в цьому режимі потрібно бути
власником будинку, що авторизувався."})
        else:
            try:
                homeowner_ready =
Homeowner.objects.select_related("homeownerindjango_id").get(homeowner
indjango_id__username = homeowner_urname)
            except Homeowner.DoesNotExist:
                homeowner_ready = None

```

```

        if not homeowner_ready:
            return render(request, "renovatorapp/error.html",
{"error": "Такого власника будинку не знайдено."})
        if request.user.is_authenticated:
            homerenovator_ready =
Renovatorcrewaccount.objects.select_related("renovatorcrewaccountindja
ngo_id", "renovatorcrew_id",
"renovatorcrew_id__renovator_id").get(renovatorcrewaccountindjango_id
= request.user)
        else:
            homerenovator_ready = None
            homeowner_renovations =
Homer renovation.objects.select_related("renovationcity_id",
"homer renovationstyle_id").filter(homeowner_id = homeowner_ready)
            homeowner_renovations = homeowner_renovations.order_by("-
homer renovationcreatetime")
            homeowner_renovations = Paginator(homeowner_renovations, 10)
            homeowner_renovations_page = request.GET.get('page')
            homeowner_renovations =
homeowner_renovations.get_page(homeowner_renovations_page)
            for homeowner_renovation in homeowner_renovations:
                homer renovation_bookmark =
Homer renovationbookmark.objects.filter(homer renovation_id =
homeowner_renovation, renovatorcrewaccount_id = homer renovator_ready)
                if homer renovation_bookmark:
                    homeowner_renovation.booked = True
                else:
                    homeowner_renovation.booked = False
                homeowner_renovation.photo =
Homer renovationphoto.objects.filter(homer renovation_id =
homeowner_renovation)
                homeowner_renovation.photo =
