МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Сумський державний університет

Факультет електроніки та інформаційних технологій

Кафедра комп'ютерних наук

«До захисту допущено»
В.о. завідувача кафедри
Ігор ШЕЛЕХОВ
(підпис)
червня 2024 р.

КВАЛІФІКАЦІЙНА РОБОТА
на здобуття освітнього ступеня бакалавр

зі спеціальності 122 «Комп'ютерні науки»,
освітньо-професійної програми «Інформатика»
на тему: «Інформаційне та програмне забезпечення мобільного додатку супроводження діяльності рієлтора»
здобувача групи ІН-02 Горбенка Данііла Андрійовича

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

Данііл ГОРБЕНКО
(підпис)

Керівник, ст. викл.,
канд. фіз.-мат. наук, доц.
Оксана ШОВКОПЛЯС
(підпис)

Суми – 2024

Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра комп'ютерних наук

«Затверджую»
В.о. завідувача кафедри
_____ Ігор ШЕЛЕХОВ
(підпис)

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня бакалавра

зі спеціальності 122 «Комп'ютерні науки», освітньо-професійної програми «Інформатика»
здобувача групи ІН-02 Горбенка Данііла Андрійовича

1. Тема роботи: «Інформаційне та програмне забезпечення мобільного додатку супроводження діяльності рієлтора»
затверджую наказом по СумДУ від *«22» квітня 2024 р. №* 0414-VI
2. Термін здачі здобувачем кваліфікаційної роботи *до 01 червня 2024 року*
3. Вхідні дані до кваліфікаційної роботи
4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити) *1) Аналіз проблеми предметної області, постановка й формування завдань дослідження. 2) Огляд технологій, що використовуються для створення мобільних додатків. 3) Розробка інформаційного та програмного забезпечення мобільного додатку супроводження діяльності рієлтора. 4) Аналіз результатів*
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
6. Консультанти до проекту (роботи), із значенням розділів проекту, що стосується їх

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання «06» травня 2024 р.
Завдання прийняв до виконання _____ Керівник _____
(підпис) (підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Термін виконання	Примітка
1	*Аналіз проблеми предметної області, існуючих рішень, постановка й формування завдань дослідження*	06.05.24 - 09.05.24	
2	*Огляд технологій, що використовуються для створення мобільних додатків*	09.05.24 - 12.05.24	
3	*Розробка інформаційного та програмного забезпечення мобільного додатку супроводження діяльності рієлтора*	12.05.24 - 28.05.24	
4	*Аналіз отриманих результатів*	28.05.24 - 29.05.24	
5	*Оформлення пояснювальної записки до кваліфікаційної роботи*	29.05.24 - 31.05.24	

Здобувач вищої освіти _____ Керівник _____
(підпис) (підпис)

АНОТАЦІЯ

Записка: 153 стр., 110 рис., 2 додатки, 25 використаних джерел.

Обґрунтування актуальності теми роботи – Тема кваліфікаційної роботи є актуальною, оскільки присвячена розв’язанню важливої практичної задачі цифровізації робочого процесу рієлторів агентства нерухомості шляхом розробки відповідних інформаційного та програмного забезпечення.

Об’єкт дослідження — процес операцій та опрацювання даних з боку рієлторів під час операцій купівлі/продажу та (або) оренди/здавання в оренду.

Мета роботи — розробка інформаційного та програмного забезпечення мобільного додатку супроводження діяльності рієлторів з попереднім проєктуванням та використанням сучасних засобів розробки.

Методи дослідження — аналогія, аналітичне узагальнення та порівняльний аналіз програмних продуктів з схожим функціоналом, аналіз, опис, узагальнення засобів реалізації інформаційного та програмного забезпечення.

Результати — розроблено інформаційне та програмне забезпечення мобільного додатку супроводження діяльності рієлторів, що дозволяє рієлторам агентсва нерухомості частково проводити робочий процес в онлайн-форматі.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ І ТЕРМІНІВ

COVID – **CO**rona**VI**rus **D**isease – коронавірусна хвороба.

iOS – IPhone Operating System – операційна система iPhone.

УНІАН – Українське незалежне інформаційне агентство новин.

JVM – Java Virtual Machine – віртуальна машина Java.

OS – Operating System – операційна система.

.NET – Network Enabled Technologies – мережеві технології.

IoT – Internet of Things – інтернет речей, система фізичних пристроїв, що взаємопов’язані між собою через програмне забезпечення та через інші технології.

CLR – Common Language Runtime – загальномовне виконуюче середовище для виконання коду, написаного на мовах програмування платформи .NET.

CTS – Common Type System – система спільних типів платформи .NET.

API – Application Programming Interface – інтерфейс програмування додатків.

ASP.NET – Active Server Pages for .NET – активні серверні сторінки для плафторми .NET, технологія створення веб-застосунків та сервісів.

SQL – Structured Query Language – структурована мова запитів до реляційних баз даних.

IDE – Integrated Development Environment – інтегроване середовище розробки.

XML – eXtensible Markup Language – розширювана мова розмітки.

Activity – Актівіті – сторінка додатку.

ВСТУП

Актуальність. Тема кваліфікаційної роботи є актуальною, оскільки присвячена розв'язанню важливої практичної задачі цифровізації робочого процесу рієлторів агентства нерухомості шляхом розробки відповідних інформаційного та програмного забезпечення.

Об’єкт дослідження. Процес операцій та опрацювання даних з боку рієлторів під час операцій купівлі/продажу та (або) оренди/здавання в оренду.

Предмет дослідження. Методи та технології, що використовуються для створення інформаційного та програмного забезпечення мобільного додатку супроводження діяльності рієлторів.

Гіпотеза. Застосування інформаційного та програмного забезпечення для мобільного додатку супроводження діяльності рієлторів дозволить ефективно цифровізувати процеси купівлі/продажу та оренди/здавання в оренду нерухомості, що значно підвищить ефективність їхньої діяльності та зручність управління інформацією.

Новизна. Розроблено унікальне інформаційне та програмне забезпечення мобільного додатку, що дозволяє рієлторам агентства нерухомості частково проводити робочий процес в онлайн-форматі, забезпечуючи високий рівень продуктивності. Мобільний додаток інтегрує ключові функції для цифровізації діяльності рієлторів, забезпечуючи зручність, підвищену продуктивність та безпеку. Ця розробка поєднує перевірені методи з новими підходами, що дозволяє підвищити ефективність роботи агентств нерухомості.

Апробація матеріалів роботи. Основні результати роботи оприлюднені

та обговорені на міжнародній науково-технічній конференції студентів та молодих вчених «Інформатика, математика, автоматика» (ІМА – 2024).

Структура. Робота складається з кількох ключових розділів: у вступі обґрунтовується актуальність та новизна дослідження, аналітичний огляд сучасного стану інформаційних систем для агентств нерухомості, аналіз існуючих програмних рішень та визначення завдань дослідження, обґрунтування вибору інструментів та технологій для розробки мобільного додатку, детальний опис розробленого програмного забезпечення, висновки, де систематизуються отримані результати, список використаних джерел та додатки.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Визначення актуальності проблеми

Особи можуть проводити операції щодо купівлі/продажу нерухомості власноруч між двома контрагентами. Проте, купівля/продаж нерухомості власноруч тягне за собою багато ризиків та необхідність оброблювати та проводити операції з великою кількістю документів. Серед ризиків найбільшим є, як і в багатьох сферах торгівлі, шахрайство. Шахрайство може включати в себе махінації з вартістю об’єкта нерухомості, неоплачені комунальні платежі, проблеми з документами (їх фальсифікація), обман з метою привласнити гроші без передачі об’єкта у чужу власність, видавання себе за власника об’єкта, котрим особа не є насправді, продаж без згоди співвласника та/або прописаних мешканців, продаж зааре-штованого об’єкта, не підтверджене нотаріально внесення завдатку, тощо [1]. Дані ризики роблять торгівлю нерухомістю досить небезпечним інструментом.

Якщо особа вже й вирішила займатися купівлею нерухомості самостійно, то без особливих ризиків це можна зробити, наприклад, у перевірених забудовників напряму. Але й у такому випадку відсутня повна гарантія, що забудовник не отримає гроші від інвесторів-майбутніх жителів, умовно, нового багатоквартирного будинку, і не припинить будівництво у випадковий момент, залишаючи гроші собі, а інвесторів без їх законного об’єкта нерухомості. Проте, у більшості таких випадків інвестори мають достатньо доказів, щоб звернутися до суду та отримати власні кошти назад [2].

Проте завжди краще мати поряд людину, яка компетентна у справі та має змогу допомогти провести операції з нерухомістю без особливих ризиків. Таким чином з’явилося поняття «**рієлтор**». Рієлтор – агент, що здійснює посередницьку діяльність між покупцем та продавцем (орендарем та орендодавцем) у питаннях нерухомості [3]. Агентства нерухомості - підприємства, що мають у своєму складі працівників-рієлторів та здійснюють аналогічну посередницьку діяльність. Серед задач рієлтора можуть бути:

- відповідальність за правомірність угоди;
- звітування щодо виконаної роботи;
- здійснювання контролю за виконанням прописаної для контрагентів угодою;
- допомога у оформленні необхідних документів;
- допомога у погодженні умов для укладання угоди;
- проведення консультацій для клієнтів;
- допомога у огляді об’єктів нерухомості. [3]

Повертаючись до шахрайства, потенційні та компетентні рієлтори відіграють велику роль у забезпеченні безпеки угоди між двома сторонами угоди. За даними інформаційного агентства УНІАН, станом на довоєнний 2021 рік частка угод на вторинному ринку нерухомості в Україні, які пов’язані з шахрайством, становить від 1.5 до 2 відсотків. Цей показник був набагато більшим у 2016 році і становив на 35-40 відсотків більше, ніж у 2021 році. Зниження показника спровоковано збільшенням кількості угод, що були проведенні за участі рієлторів, на 15 відсотків щорічно [4]. Під час повномасштабного вторгнення, у зв’язку з необхідністю переселенцям знайти нове житло, показник шахрайства, особливо у сфері оренди нерухомості, збільшився в декілька разів [5], що насамперед пов’язано з проведенням таких операцій без участі рієлторів.

Нині більша частина діяльності рієлтора проходить в офлайн форматі. Це включає багато паперової роботи, котру необхідно проводити офлайн на робочому місці, облік клієнтів, об’єктів та угод у паперовому вигляді або, в кращому випадку, в програмах пакету Microsoft Office десь на робочому місці. Це може бути досить незручно в деяких обставинах, коли рієлтор не може знаходитися фізично в офісі агентства. Проте технології не стоять на місці. У сучасному світі, світі, у котрому домінують комп’ютери, всі сфери життя мають тенденцію до цифровізації. Комп’ютери стали доступними для широкої сфери суспільства, звичайні стільникові кнопкові телефони перетворилися у міні-комп’ютери з сенсорними екранами прямо у людини в кишені та стали називатися

«смартфонами». З них можна мати доступ до мережі інтернет, встановлювати різноманітні додатки для різних потреб(проводити час за іграми, спілкування в соцмережах, додатки для роботи, тощо), можна мати навіть власні державні документи – і все це у невеликому пристрої, котрий дійсно може вміститися у кишені. Яскравим прикладом цифровізації може бути Україна. Україна є першою державою у світі, яка визнала цифрове посвідчення особи на рівні офіційного документа. На даний момент близько 19 мільйонів громадян України мають на своїх смартфонах встановлений додаток «Дія». Даний додаток став одним з перших та головних результатів створення Міністерства цифрової трансформації. «Дія» надає можливість громадянам України мати майже весь набір офіційних документів прямо у смартфоні – від паспорта до посвідчення водія, від номера картки платника податків до студентського квитка. Крім документів, важливий функціонал додатку включає різноманітні державні послуги, котрі раніше були доступні лише в офлайн-режимі у відповідних установах. Такі послуги включають реєстрацію місця проживання, оформлення пенсії, оплата штрафів за порушення правил дорожнього руху, зміна місця проживання, судові послуги, COVID-сертифікати, електронний підпис та багато інших державних послуг [6]. Зокрема, повертаючись до теми купівлі нерухомості, в кінці 2023 року у додатку з’явилася можливість купівлі квартир та будинків не старше трьох років за пільговою іпотекою від держави. І все це – у додатку на смартфоні, а також в аналогічному інтернет-сайті.

Більшість смартфонів нині працюють на базі операційної системи Android. За різними даними станом на кінець 2023 – початок 2024 років, частка смартфонів на базі Android становить близько 71.74% від світового ринку. Найближчий конкурент Android, операційна система iOS, займає приблизно 27.63% за аналогічним показником (рис. 1.1), випереджаючи Android лише в окремих країнах, наприклад, Сполучених Штатах Америки (56.6%) (табл. 1.1). Станом на кінець 2023 року, близько 3.3 мільярди людей користуються смартфонами саме з Android [7]:

Operating System	Percentage Of Mobile Devices
Android	71.74%
iOS	27.63%
Samsung	0.35%
Unknown	0.12%
KaiOS	0.11%
Windows	0.02%

Рисунок 1.1 – Частки операційних систем на ринку смартфонів у всьому світі станом на кінець 2023 року

Таблиця 1.1 – Частки операційних систем Android та iOS у Сполучених Штатах Америки

Країна	Частка мобільних пристроїв на Android, %	Частка мобільних пристроїв на iOS, %
Сполучені Штати Америки	43	56.6

1.2 Аналіз програмних продуктів-аналогів

Перед початком роботи над проєктом, доцільно провести аналіз програмних продуктів, які можуть служити аналогами. Насамперед, нас цікавить можливості програмних продуктів для рієлторів. Виконавши пошук, розглянемо деякі потенційні аналоги. Виокремимо наступні web-ресурси/мобільні додатки:

- rieltor.ua;
- dom.ria.com;
- zlat.com.ua;
- сторінка агентства нерухомості «Прем’єр» у соціальній мережі Instagram.