homeowner_renovation.photo[0]
            return render(request, "renovatorapp/ownerratings.html",
{"renovations": homeowner_renovations,
"homeowner": homeowner_ready })

```

```

def homeowner_renovation_view(request, homer renovation_id):
    try:
        home_renovation =
Homer renovation.objects.select_related("renovationcity_id",
"homer renovationstyle_id").filter(id = homer renovation)
    except Homer renovation.DoesNotExist:
        home_renovation = None

```

```

        if not home_renovation:
            return render(request, "renovatorapp/error.html",
{"error": "Такого ремонта будинку не знайдено."})
        if request.user.is_authenticated:
            homerenovator_ready =
Renovatorcrewaccount.objects.select_related("renovatorcrewaccountindja
ngo_id", "renovatorcrew_id",
"renovatorcrew_id__renovator_id").get(renovatorcrewaccountindjango_id
= request.user)
            homerenovation_bookmark =
Homer renovationbookmark.objects.filter(homerenovation_id =
home_renovation, renovatorcrewaccount_id = homerenovator_ready)
            if homerenovation_bookmark:
                home_renovation.booked = True
            else:
                home_renovation.booked = False
            home_renovation_photos =
Homer renovationphoto.objects.filter(homerenovation_id =
home_renovation)
            home_renovation_works =
Homer renovationwork.objects.select_related("renovationwork_id",
"renovationwork_id__renovationworkgroup_id").filter(homerenovation_id
= home_renovation)
            home_renovation_rooms =
Homer renovationroom.objects.filter(homerenovation_id = home_renovation)
            for home_renovation_room in home_renovation_rooms:
                home_renovation_room.photos =
Homer renovationroomphoto.objects.filter(homerenovationroom_id =
home_renovation_room)
            from datetime import datetime
            watching_time = datetime.now()