Rieltor.ua – це web-додаток, сайт, спрямований на допомогу людям у знаходженні житла для покупки, продажу, чи оренди, а також, в першу чергу,

для рієлторів та агентств нерухомості. Зовнішій вигляд головної сторінки сайту наведений на рисунку 1.2:

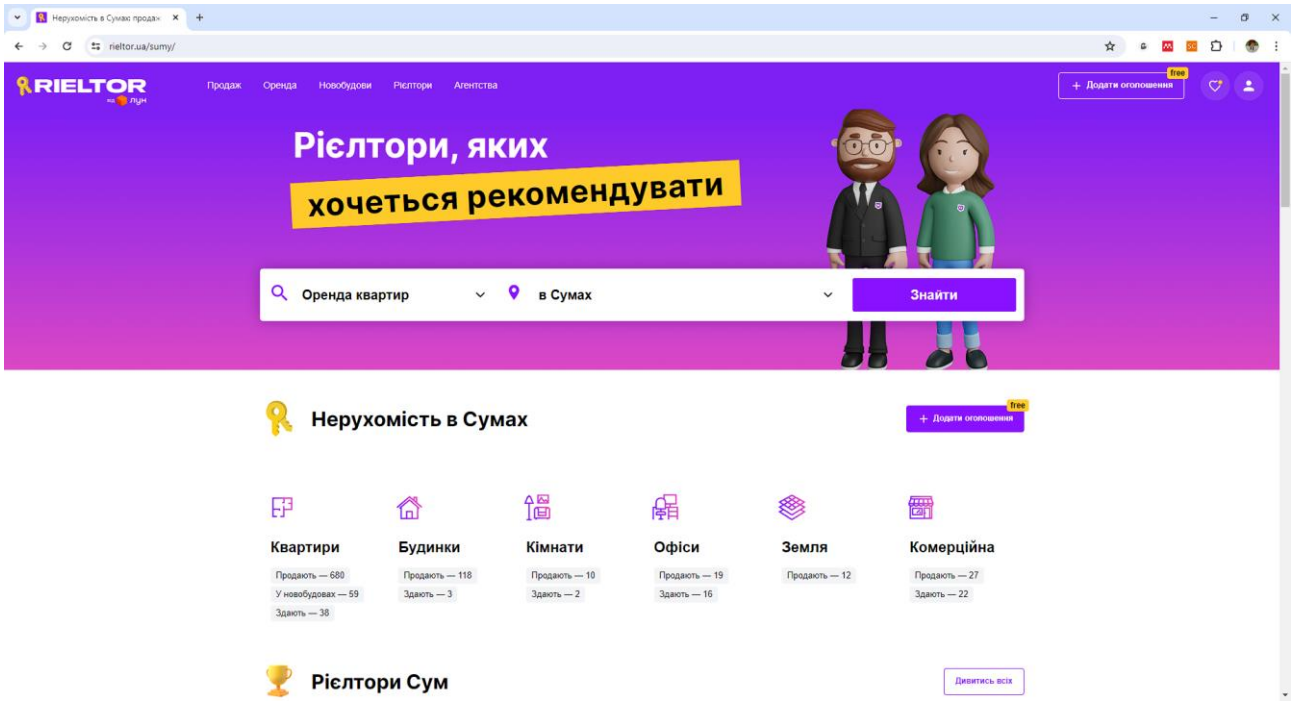


Рисунок 1.2 – Зовнішній вигляд головної сторінки сайту rieltor.ua

Сайт дає змогу шукати об’єкти нерухомості у бажаних регіонах України (рис. 1.3), переглядати їх фотографії, характеристики, зв’язуватися з рієлторами, що відповідають за їх продаж або оренду (рис. 1.4):

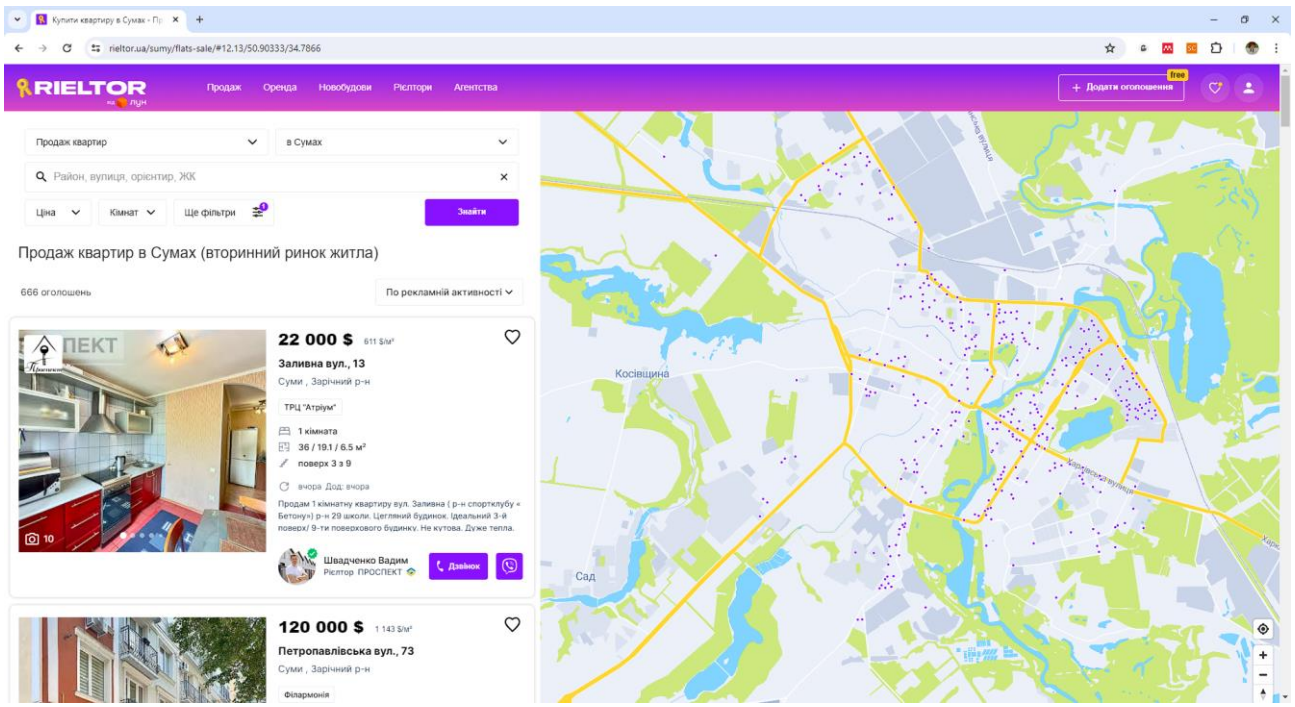


Рисунок 1.3 – Пошук об’єктів нерухомості на сайті rieltor.ua

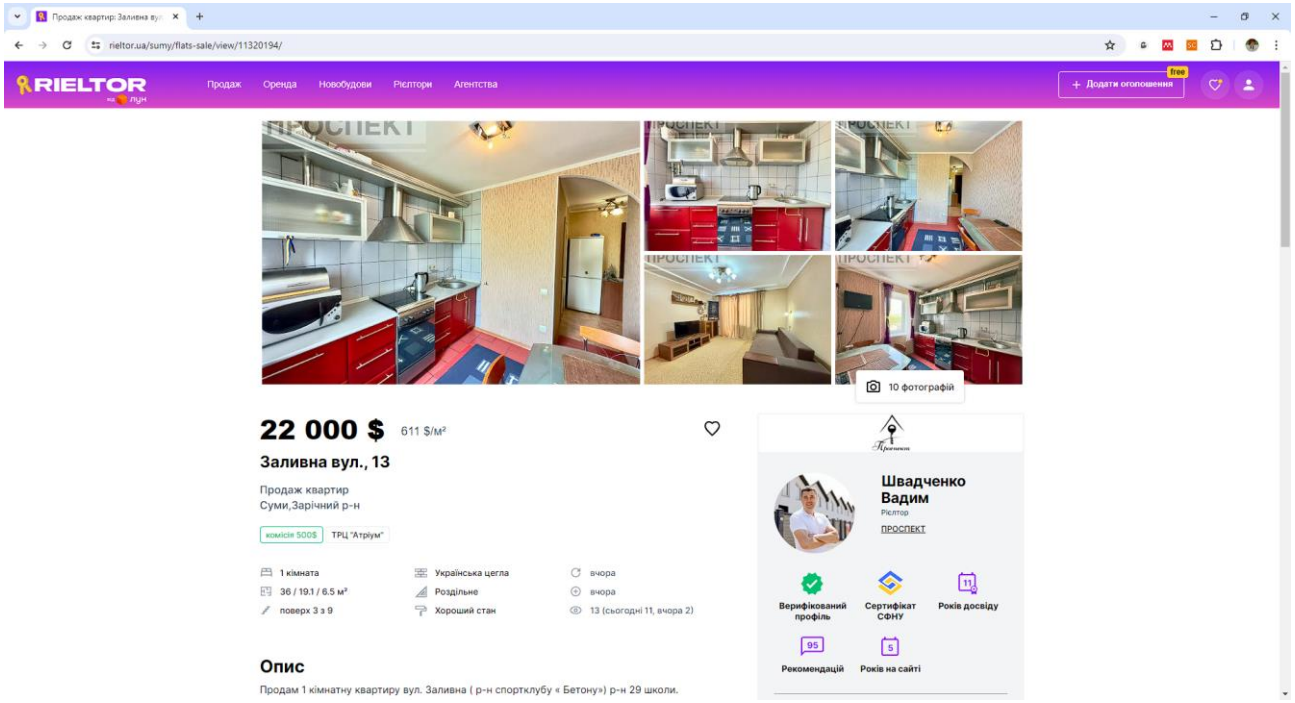


Рисунок 1.4 – Перегляд детальних характеристик об’єкта нерухомості на сайті rieltor.ua

Також сайт дає змогу знаходити агентства нерухомості (рис. 1.5) та окремих рієлторів (рис. 1.6):

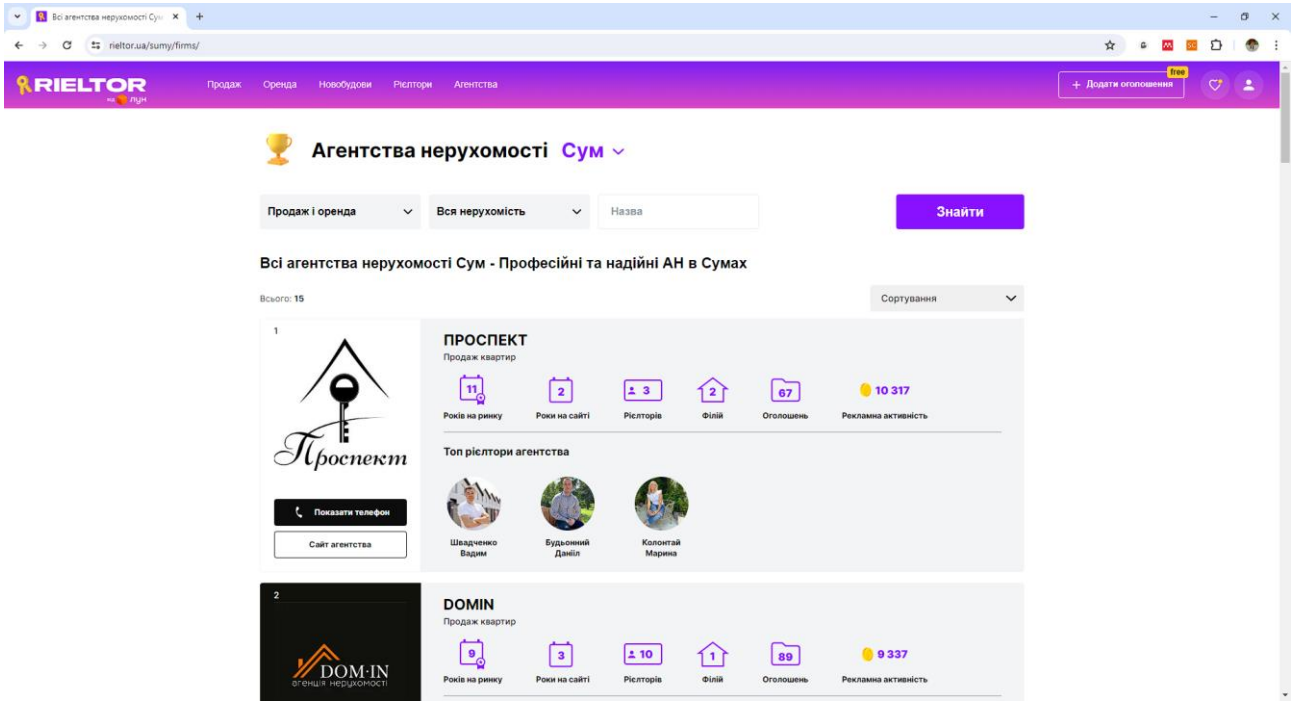


Рисунок 1.5 – Пошук агентств нерухомості на сайті rieltor.ua

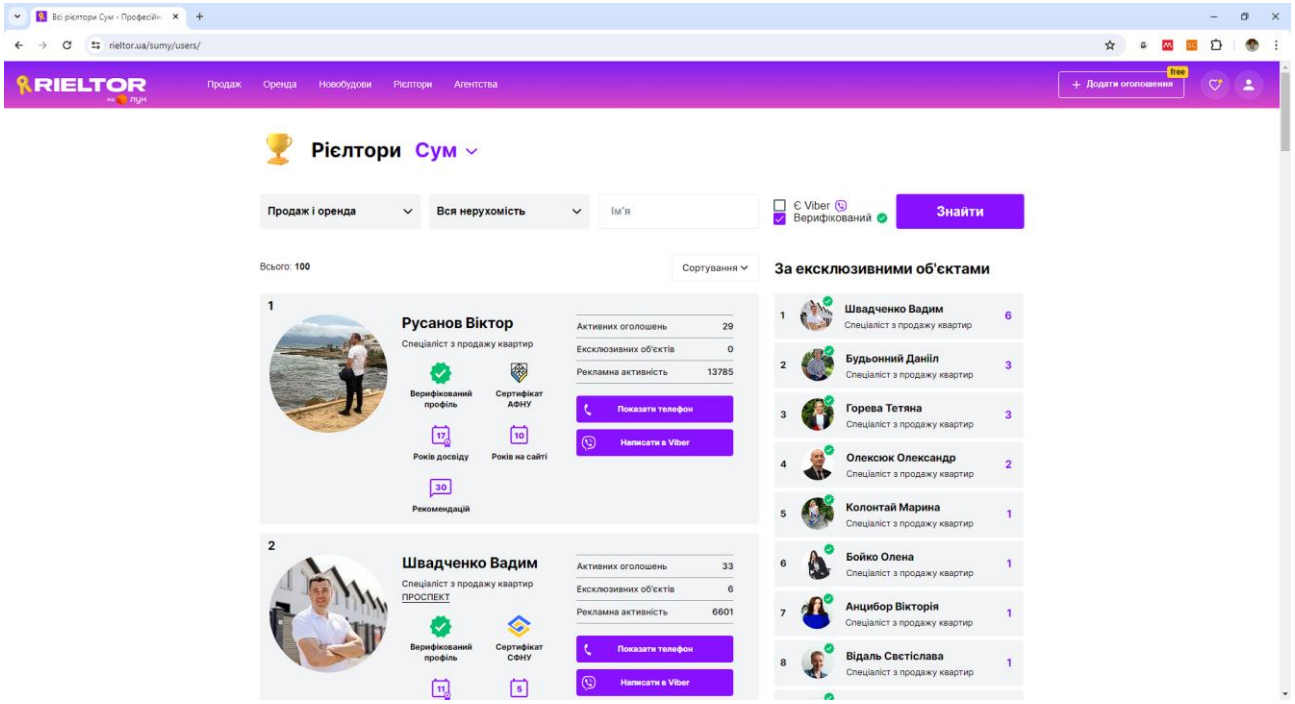


Рисунок 1.6 – Пошук рієлторів на сайті rieltor.ua

Для рієлторів та агентств нерухомості доступна можливість реєстрації особистого кабінету. Зовнішній вигляд особистого кабінету (рис. 1.7):

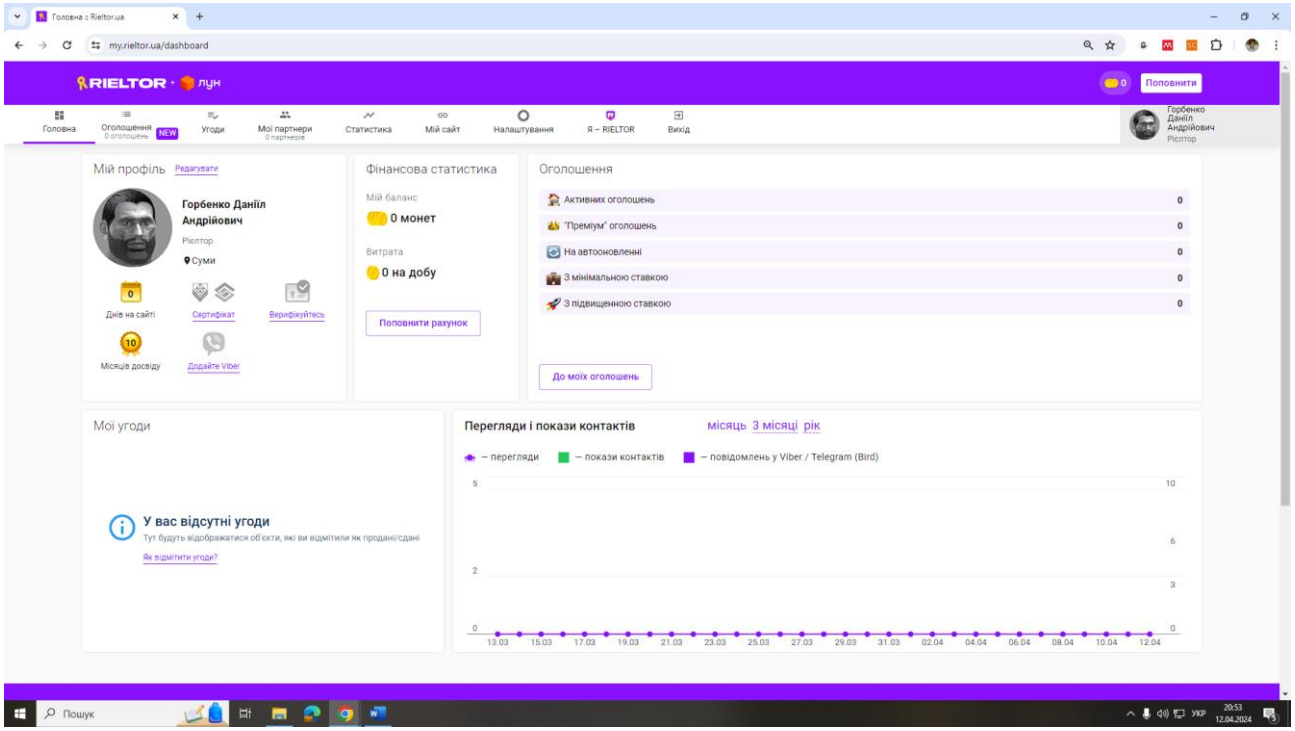


Рисунок 1.7 – Особистий кабінет рієлтора на сайті rieltor.ua

Для особистого кабінету існує окремий мобільний додаток, який повторює функціонал кабінету на сайті. Зовнішній вигляд мобільного додатку (рис. 1.8):

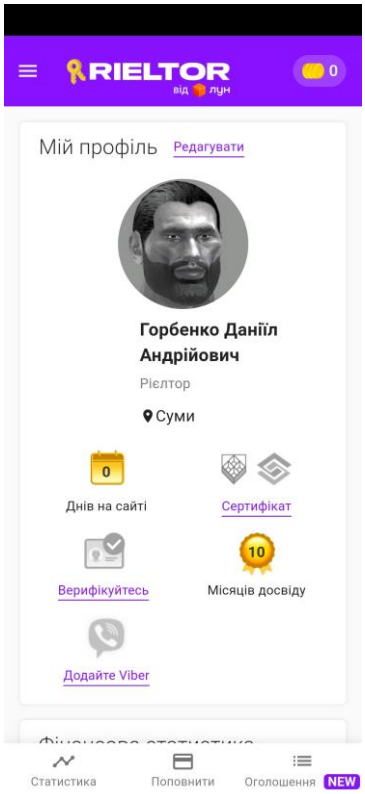


Рисунок 1.8 – Особистий кабінет рієлтора у мобільному додатку rieltor.ua

В особистому кабінеті можна прикріпити сертифікати рієлтора або агентства нерухомості, додати контакти з месенджера Viber для комунікації з клієнтами, переглядати статистику переглядів оголошень, контактів та повідомлень у Viber або месенджері Telegram. Рієлтор має можливість вести облік (рис. 1.9), опубліковувати оголошення щодо об’єктів нерухомості (рис. 1.10), а також закріпляти за ними продаж(оренду) за допомогою угод, у випадку заключення таких між клієнтами (рис. 1.11):

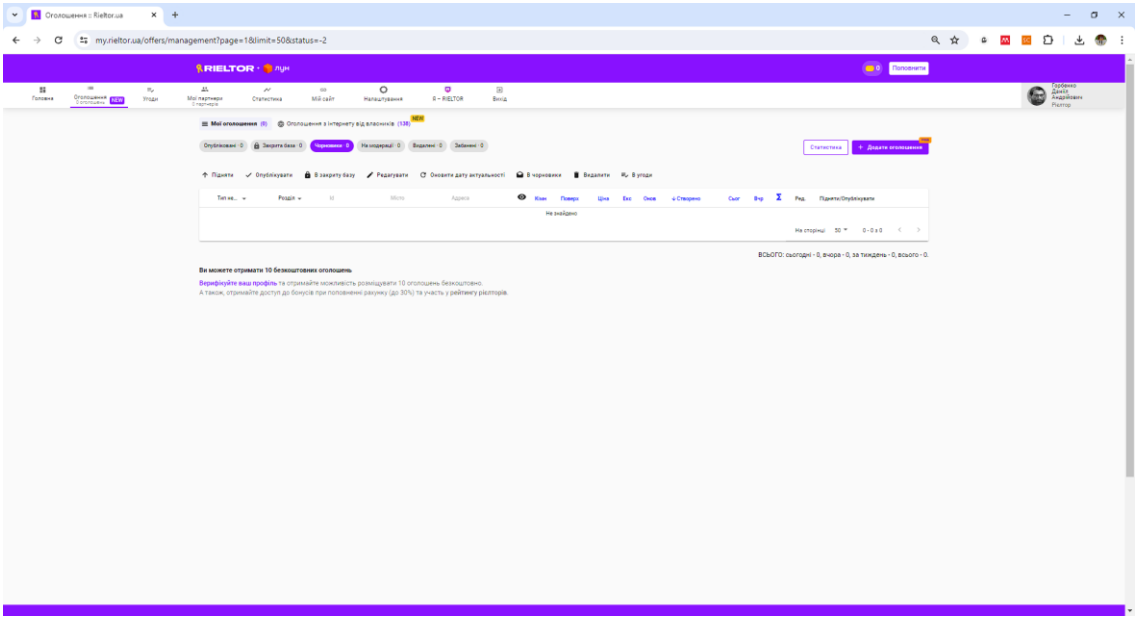


Рисунок 1.9 – Сторінка обліку оголошень на сайті rieltor.ua

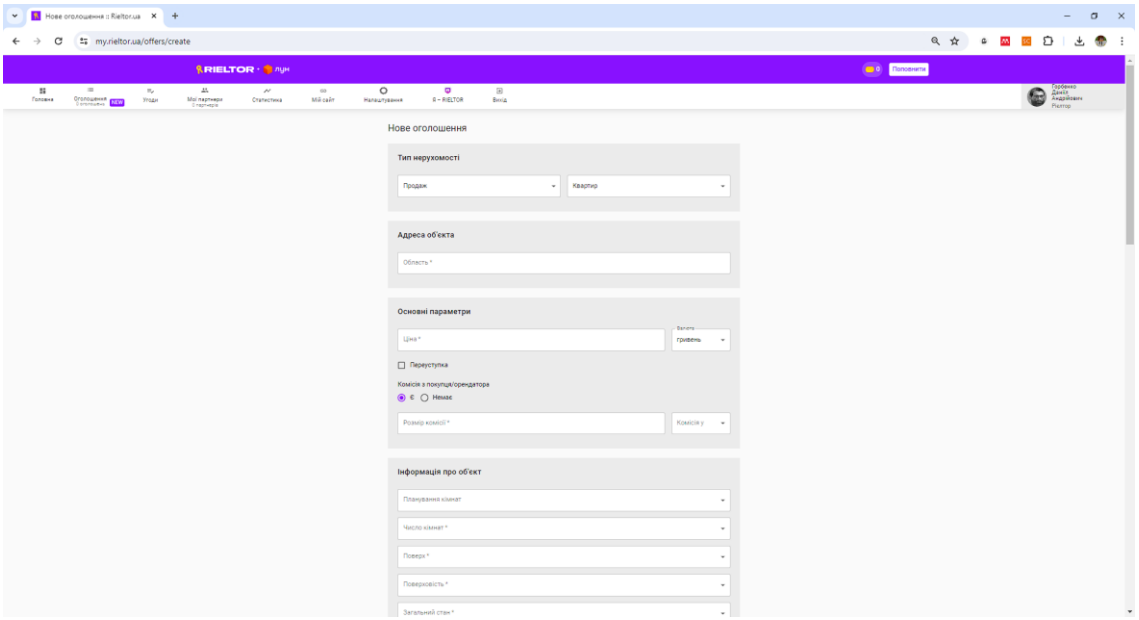


Рисунок 1.10 – Сторінка створення оголошень на сайті rieltor.ua

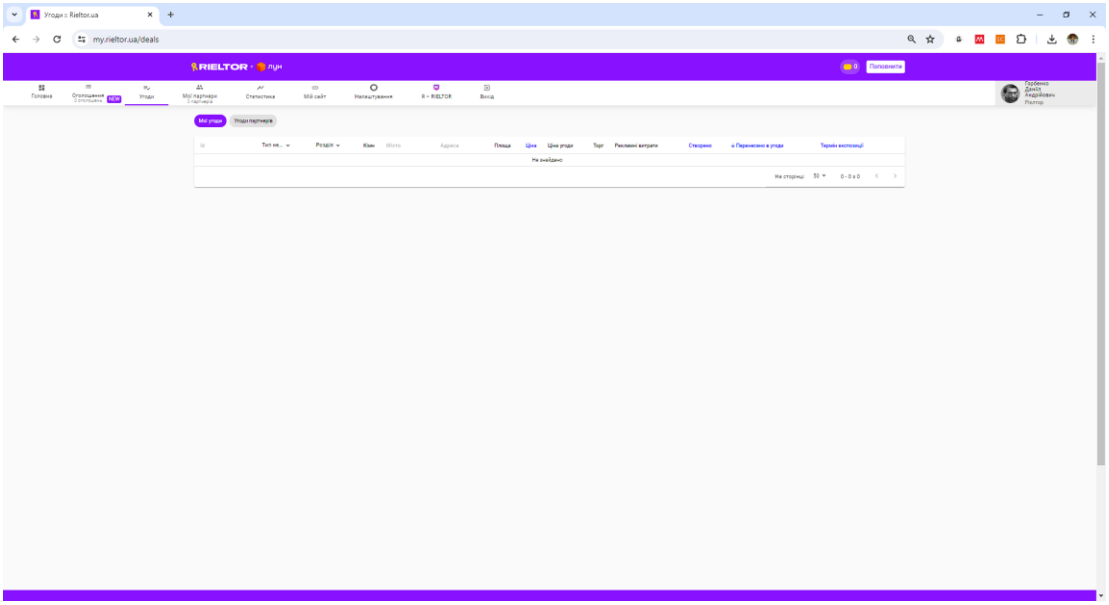


Рисунок 1.11 – Сторінка угод на сайті rieltor.ua

Розділ «Мої партнери» дозволяє створювати список рієлторів, між котрими організовується спільна робота над угодами та об’єктами, таким чином частково організовуючи роботу агентства нерухомості.