            home_renovation_applications =
Homer renovationapplicationstatus.objects.filter(renovationstatus__in =
[3, 5, 6, 7], homerenovationapplication_id__homerenovation_id =
home_renovation)
            home_renovation.applied = False
            if home_renovation_applications:
                home_renovation.applied = True
            return render(request, "renovatorapp/renovation.html",
{"renovation": home_renovation,
"photos": home_renovation_photos,
"works": home_renovation_works,
"rooms": home_renovation_rooms })

```

```

def renovator_crews_view(request, renovator_id = None):
    if not renovator_id:
        if request.user.is_authenticated:
            homerenovator_ready =
Renovatorcrewaccount.objects.select_related("renovatorcrewaccountindja
ngo_id", "renovatorcrew_id",
"renovatorcrew_id__renovator_id").get(renovatorcrewaccountindjango_id
= request.user)
            if not homerenovator_ready:
                return render(request,
"renovatorapp/error.html", {"error": "Для роботи в цьому режимі
потрібно бути робітником, що авторизувався."})
            else:
                return render(request, "renovatorapp/error.html",
{"error": "Для роботи з оцінюванням в цьому режимі потрібно бути
власником робітником, що авторизувався."})
            renovator_company =
homerenovator_ready.renovatorcrew_id.renovator_id
        else:
            try:
                renovator_company =
Renovator.objects.select_related("renovationcity_id").get(id =
renovator_id)
            except Renovator.DoesNotExist:
                renovator_company = None
            if not renovator_company:
                return render(request, "renovatorapp/error.html",
{"error": "Такої компанії з ремонту будинків не знайдено."})
            renovator_crews = Renovatorcrew.objects.filter(renovator_id
= renovator_company)
            if request.user.is_authenticated:
                homeowner_ready =
Homeowner.objects.select_related("homeownerindjango_id").get(homeowner
indjango_id = request.user)
                if homeowner_ready:
                    renovator_bookmark =
Renovatorbookmark.objects.filter(renovator_id = renovator_company,
homeowner_id = homeowner_ready)
                    if renovator_bookmark:
                        renovator_company.booked = True
                    else:
                        renovator_company.booked = False
                return render(request, "renovatorapp/renovatorcrews.html",

```

```

        {"crews": renovator_crews,
         "renovator": renovator_company })

def renovator_crew_view(request, renovatorcrew_id = None):
    if not renovatorcrew_id:
        if request.user.is_authenticated:
            homerenovator_ready =
Renovatorcrewaccount.objects.select_related("renovatorcrewaccountindja
ngo_id", "renovatorcrew_id",
"renovatorcrew_id__renovator_id").get(renovatorcrewaccountindjango_id
= request.user)
            if not homerenovator_ready:
                return render(request,
"renovatorapp/error.html", {"error": "Для роботи в цьому режимі
потрібно бути робітником, що авторизувався."})
            else:
                return render(request, "renovatorapp/error.html",
{"error": "Для роботи з оцінюванням в цьому режимі потрібно бути
власником робітником, що авторизувався."})
            renovator_crew = homerenovator_ready.renovatorcrew_id
        else:
            try:
                renovator_crew =
Renovatorcrew.objects.select_related("renovator_id",
"renovator_id__renovationcity_id").get(id = renovatorcrew_id)
            except Renovatorcrew.DoesNotExist:
                renovator_crew = None
            if not renovator_crew:
                return render(request, "renovatorapp/error.html",
{"error": "Такої бригади з ремонту будинків не знайдено."})