Аналогічний функціонал особистого кабінету доступний і у мобільному додатку Rieltor.ua(рис. 1.12–1.13 а–б):

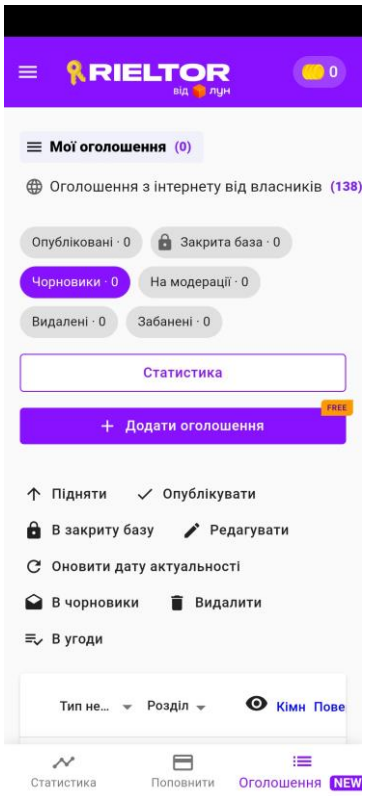
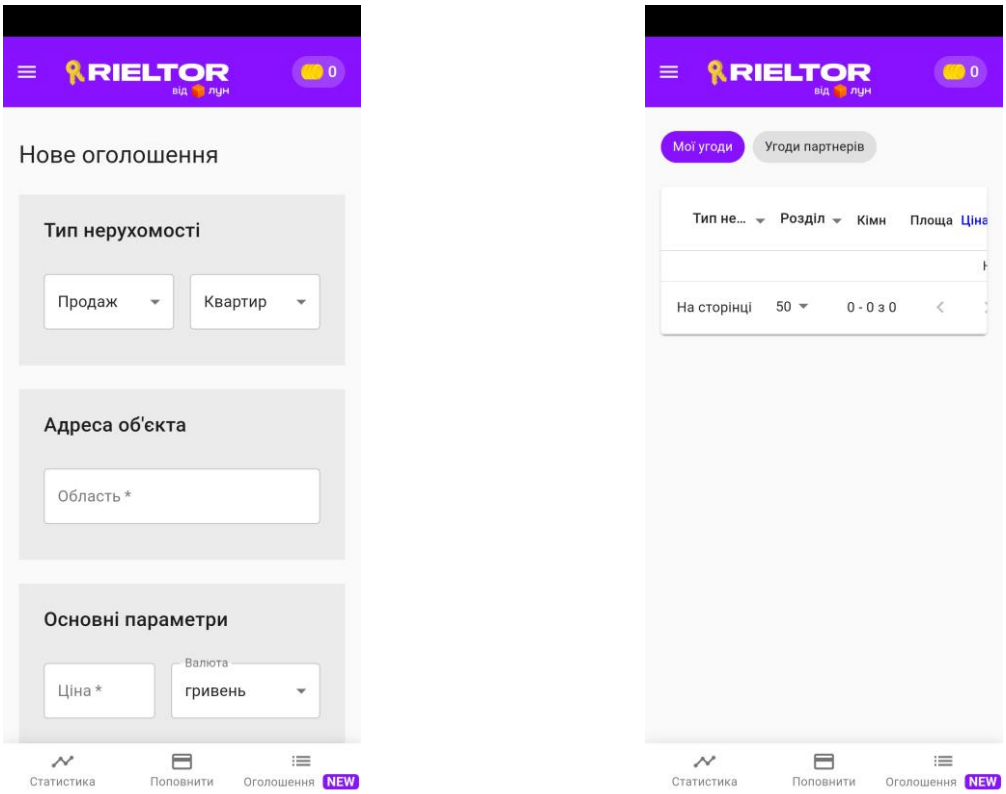


Рисунок 1.12 – Сторінка обліку оголошень мобільного додатку rieltor.ua

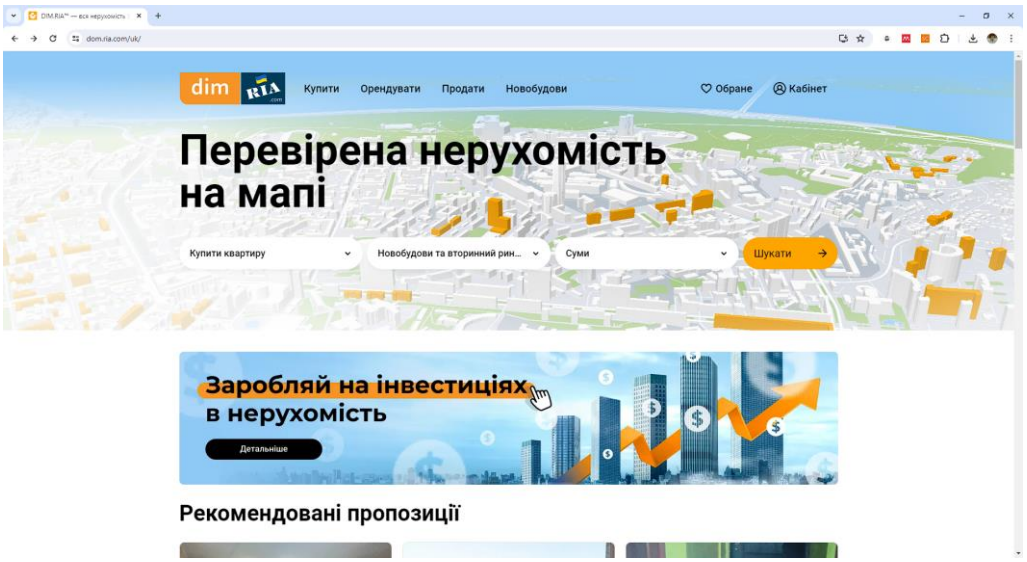


а б

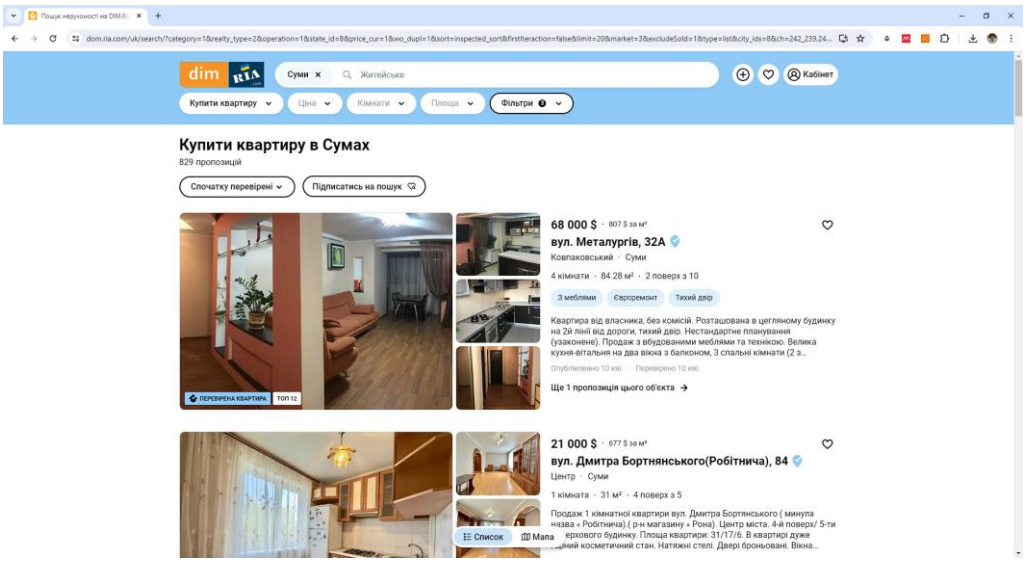
Рисунок 1.13 – Сторінки мобільного додатку rieltor.ua: створення оголошень (а), угод (б)

Dom.ria.com – web-ресурс для рієлторів, агентств нерухомості та людей, що шукають об’єкти нерухомості або хочуть їх продати. Зовнішній вигляд головної сторінки наведений на рисунку 1.14 а. Дозволяє здійснювати пошук об’єктів нерухомості, що виставлені на продаж або на оренду за відповідними оголошеннями (рис. 1.14 б-в), переглядати фотографії даних об’єктів, їх характеристики, зв’язуватися з особою, що виставила оголошення щодо об’єкта (рис. 1.15).

а



б



в

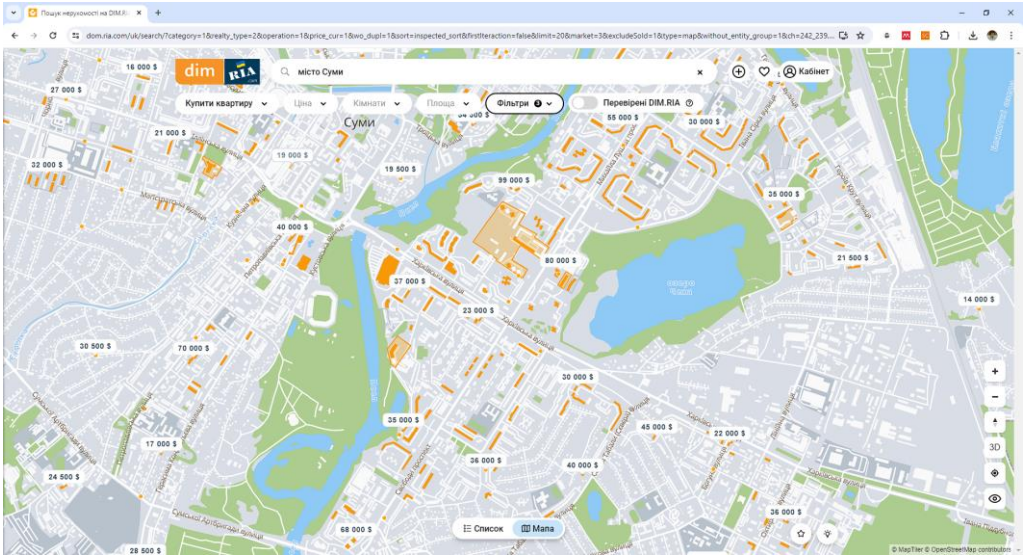


Рисунок 1.14 – Сторінки сайту dom.ria.com: головна (а), пошуку об’єктів (б), інтерактивної карти пошуку (в)

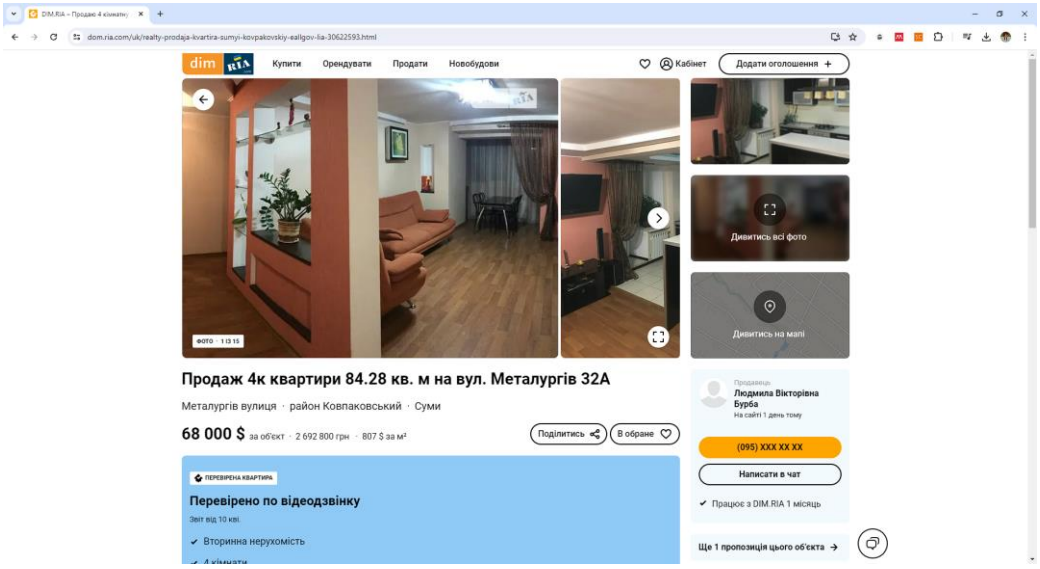


Рисунок 1.15 - Перегляд детальних характеристик об’єкта нерухомості на сайті dom.ria.com

На відміну від Rieltor.ua, допускає публікацію оголошень звичайними людьми, не тільки рієлторами. Для цього достатньо навести курсор на вкладку «Продати», натиснути кнопку додавання оголошення (рис. 1.16) та виконати верифікацію і дії за підказками (рис. 1.17):

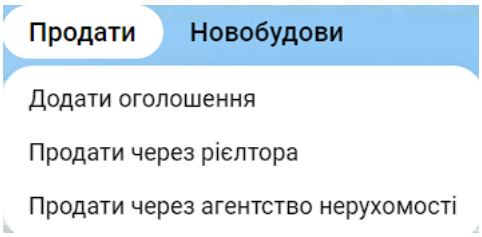


Рисунок 1.16 – Кнопка додавання оголошень на сайті dom.ria.com

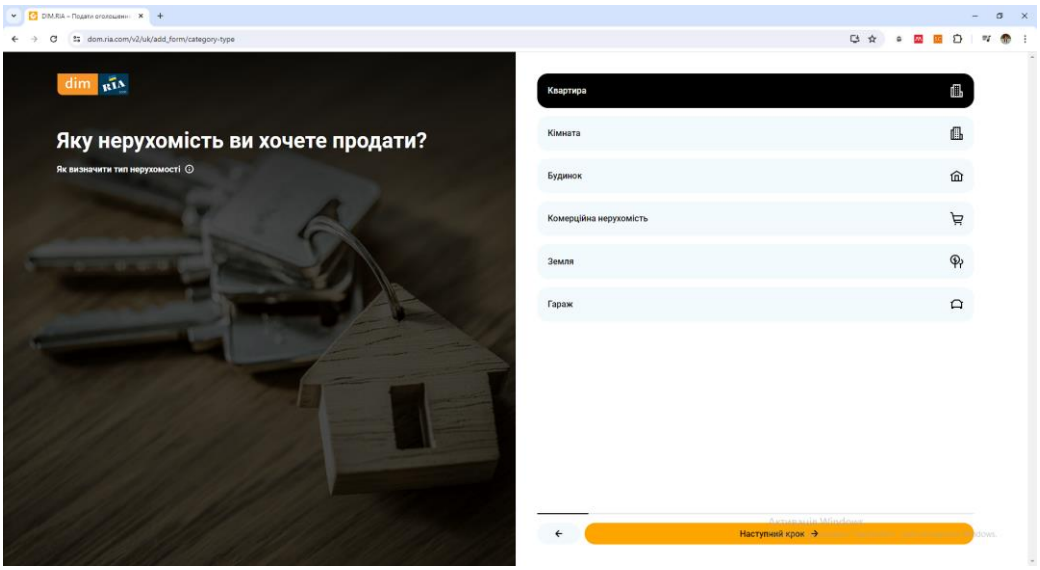


Рисунок 1.17 – Один з кроків додавання оголошення на сайті dom.ria.com

Має інструментарій для рієлторів та агентств нерухомості, схожий на аналогічний в Rieltor.ua. Вигляд особистого кабінету стандартного користувача (рис. 1.18):

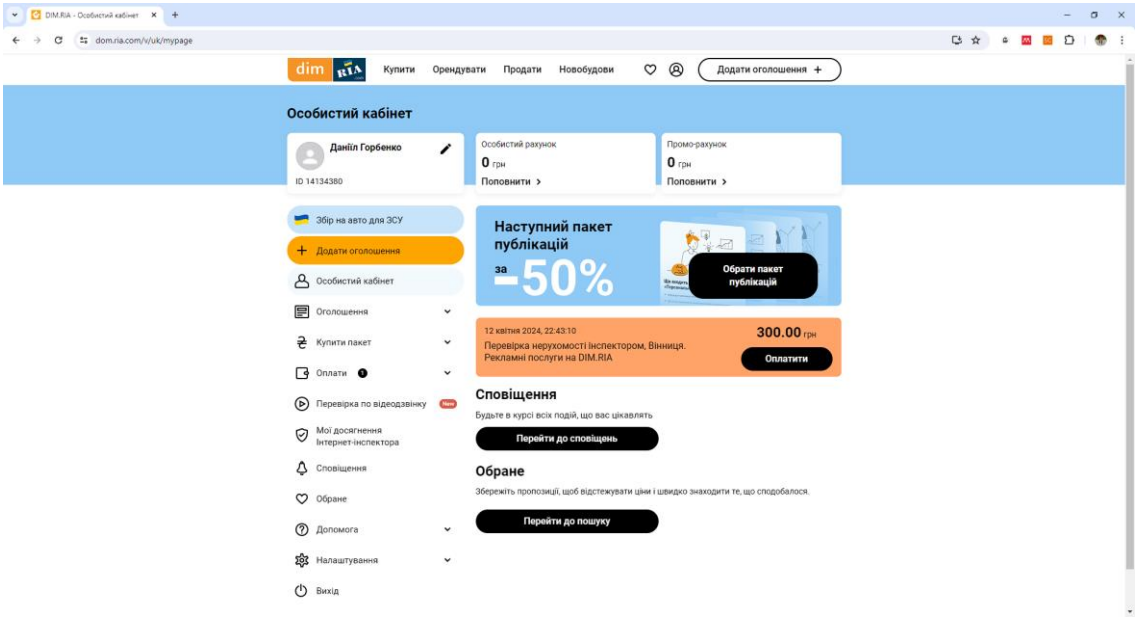


Рисунок 1.18 – Особистий кабінет користувача сайту dom.ria.com

Можна відмітити більш функціональний розділ управління оголошеннями (рис. 1.19). Можна вести облік опублікованих, архівних оголошень. У випадку вкладення угоди між двома клієнтами, активне оголошення можна перенести в архівне. Вкладка чернеток дозволяє заздалегідь підготувати шаблони оголошень для подальшої публікації. Графік перевірок дозволяє спланувати графік перевірок об’єктів нерухомості на предмет відповідності їх фізичного представлення з представленням в оголошенні. Перевірки можуть проводитися за відеодзвінком, відеозаписом або наживо з залученням експерта. Перевірки прямо впливають на просування оголошення, тривалість його перебування на сайті та необхідність сплачувати кошти за користування сайтом.

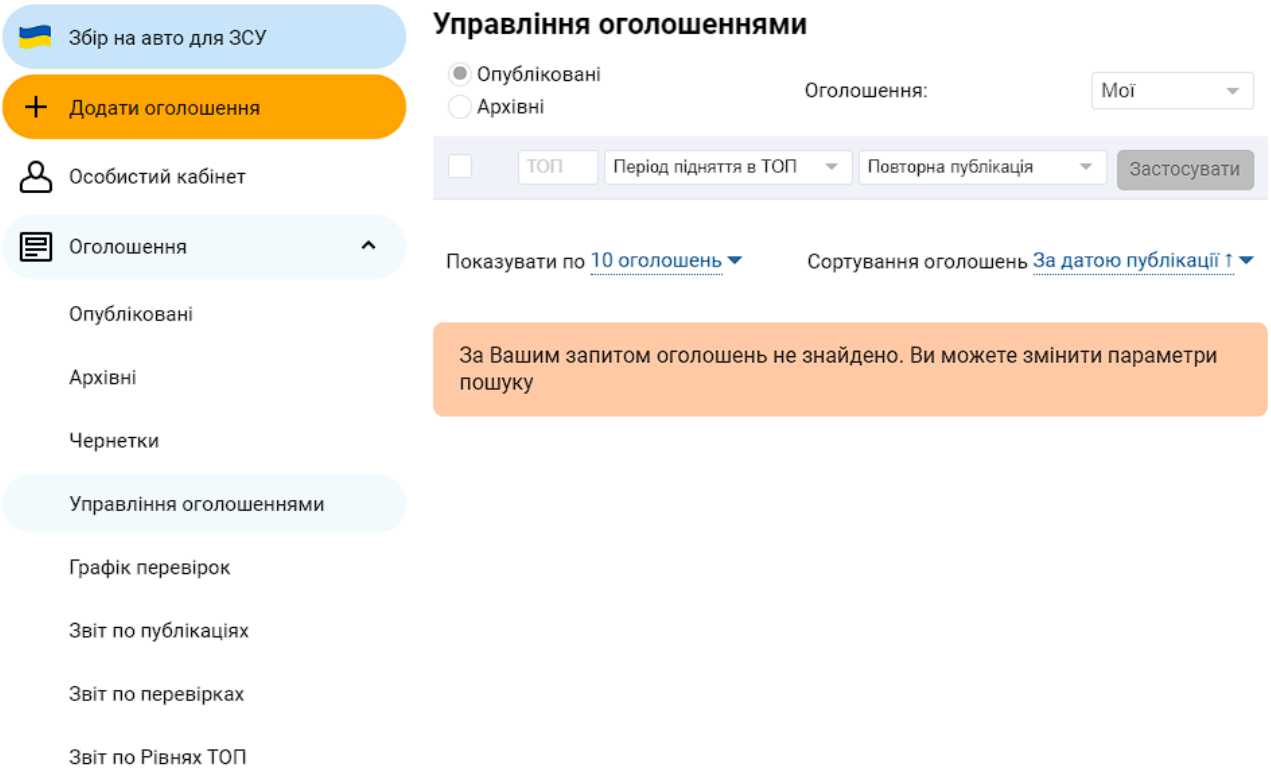


Рисунок 1.19 – Управління опублікованими оголошеннями на сайті dom.ria.com

Проте на даному етапі гарні сторони даного сайту для рієлторів та агентств нерухомості закінчуються. Одним з головних недоліків сайту dom.ria.com є необхідність сплачувати кошти за користування тими чи іншими послугами. Наприклад, рієлтори повинні сплачувати гроші за кожний перший дзвінок потенційного покупця, а агентства нерухомості повинні купляти спеціальні пакети, які мають певні обмеження щодо максимальної кількості можливих публікацій оголошень від агентства, кількість перших дзвінків від потенційних клієнтів, фіксації історії дзвінків у журналі, та деяких інших показників. Також, в цілому, реєстрація в якості рієлтора або агентства є платною. Перевірка об’єкта нерухомості експертом є платною, але дозволяє краще просувати оголошення. На рисунку 1.20 наведені деякі з таких спеціальних пакетів. На рисунку 1.21 наведена ілюстрація пакету «Дзвінки»:

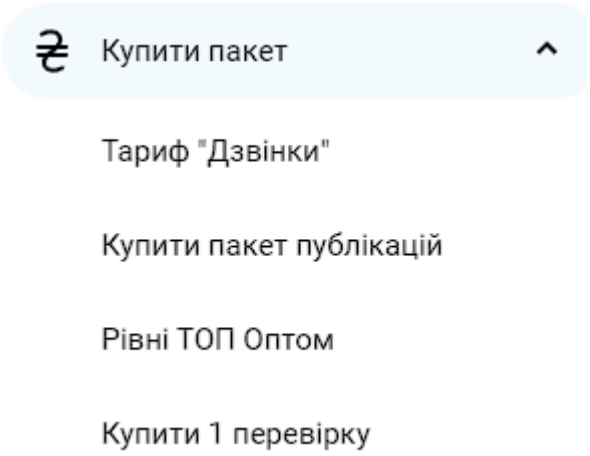


Рисунок 1.20 – Платні пакети послуг для рієлторів на сайті dom.ria.com

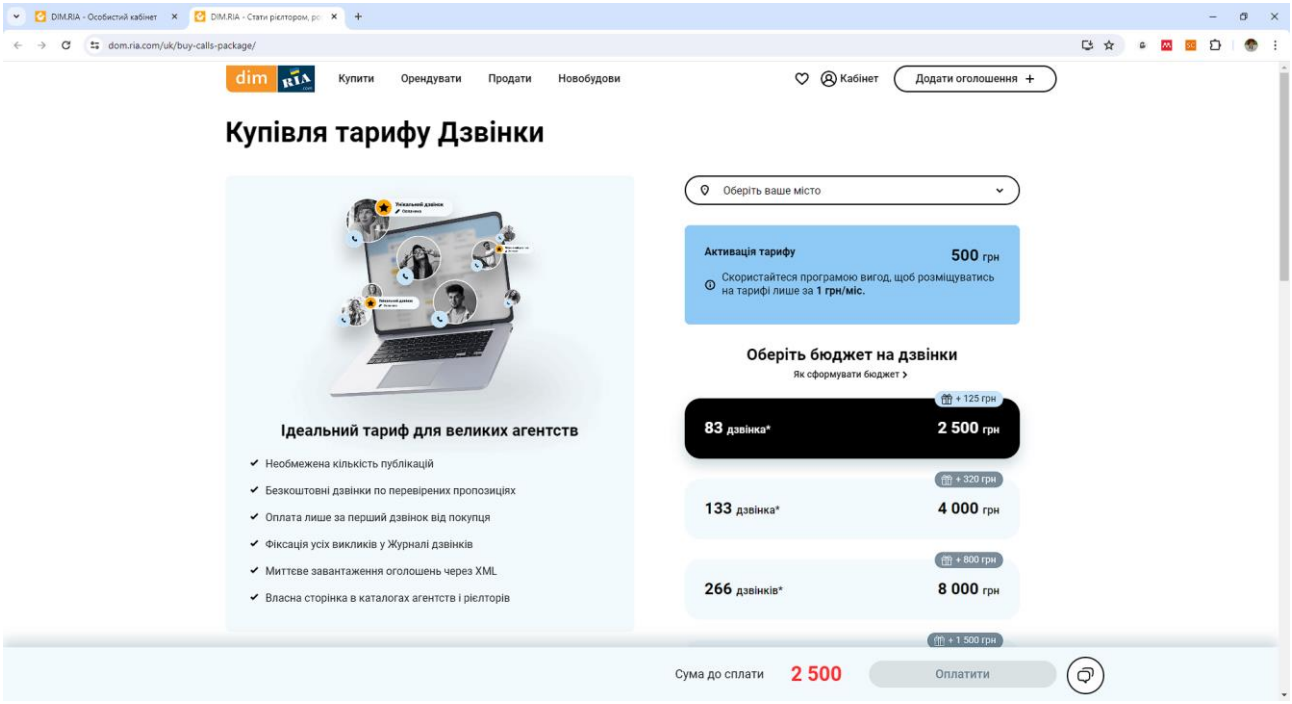
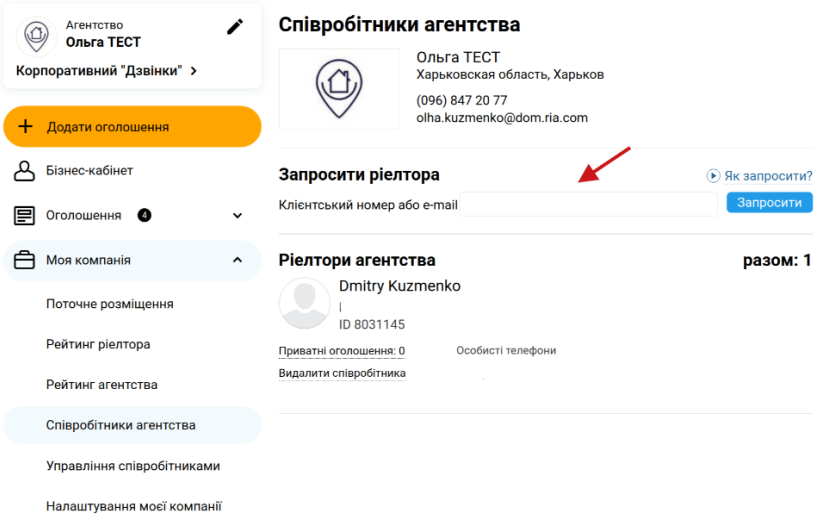


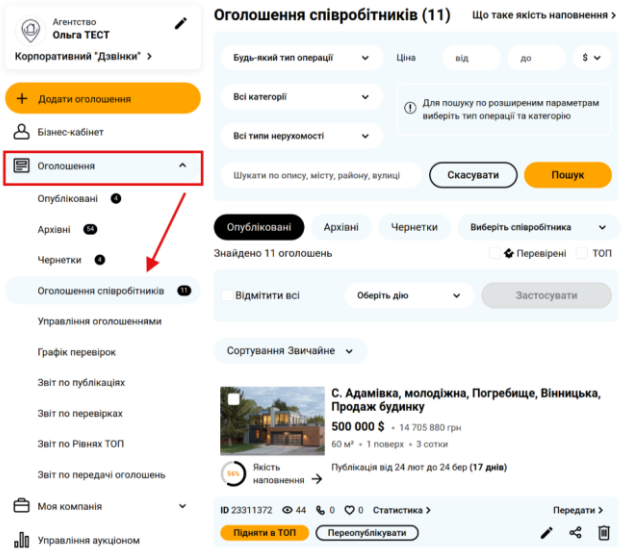
Рисунок 1.21 – Пакет «Дзвінки»