            renovator_crew_works =
Renovatorcrewwork.objects.select_related("renovatorwork_id",
"renovatorwork_id__renovationwork_id__renovationworkgroup_id").filter(
renovatorcrew_id = renovator_crew)
            renovator_crew_workers =
Renovatorcrewaccount.objects.select_related("renovatorcrewaccountindja
ngo_id").filter(renovatorcrew_id = renovator_crew)
            if request.user.is_authenticated:
                homeowner_ready =
Homeowner.objects.select_related("homeownerindjango_id").get(homeowner
indjango_id = request.user)
                if homeowner_ready:

```

```
        renovator_bookmark =
Renovatorbookmark.objects.filter(renovator_id =
renovator_crew.renovator_id, homeowner_id = homeowner_ready)
        if renovator_bookmark:
            renovator_crew.booked = True
        else:
            renovator_crew.booked = False
    return render(request, "renovatorapp/renovatorcrew.html",
        {"crew": renovator_crew,
         "works": renovator_crew_works,
         "workers": renovator_crew_workers })
```

ДОДАТОК В
Слайди презентації

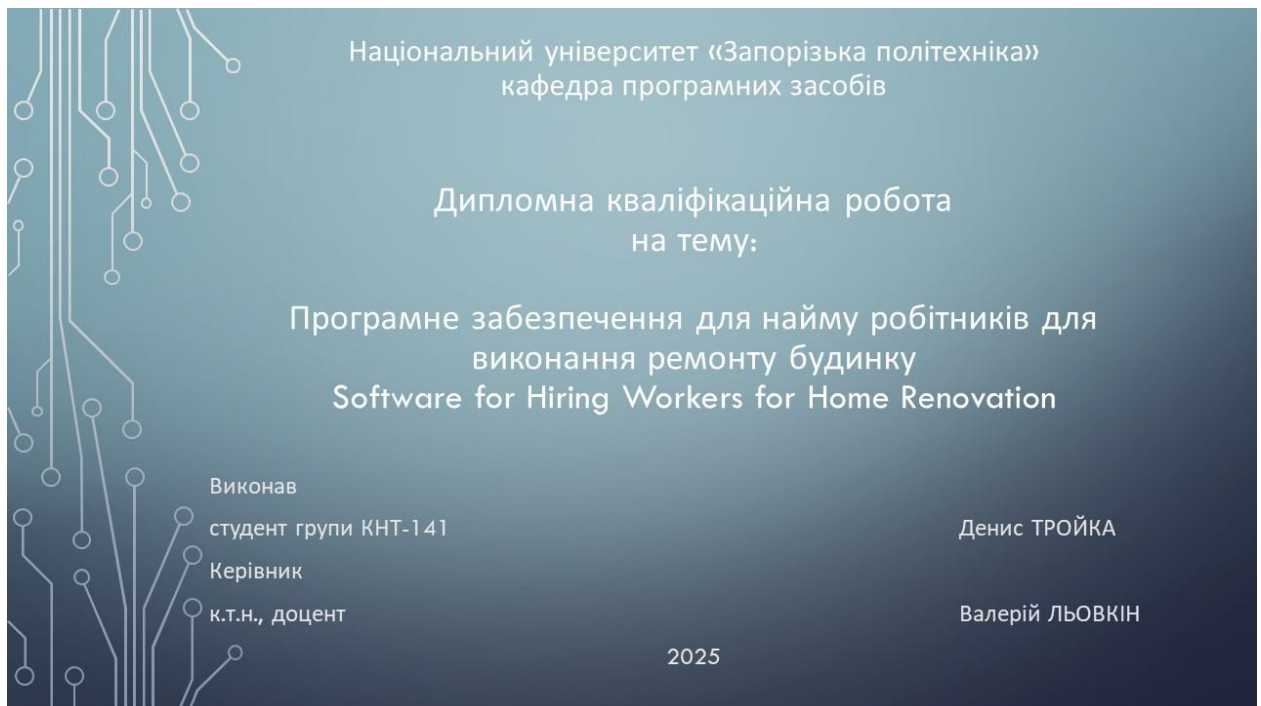


Рисунок В.1 – Початковий слайд

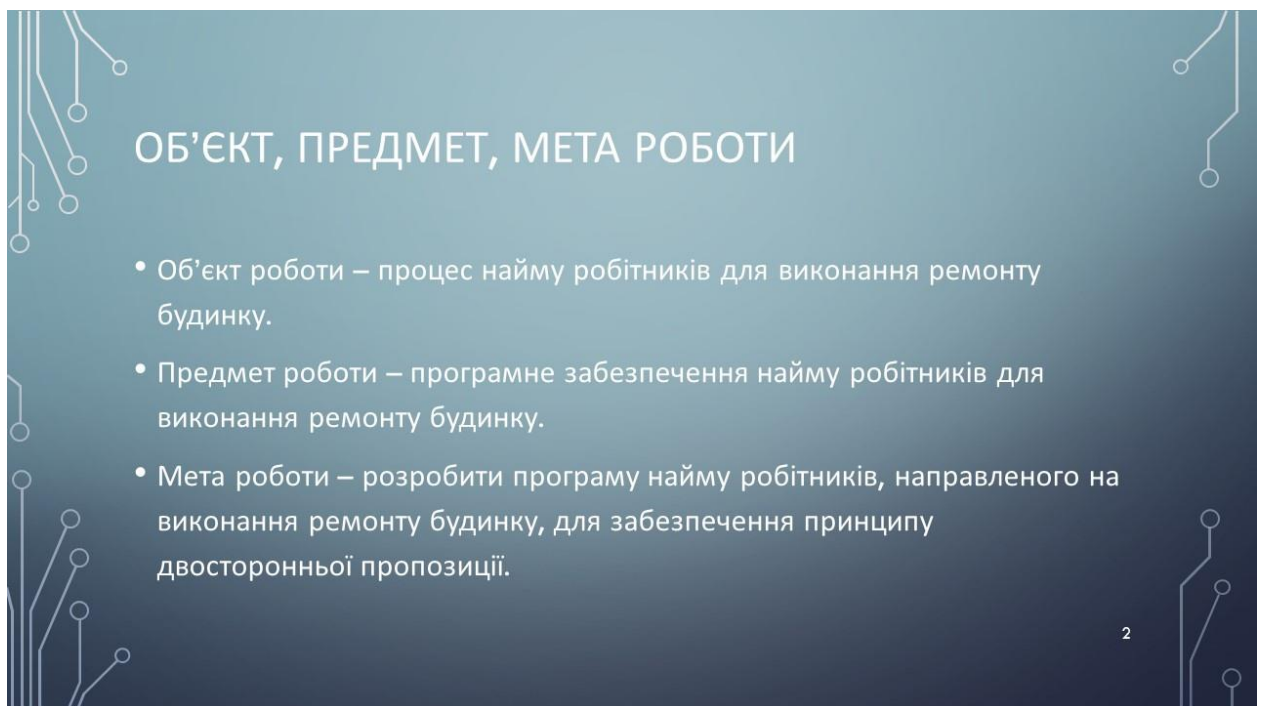


Рисунок В.2 – Об'єкт, предмет, мета роботи

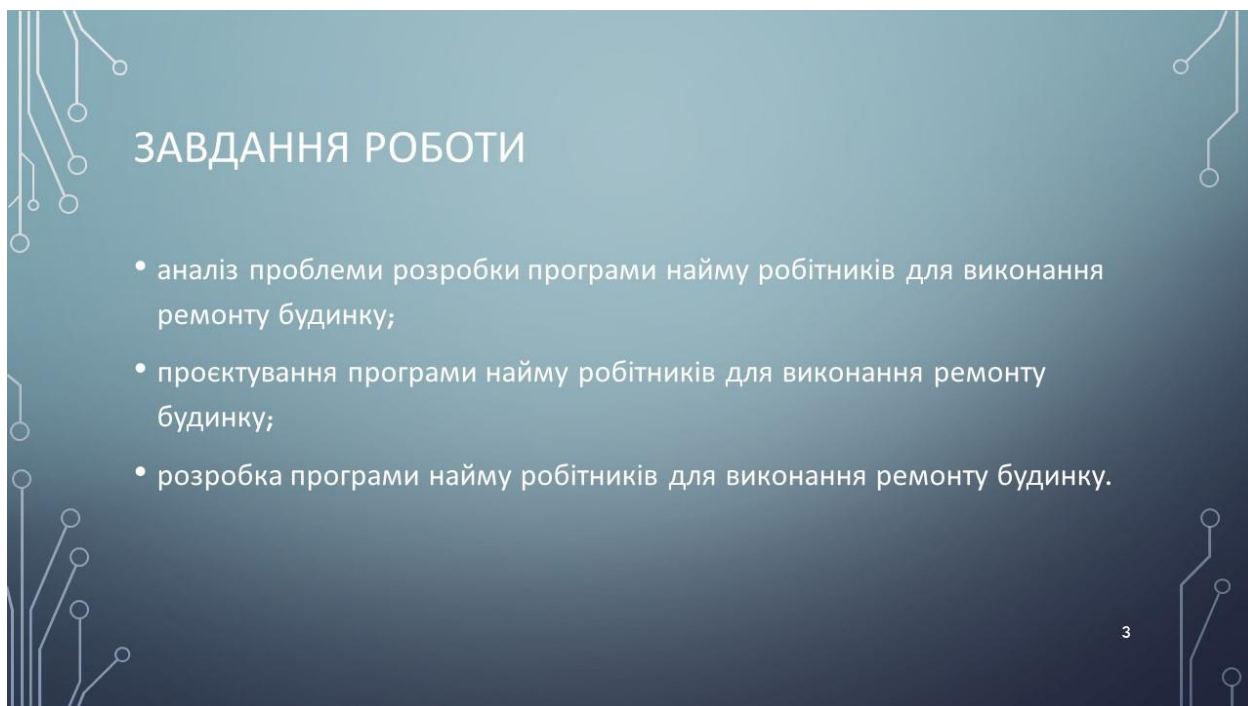


Рисунок В.3 – Завдання роботи

ПОРІВНЯННЯ ПРОГРАМ НАЙМУ РОБІТНИКІВ ДЛЯ ВИКОНАННЯ РЕМОНТУ БУДИНКУ

Характеристика	Rabotniki.ua	Ooch.com.ua	Розробка
Оголошення про необхідність ремонту будинків від власників	–	–	+
Оголошення про ремонтні послуги робітників	+	+	+
Визначення пропонованого плану робіт ремонту	–	–+	+
Замовлення ремонту з параметрами	–	–	+
Перегляд доступності бригади або робітника для виконання ремонту	–	–	+
Визначення портфолію робіт	+	+	+

4

Рисунок В.4 – Порівняння програм найму робітників для виконання ремонту будинку

ВИБІР ЗАСОБІВ РОЗРОБКИ ПРОГРАМИ НАЙМУ РОБІТНИКІВ ДЛЯ ВИКОНАННЯ РЕМОНТУ БУДИНКУ		