Адміністратор агентства нерухомості має можливість запрошувати до свого агентства рієлторів (рис. 1.22 а). Далі оголошення запрошених рієлторів вважаються запрошеннями агентства (рис. 1.22 б), і адміністратор має змогу передивлятися їх, передавати між рієлторами, виконувати операції просування, видалення в архів (рис. 1.22 в) [8,9]:

а



б



в

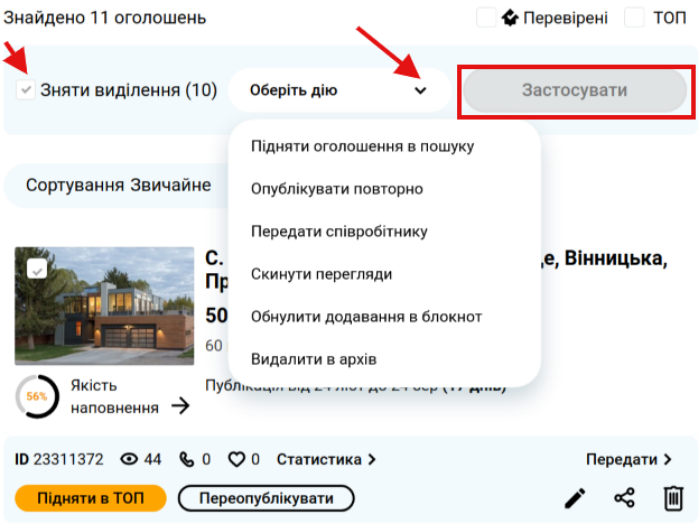
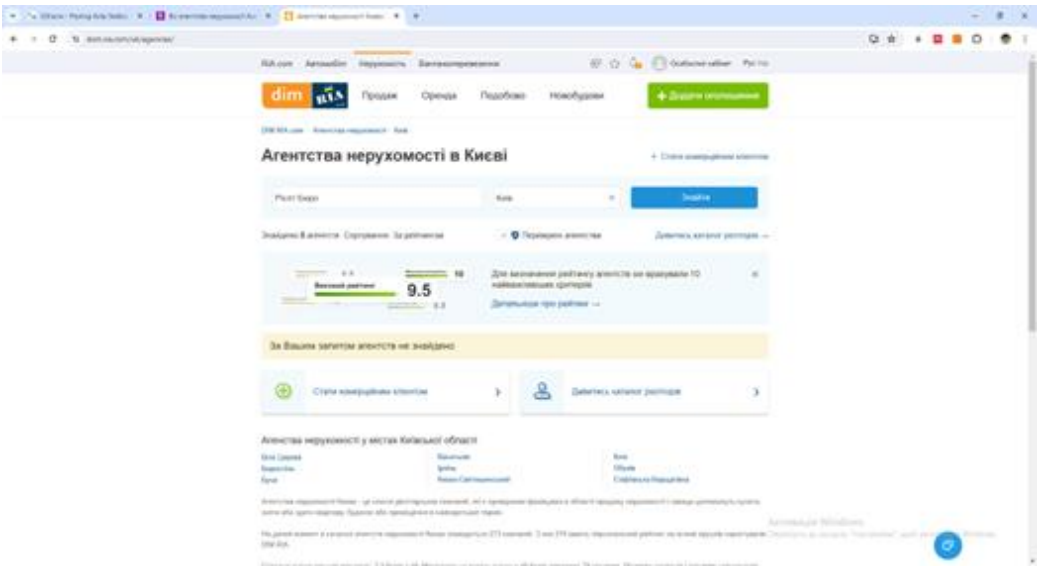


Рисунок 1.22 – Сторінки особистого кабінету агентства dom.ria.com: запрошення рієлторів (а), перегляду оголошень співробітників (б), меню операцій до оголошень (в)

Крім загальнодоступних систем, варто відзначити агентства нерухомості, котрі використовують власні системи(способи) ведення обліку об’єктів, оголошень. В результаті пошуку в мережі інтернет було знайдено веб-ресурс(**zlat.com.ua**) приватного агентства «Рієлт Бюро» одного з рієлторів міста Київ. Дане агентство не було знайдено на загальнодоступних системах dom.ria.com та rieltor.ua (рис. 1.23), а отже агентство з високою ймовірністю веде діяльність самостійно. На сайті агентства неозброєним оком можна побачити розділ «Об’єкти», де знаходиться база об’єктів нерухомості, котрі на даний момент продаються за посередництвом даного агентства (рис. 1.24):

а



б

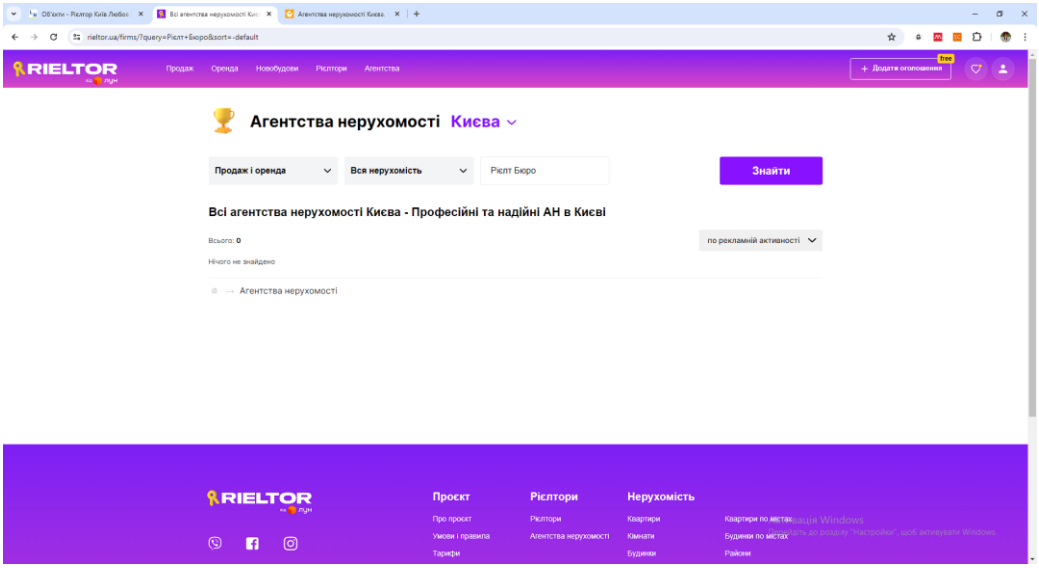


Рисунок 1.23 – Відсутність агентства «Рієлт Бюро» на сайті dom.ria.com (а) та rieltor.ua (б)

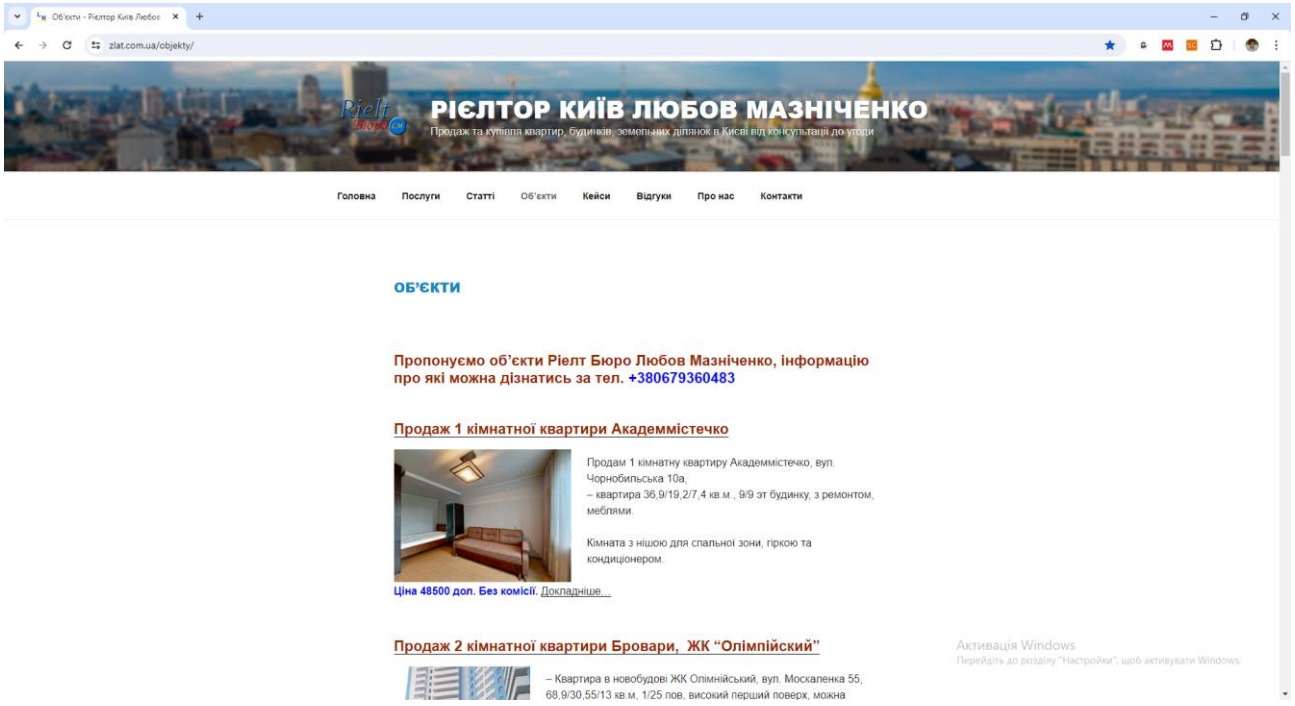
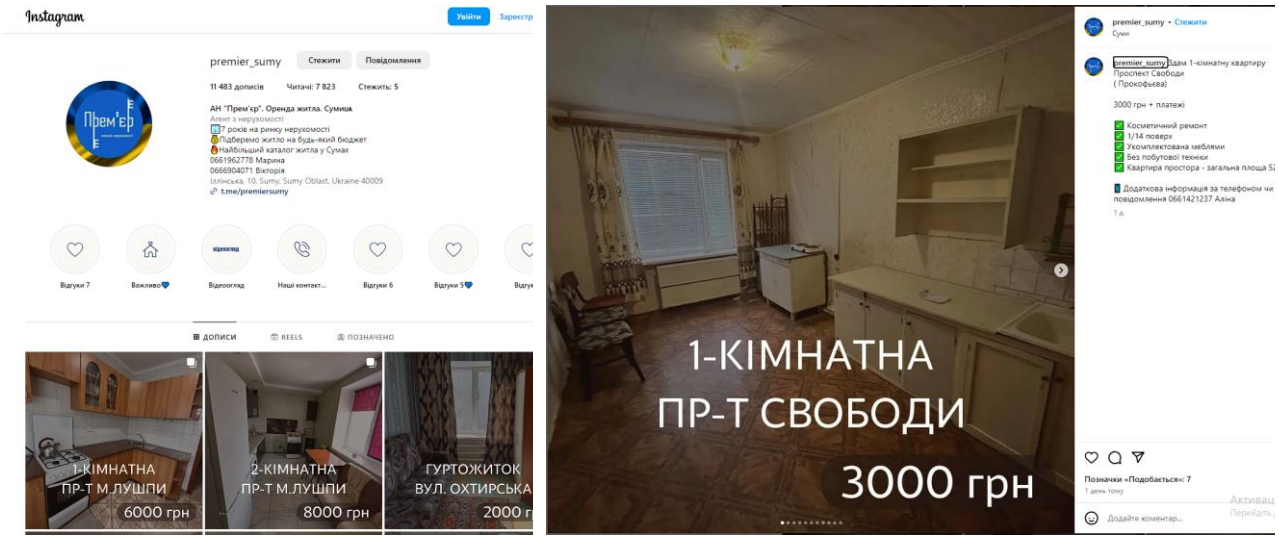


Рисунок 1.24 – Розділ «Об’єкти» сайту агентства «Рієлт Бюро»

Існують агентства, що ведуть свою діяльність у соціальних мережах. Прикладом може слугувати агентство нерухомості «Прем’єр» з міста Суми. Агентство використовує інструментарій соціальної мережі **Instagram**(рис. 1.25 а), маючи акаунт, у якому в якості звичайних постів публікуються фотографії об’єктів з дописами щодо їх характеристик та ціни (рис. 1.25 б). Якщо об’єкт продається або здається в оренду, пост редагується і помічається відповідними словами. В якості «stories»(короткочасні відео до однієї хвилини довжиною) публікуються відеоогляди об’єктів та відгуки клієнтів (рис 1.26). Агентство залишає контакти в описі профілю та під кожним постом, таким чином роблячи можливою комунікацію з клієнтами.



а б

Рисунок 1.25 – Сторінка агентства «Прем’єр» у соціальній мережі Instagram: загальний вигляд (а), пост щодо об’єкта нерухомості (б)

Рисунок 1.26 – Доступні «stories» на сторінці агентства «Прем’єр»

Виокремимо деякі характеристики оглянутих веб-ресурсів/мобільних додатків й складемо порівняльну таблицю результатів аналізу (табл. 1.2):

Таблиця 1.2 – Порівняльна таблиця результатів аналізу додатків-аналогів

Характеристика	rieltor.ua	dom.ria.com	zlat.com.ua	Сторінка агентства нерухомості «Прем’єр» в Instagram
Зручний та зрозумілий інтерфейс	+	+	+	+
Чітка спрямованість на сферу нерухомості	+	+	+	+

Продовження таблиці 1.2

Наявність функціоналу для продажу/оренди об’єктів через рієлторів/агентства нерухомості	+	+	-	-
Наявність функціоналу для ведення рієлторами обліку об’єктів(оголошень)	+	+	+	+
Наявність функціоналу для ведення рієлторами обліку завершених угод	+	+	невідомо	+/-
Наявність функціоналу для ведення рієлторами обліку клієнтів	-	-	невідомо	-
Наявність функціоналу для комунікації між рієлторами та клієнтами за допомогою сторонніх ресурсів	+	+	+	+
Наявність функціоналу для комунікації між рієлторами та клієнтами за допомогою власних ресурсів сайту/додатку	-	+	-	+
Наявність функціоналу для адміністрування агентства нерухомості	+	+	-	-
Відсутність оплати за користування ресурсом або за окремі послуги	+	-	+	+

1.3 Аналіз засобів вирішення проблеми

1.3.1 Мови програмування

Java – одна з найбільш популярних мов програмування. Java створена за зразком традиційних мов програмування C/C++ і є досить легкою для опанування за умови попереднього вивчення програмістом мов програмування

C/C++ [10]. Java є повністю об’єктно-орієнтованою мовою програмування, а також платформно-незалежною, тобто може працювати на будь-яких комп’ютерних пристроях, котрі підтримують JVM(Java Virtual Machine, віртуальна машина Java) [10]. Підтримує «збиральник сміття». Платформо-незалежність Java робить її популярною у розробці програм для мобільних пристроїв, особливо під Android. Може використовуватися як для back-end частин додатків, так і для front-end. Використання JVM дає змогу транслювати код Java у байткод JVM, і вже потім у машинний код. Для кожної системи JVM дещо різниться в зв’язку з особливостями системи, але дає змогу запускати один і той самий код як і на Windows, так і на Linux, Mac OS, Android, тощо. Це дає змогу досягти платформо-незалежного статусу Java.