Характеристика	Pyramid	Django
Мова програмування	Python	Python
Відносний час розробки	Більший	Менший
Підтримка роботи з користувачами для програми	Базово відсутня	Забезпечується авторизація всіх користувачів, підтримка адміністраторів
Взаємодія з базою даних	Необхідно налаштувати	Встановлено

Рисунок В.5 – Вибір засобів розробки програми найму робітників для виконання ремонту будинку



Рисунок В.6 – Діаграми прецедентів програми найму робітників для виконання ремонту будинку

СХЕМА БАЗИ ДАНИХ ПРОГРАМИ НАЙМУ РОБІТНИКІВ ДЛЯ ВИКОНАННЯ РЕМОНТУ БУДИНКУ

Рисунок В.8 – Схема бази даних програми найму робітників для виконання ремонту будинку

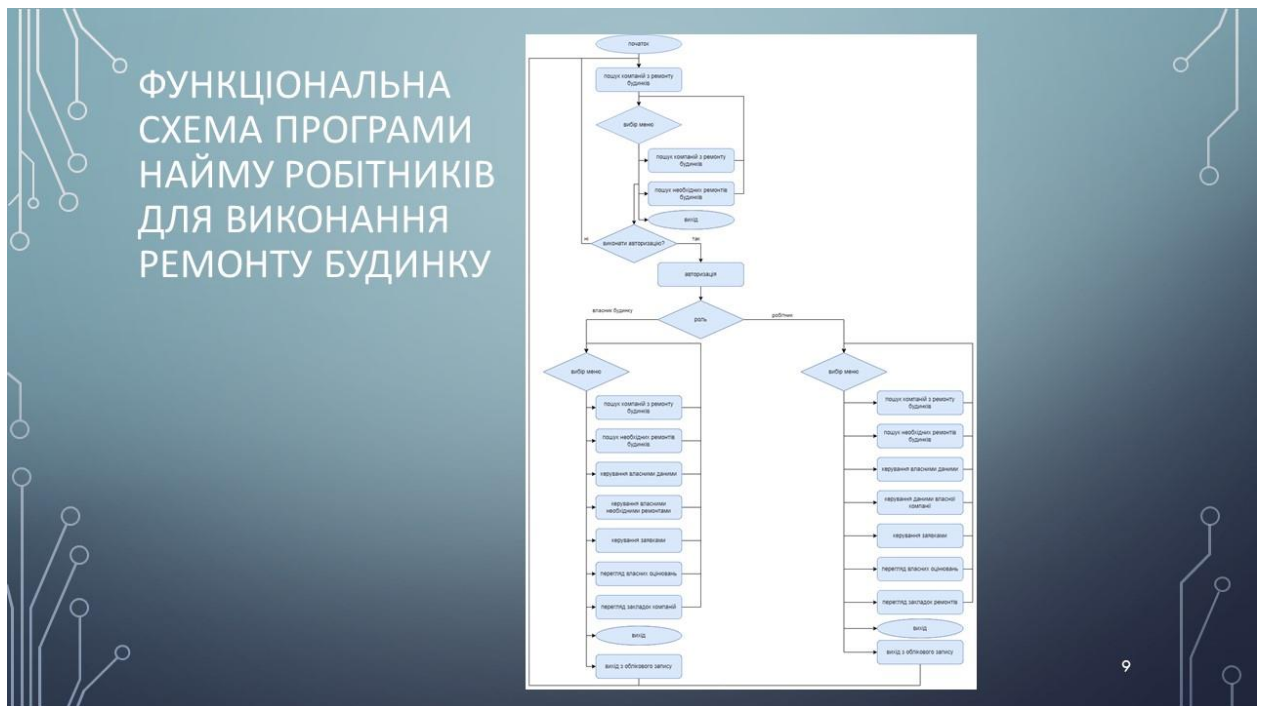


Рисунок В.9 – Функціональна схема програми найму робітників для виконання ремонту будинку

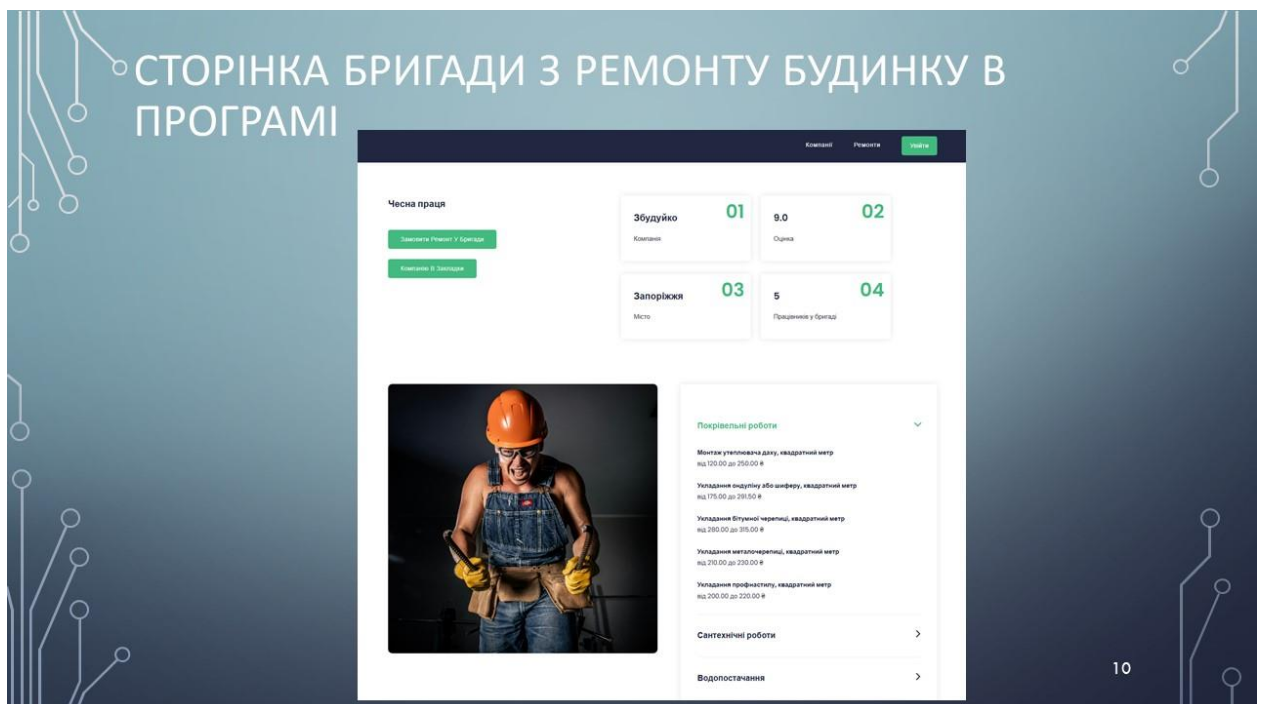


Рисунок В.10 – Сторінка бригади з ремонту будинку в програмі

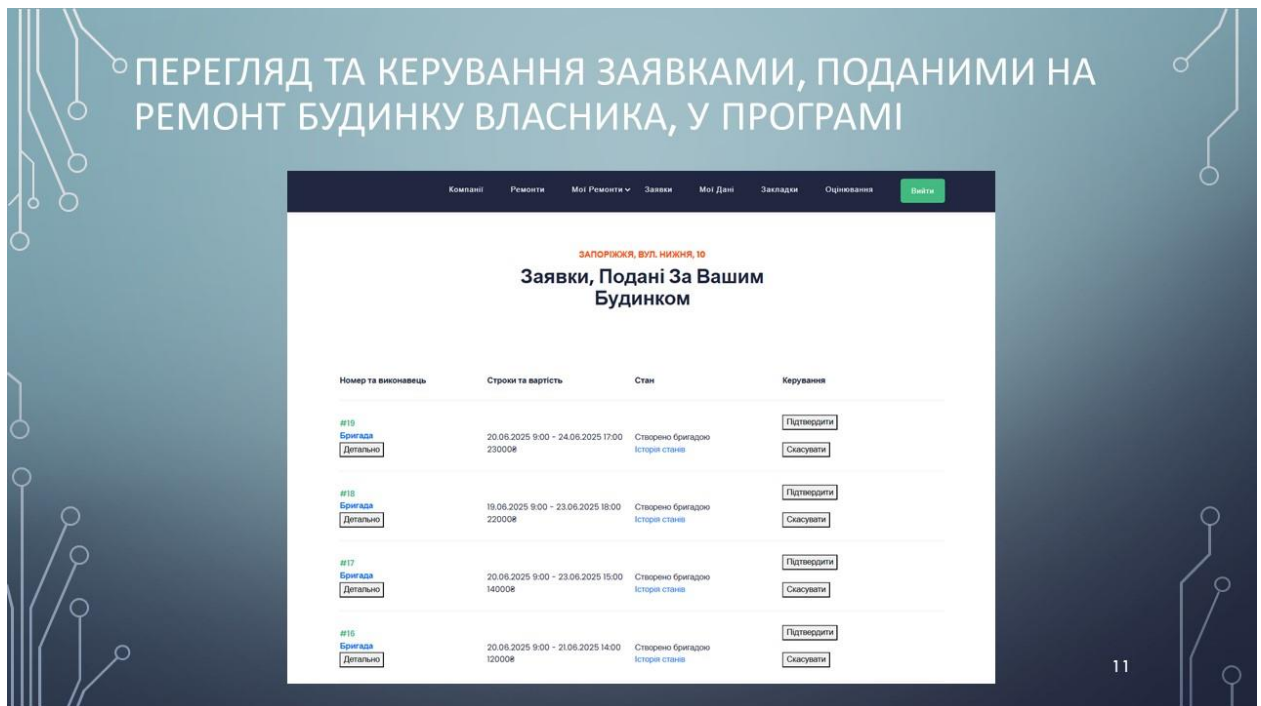


Рисунок В.11 – Перегляд та керування заявками, поданими на ремонт будинку власника, у програмі

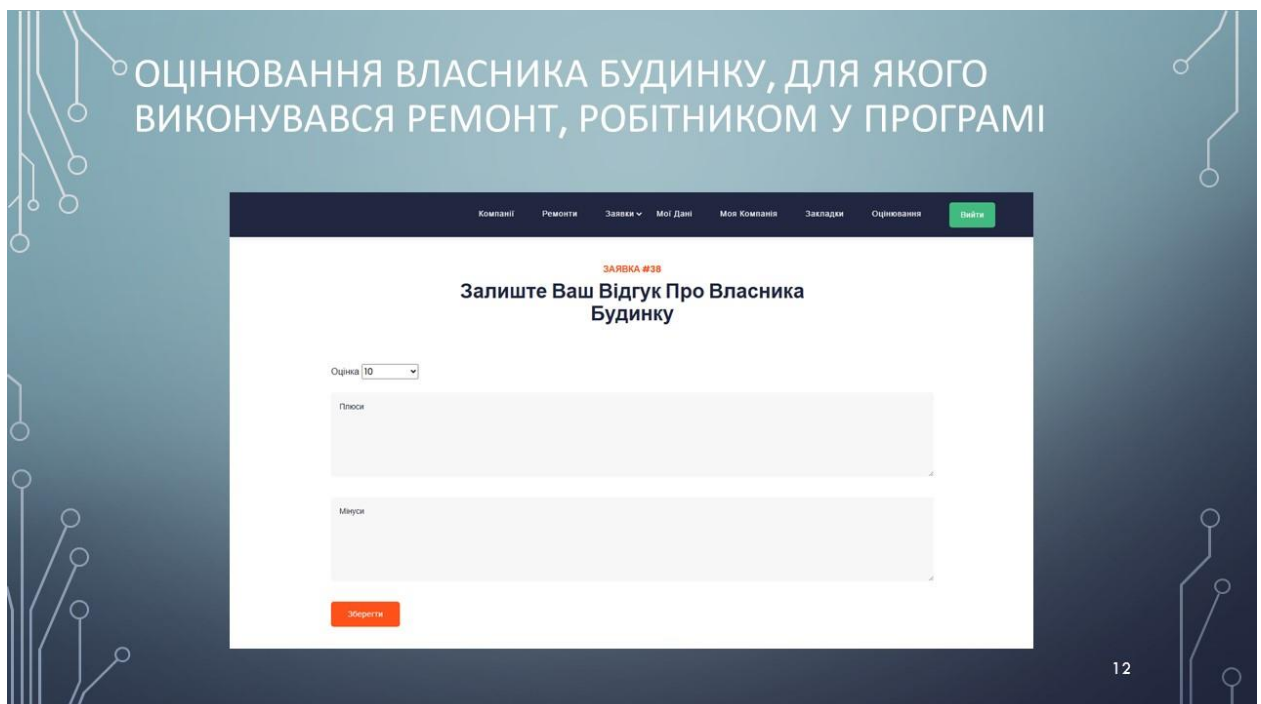


Рисунок В.12 – Оцінювання власника будинку, для якого виконувався ремонт, робітником у програмі

ТЕСТУВАННЯ ПРОГРАМИ НАЙМУ РОБІТНИКІВ ДЛЯ ВИКОНАННЯ РЕМОНТУ БУДИНКУ												
Функція	Користувач до авторизації	Власник будинку	Робітник з ремонту будинків	Функція	Користувач до авторизації	Власник будинку	Робітник з ремонту будинків	Функція	Користувач до авторизації	Власник будинку	Робітник з ремонту будинків	
Перегляд переліку ремонтних робіт, які здали виконавці компанії	+	+	+	Управління станом заявок на виконання ремонту будинку		+	+	Вилучення ремонтної роботи для компанії з ремонту будинків			+	
Перегляд даних власника будинку	+	+	+	Перегляд заявок на виконання ремонту будинку		+	+	Додавання бригади з ремонту будинків компанії			+	
Перегляд даних компанії з ремонту будинків	+	+	+	Перегляд заявок на виконання ремонту за будинком		+	+	Оновлення даних бригади з ремонту будинків компанії			+	
Авторизація		+	+	Перегляд власних поданих заявок на виконання ремонту будинку		+	+	Вилучення даних робітника з ремонту будинків			+	
Перегляд переліку бригад з ремонту будинків компанії	+	+	+	Перегляд станів, які визначилися за заявкою на виконання ремонту будинку		+	+	Перегляд складу робітників бригади з ремонту будинків			+	
Перегляд даних бригади з ремонту будинків	+	+	+	Оцінювання компанії з ремонту будинків		+		Додавання часового лоту для роботи бригади з ремонту будинків			+	
Перегляд доступних для роботи бригад у компанії з ремонту будинків на заданий час	+	+	+	Перегляд оцінень власником будинку		+		Оновлення часового лоту для роботи бригади з ремонту будинків			+	
Перегляд переліку необхідних ремонтів будинків власника	+	+	+	Перегляд оцінень компанії з ремонту будинків власником будинку		+		Вилучення часового лоту для роботи бригади з ремонту будинків			+	
Перегляд оголошень про необхідність ремонту будинку	+	+	+	Створення закладок для компанії з ремонту будинку		+		Додавання ремонтної роботи до можливостей бригади			+	
Перегляд оцінювань власника будинку	+	+	+	Вилучення закладок для компанії з ремонту будинку		+		Вилучення ремонтної роботи з можливостей бригади			+	
Перегляд оцінювань компанії з ремонту будинків	+	+	+	Перегляд переліку компаній з ремонту будинку в закладах		+		Перегляд заявок на виконання ремонту будинку			+	
Пошук оголошень про необхідність ремонту будинку	+	+	+	Визначення власної компанії з ремонту будинків			+	Додавання роботи до плану ремонту будинку за заявкою			+	
Пошук компанії з ремонту будинку	+	+	+	Оновлення даних компанії з ремонту будинків			+	Вилучення роботи з плану ремонту будинку за заявкою			+	