Проте, з появою мови програмування **Kotlin** як офіційної мови програмування під Android, доля Java на ринку мов програмувань для мобільних пристроїв почала зменшуватися. Згідно з даними за 2023 рік від порталу StackOverflow [11] щодо використання мов програмування програмістами, серед усіх мов програмування в усіх сферах Java тримає частку 30.55%, Kotlin – 9.06% (рис. 1.27). Проте, у сфері конкретно мобільних пристроїв, в даному випадку пристроїв з операційною системою Android, Kotlin поступово набирає оберти та за даними від порталу K&C [12] вже на кінець 2022 року тримав за собою частку 51.28% серед усіх інших мов програмування, на котрих були написані найновіші додатки. Крім того, частка написаних на Kotlin додатків, що знаходяться у топ-500 найкращих додатків у Сполучених Штатах Америки станом на кінець 2022 року, становить домінуючі 86.87%. Загальна частка Kotlin серед усіх мов програмування під Android становить 21.8% на кінець 2022 року (рис. 1.28), що, враховуючи молодість даної мови програмування та величезну кількість старших додатків, написаних ще до появи Kotlin, є доволі високим показником.

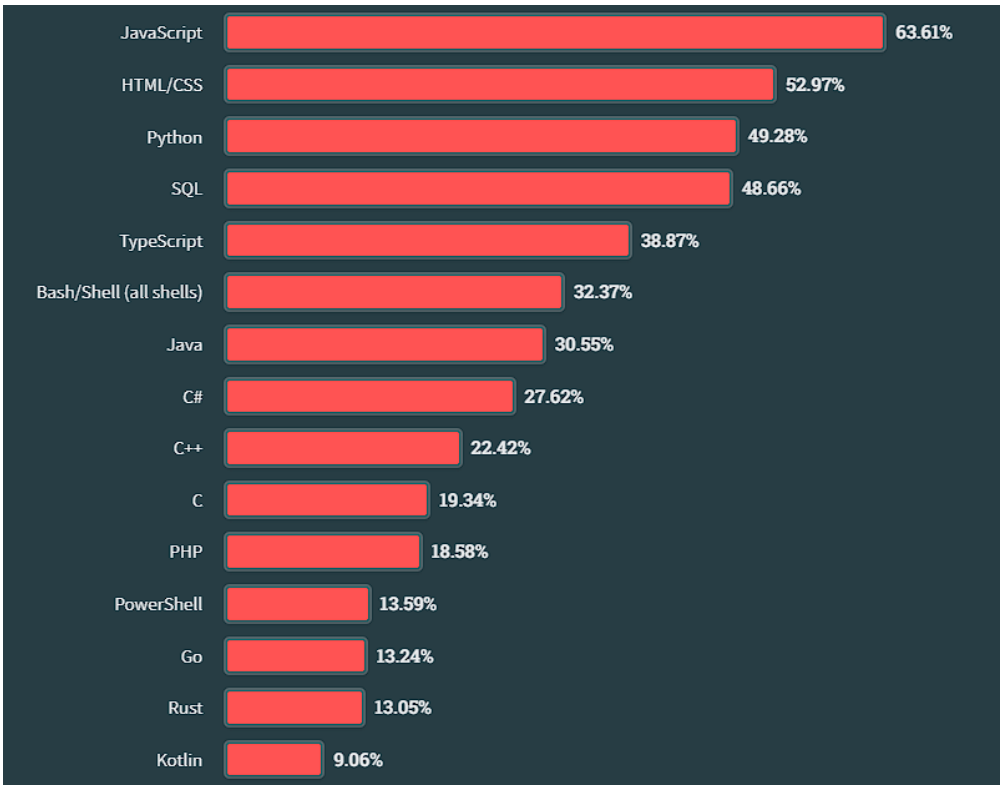


Рисунок 1.27 – Статистика використання мов програмування за 2023 рік

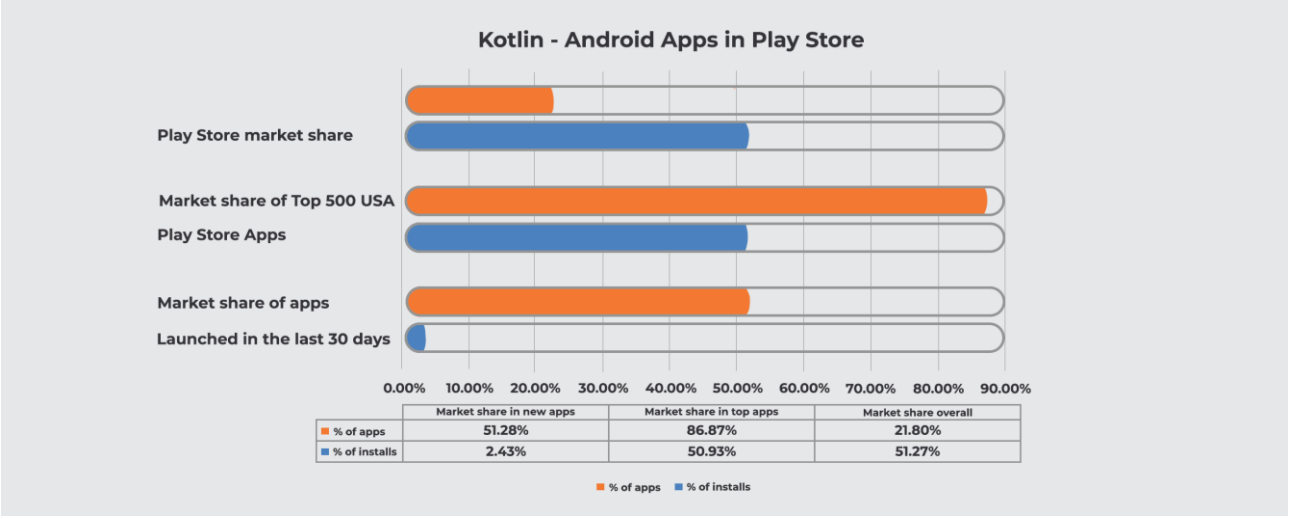


Рисунок 1.28 – Статистика використання мови програмування Kotlin серед Android-додатків за 2022 рік

Kotlin – одна з найновіших мов програмування, розроблювана компанією JetBrains. Розробники мови ставили перед собою мету створити більш лаконічну мову з більшою безпекою типів даних, ніж Java, та простішу, ніж мова програмування Scala. В результаті вийшла статично типізована мова програмування, що працює поверх Java Virtual Machine, а отже є сумісною з Java [13]. Kotlin

вважається більш readable та простішою мовою програмування, ніж її «натхненник» Java. Крім спрощеного синтаксису, однією з найцікавіших відмінностей Kotlin є можливість оголошувати статичні методи не тільки всередині тіл класів, а й поза ними на верхньому рівні пакетів [13]. Згідно з статистичними даними за 2022 та 2023 роки (див. рис. 1.27–1.28), Kotlin поступово замінює Java як основну мову програмування під операційну систему Android. Також, у 2019 році Kotlin було визнано рекомендованою мовою програмування для застосунків на Android компанією Google, котра є розробником Android.

C# - одна з найпопулярніших мов програмування, що постійно прогресує. C# є строго статично типізованою, об'єктно-орієнтованою, високорівневою мовою програмування. Була створена у 2002 році компанією Microsoft як відповідь на популярну тоді мову програмування Java. Таким чином, C# поєднав чітку об'єктно-орієнтовану структуру Java з старими елементами мов програмування роду C/C++, прикладом тому є наявність можливості перевантаження операторів, або ж наявність збиральника сміття. C# посів вагоме місце у сфері розробки веб-сайтів, хмарних систем, IoT пристроїв, машинного навчання, десктопних програм, ігровій сфері, утиліт командного рядка, і також у сфері мобільних додатків [14]. Багато в чому успіх C# вдячний платформі .NET. Платформа .NET дозволяє C# бути платформо-незалежним та працювати на більшості сучасних операційних систем, включаючи Windows, Linux та похідні від них. Платформа .NET містить у собі runtime (CLR, Common Language Runtime) та головні бібліотеки-класи, що використовують програми на C#. CLR містить у собі CTS (Common Type System, система спільних типів), котра дозволяє всім мовам програмування платформи .NET використовувати бібліотеки інших мов платформи .NET. Тобто, наприклад, бібліотеки, написані на Visual Basic, можуть використовуватися у C#, бібліотеки C# - у F#. Крім бібліотек, .NET містить вбудовані фреймворки для створення веб-додатків, веб-API, десктопних та мобільних додатків, а також підтримує багато сторонніх фреймворків [14]. Серед таких фреймворків можуть бути, наприклад, ASP.NET для створення веб-додатків , Entity Framework

Core для роботи з базами даних, тощо. За статистикою на кінець 2023 року (див. рис. 1.27), C# широко використовується і «дихає у спину» своєму головному конкуренту, Java, тримаючи показник 27.62% використання серед усіх мов програмування.

1.3.2 Бази даних

MySQL – одна з найбільш популярних середовищ керування базами даних. MySQL є реляційною базою даних з відкритим кодом. Серед усіх додатків, що використовують бази даних, MySQL посідає друге місце за частотою використання, поступаючись Oracle (рис. 1.29) [15]. Серед найновіших додатків, за даними сайту StackOverflow за 2023 рік, MySQL також є другою базою даних, незначно поступаючись PostgreSQL (рис. 1.30) [11]. Серед переваг MySQL:

- зручність використання;
- надійність;
- масштабованість;
- продуктивність;
- висока доступність;
- безпека;
- гнучкість. [16]

Oracle – одна з найстаріших та найвідоміших реляційних баз даних, що використовують мову SQL. Серед усіх додатків, що використовують бази даних, Oracle Database посідає перше місце (рис. 1.29) [15]. Проте, якщо брати найновіші розробки, Oracle поступово відходить на другий план, значно поступаючись PostgreSQL та MySQL (рис. 1.30) [11]. Oracle є чи не найшвидшою СКБД на даний момент. Вона відома високою швидкістю у роботі з великим об’ємом даних.

PostgreSQL – нині популярна реляційна база даних, що також використовує мову SQL. Серед усіх додатків, що використовують бази даних, PostgreSQL посідає четверте місце (рис. 1.29) [15]. У найновіших додатках картина вже виглядає інакше – PostgreSQL є лідером за даними StackOverflow (рис. 1.30) [11].

PostgreSQL відома найбільшою кількістю типів даних, проте в деяких випадках може бути повільнішою за конкурентів MySQL та Oracle.

Rank			DBMS	Database Model	Score		
Apr 2024	Mar 2024	Apr 2023			Apr 2024	Mar 2024	Apr 2023
1.	1.	1.	Oracle	Relational, Multi-model	1234.27	+13.21	+5.99
2.	2.	2.	MySQL	Relational, Multi-model	1087.72	-13.77	-70.06
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	829.80	-16.01	-88.73
4.	4.	4.	PostgreSQL	Relational, Multi-model	645.05	+10.15	+36.64
5.	5.	5.	MongoDB	Document, Multi-model	423.96	-0.57	-17.93
6.	6.	6.	Redis	Key-value, Multi-model	156.44	-0.56	-17.11
7.	7.	↑ 8.	Elasticsearch	Search engine, Multi-model	134.78	-0.01	-6.29
8.	8.	↓ 7.	IBM Db2	Relational, Multi-model	127.49	-0.26	-18.00
9.	9.	↑ 12.	Snowflake	Relational	123.20	-2.18	+12.07
10.	10.	↓ 9.	SQLite	Relational	116.01	-2.15	-18.53

Рисунок 1.29 – Статистика використання баз даних за увесь час від порталу db-engines.com