Рестрація облікового запису як власника будинку		+		Додавання ремонтної роботи для компанії з ремонту будинків			+	Оновлення роботи з плану ремонту будинку за заявкою			+	
Рестрація облікового запису як робітника з ремонту будинків			+	Оновлення даних компанії з ремонту будинків			+	Оцінювання власника будинку			+	
Визначення даних власника будинку		+		Оновлення ремонтної роботи для компанії з ремонту будинків			+	Перегляд оцінень власників будинків бригадою з ремонту будинків			+	
Створення оголошення про необхідний ремонт будинку		+		Подання заявки на виконання ремонту будинку		+	+	Створення закладок за ремонтом будинку			+	
Оновлення оголошення про необхідний ремонт будинку		+		Додавання роботи до плану ремонту будинку		+		Вилучення закладок за ремонтом будинку			+	
Вилучення оголошення про необхідний ремонт будинку		+		Вилучення роботи з плану ремонту будинку		+		Перегляд переліку ремонтів будинків у закладах			+	
Додавання кімнати до даних ремонту будинку		+		Оновлення роботи з плану ремонту будинку		+						
Оновлення кімнати з даних ремонту будинку		+										
Вилучення кімнати з даних ремонту будинку		+										

13

Рисунок В.13 – Тестування програми найму робітників для виконання ремонту будинку

ВИСНОВКИ												
У роботі було розроблено програму найму робітників, направленою на виконання ремонту будинку, для забезпечення принципу двосторонньої пропозиції. Виконання роботи базувалося на аналізі сучасного стану розробки програм, які можуть застосовуватися для найму робітників для ремонту будинків, виділяючи від них основні позитивні сторони та беручи до разунки можливості подолання негативних сторін. Розроблена програма призначена для пошуку робітників для виконання ремонту будинку з об'єднанням таких робітників у бригади, компанії з ремонту будинків, пошуку оголошень про необхідний ремонт будинку, забезпечення заявок для реалізації ремонту будинків, управління цими заявками.												

14

Рисунок В.14 – Висновки