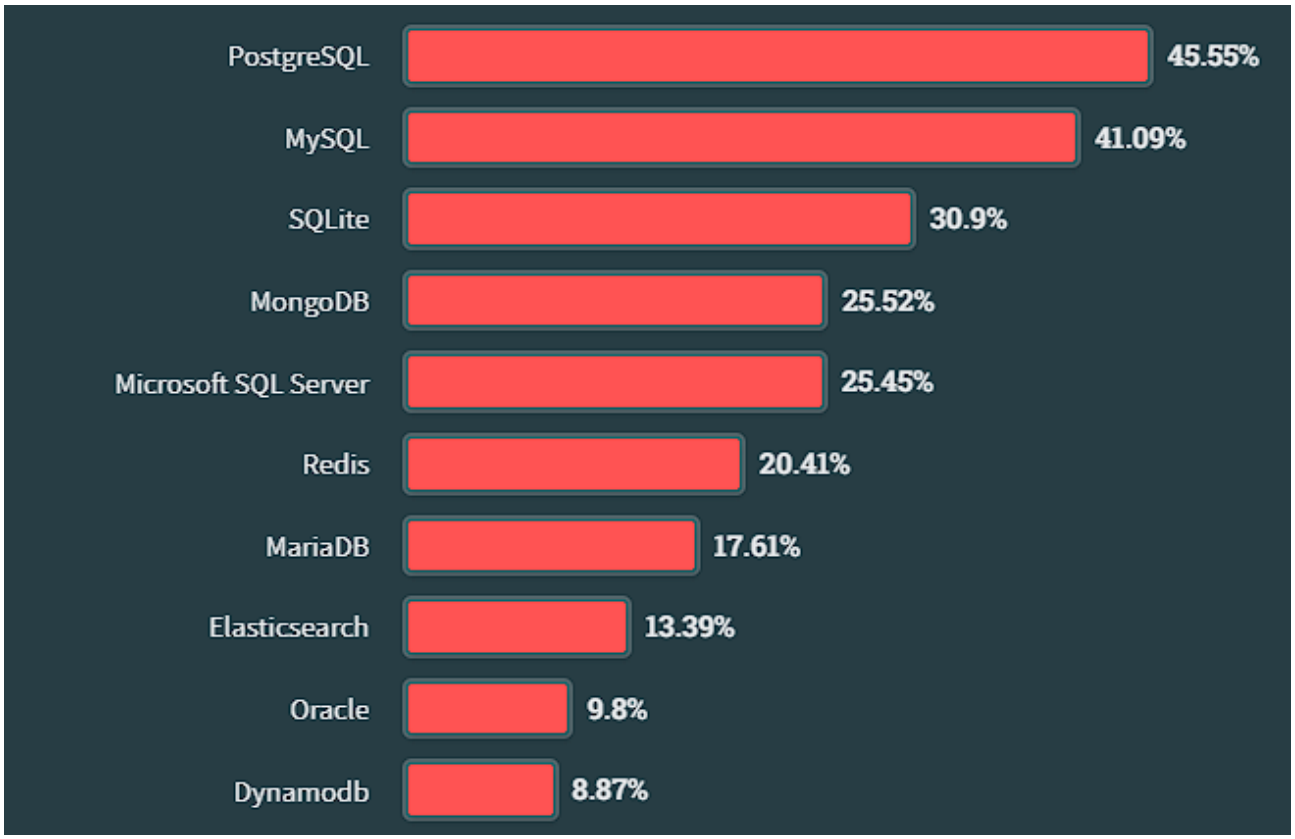


Рисунок 1.30 – Статистика використання баз даних за 2023 рік від порталу StackOverflow

1.3.3 Інтегровані середовища розробки

Програмна реалізація більшості додатків відбувається у спеціальних програмах, що називаються IDE (Integrated Development Environment, інтегроване

середовище розробки). Такі програми мають усі інструменти для зручного написання коду, підсвітки синтаксису мов програмування, виконання та дебаггингу коду, будування графічного інтерфейсу, тощо. Більшість інтегрованих середовищ розроби призначені для конкретних мов програмування (наприклад, IntelliJ IDEA для Java, PyCharm для Python, Dev C++ для C++), хоча є й універсальні (Visual Studio для C, C++, C#, F#, JavaScript, TypeScript, Visual Basic та багатьох інших мов завдяки розширенням).

Для розробки під платформу Android можна виділити наступні інтегровані середовища розробки:

– **Android Studio**. Найвідоміше, найзручніше та найпопулярніше середовище розробки додатків під платформу Android. Створена на основі IntelliJ IDEA – найбільш популярного інтегрованого середовища розробки для мови програмування Java. Дозволяє розробляти додатки на мовах програмування Java, Kotlin та C++. Створення зовнішнього вигляду інтерфейсу відбувається за допомогою XML-розмітки. Дозволяє виконувати написаний код, запускаючи його як додаток на віртуальному пристрої, що вбудований у середовище, що дозволяє побачити, як виглядатиме та працюватиме додаток, не встановлюючи його на фізичний мобільний пристрій. Створене спеціально для програмування на Android, тому має усі необхідні функції і нічого зайвого для такої розробки, що робить її дуже зручною. Також, крім мобільних пристроїв, має інструментарій для розробки під телевізори та навіть у автомобільні бортові комп’ютери. Ілюстрація вигляду Android Studio наведена на рисунку 1.31:

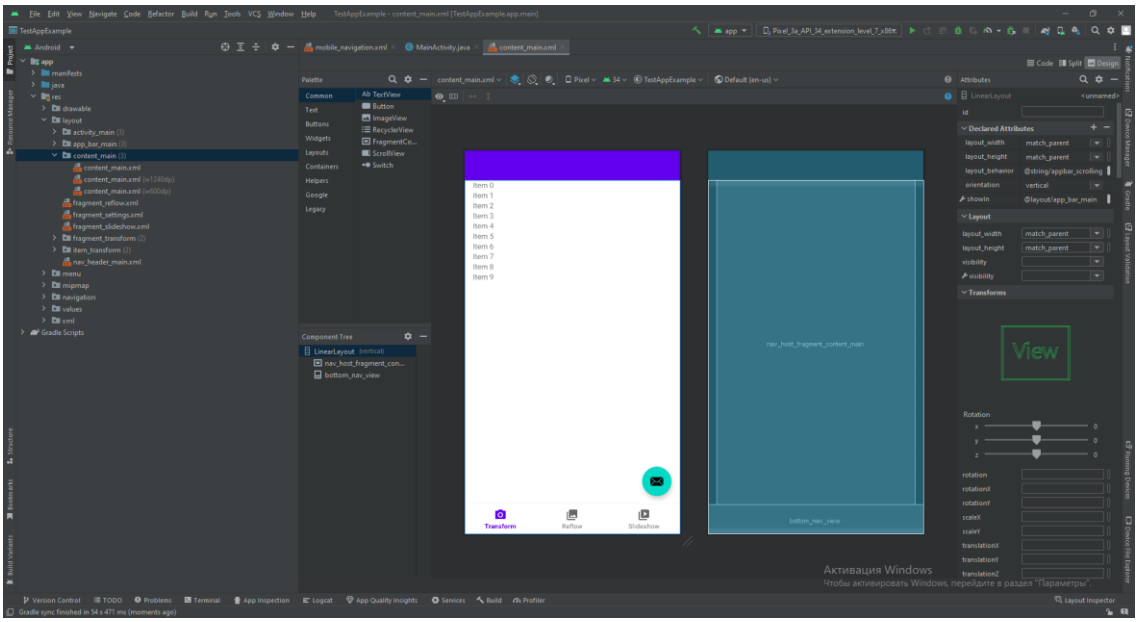


Рисунок 1.31 – Вікно середовища розробки Android Studio

– **Microsoft Visual Studio**. Також дуже популярне середовище розробки мобільних додатків. За популярністю у розробці під Android дещо поступається Android Studio, але є ведучою у розробці під iOS (операційна система). Цю можливість має завдяки платформі Xamarin від однойменної компанії, що постачається у вигляді розширення. Дозволяє розробляти додатки під Android та iOS на мові програмування C#. Також, як і Android Studio, має можливість запуску додатку на віртуальному мобільному пристрої, що вбудований в інтегроване середовище розробки. Крім розробки мобільних додатків, Visual Studio широко використовується у традиційній розробці програм, в основному на мовах програмування роду C (C, C++, C#). Є основним інтегрованим середовищем розробки під платформу .NET, а також широко використовується в ігровій індустрії. Ілюстрація вигляду Visual Studio наведена на рисунку 1.32:

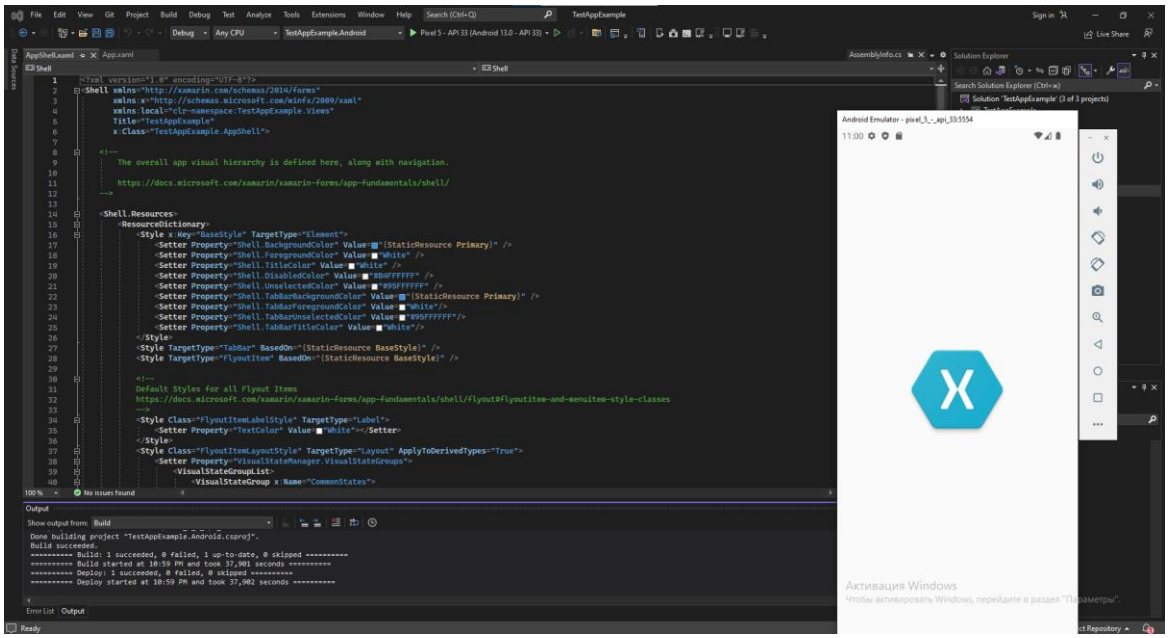


Рисунок 1.32 – Вікно середовища розробки Visual Studio

Також можна виділити редактор коду **Visual Studio Code**. Даний редактор коду має дуже потужний набір розширень, які роблять написання коду ще легшим, ніж у будь-якому повноцінному середовищі розробки. Також підтримує чи не всі наявні у світі мови програмування (якщо брати до уваги ті, що не використовуються для Android-розробки). Але, так як це є редактор коду, немає інструментарію для запуску додатків, їх дебаггингу, та інших речей, пов’язаних з розробкою (варто відзначити, що деякі розширення дозволяють проводити такі дії) – тільки написання та редагування коду. Ілюстрація вигляду Visual Studio Code наведена на рисунку 1.33:

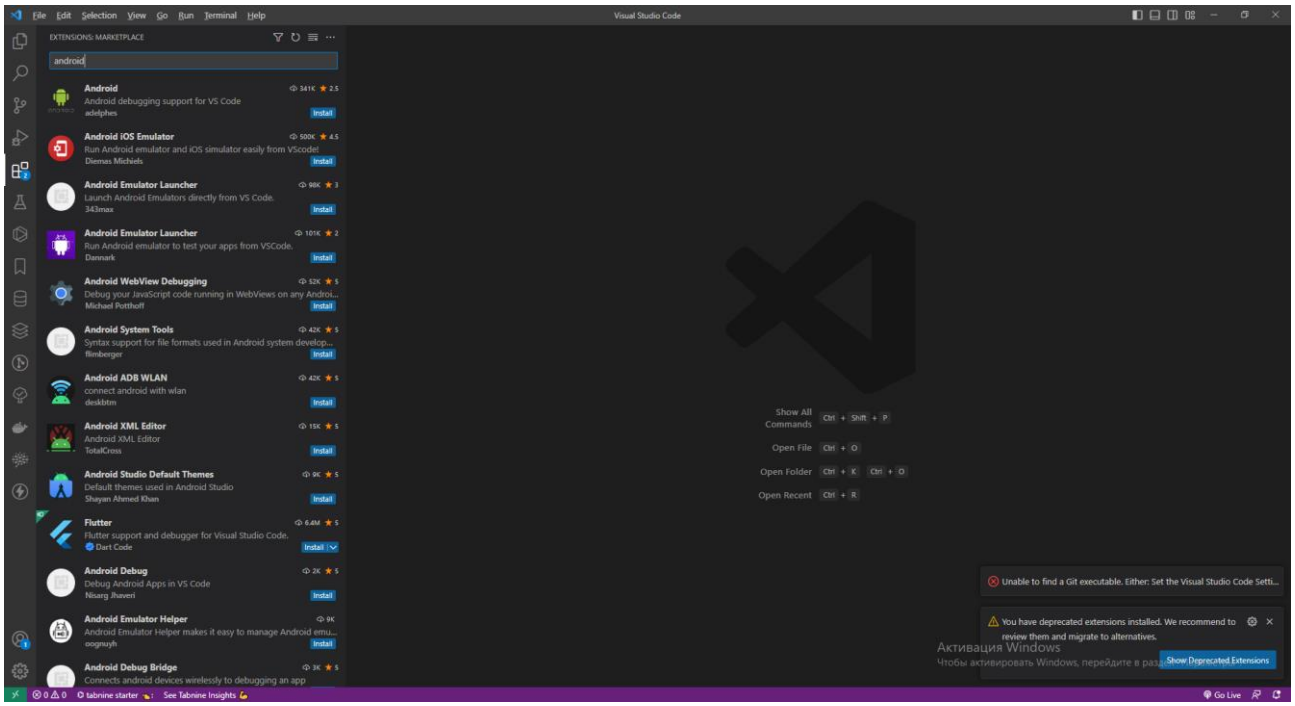


Рисунок 1.33 – Вікно редактора коду Visual Studio Code

1.4 Постановка задачі

Враховуючи дослідження, проведене у минулих пунктах (дивитися пункти 1.1, 1.2), агентству нерухомості доцільно мати мобільний додаток під операційну систему Android, котрий полегшить їх робочий процес і дозволить частково його проводити дистанційним способом. Тому, було прийнято рішення щодо створення такого додатку. Враховуючи вищевказані обставини, було встановлено наступні вимоги до додатку:

– можливість ведення обліку клієнтів;

– можливість ведення обліку клієнтських об’єктів нерухомості;

– можливість планування та ведення обліку операцій стосовно клієнтів та об’єктів нерухомості;

– можливість пов’язувати необхідні акти та документи з операціями, клієнтами та об’єктами нерухомості;

– присутність бази співробітників агентства та можливість увійти в додаток співробітником під своїм власним акаунтом;

– можливість адміністрування додатку персоналом з відповідними повноваженнями.

Для реалізації даних можливостей необхідно вирішити наступні задачі:

– вибір засобів реалізації задач;
– розробити структуру системи мобільного додатку;
– розробити логіку роботи додатку;
– розробити структуру бази даних;
– розробити проєкт інтерфейсу мобільного додатку;
– проєктування серверного API;
– здійснення програмної реалізації мобільного додатку;
– тестування мобільного додатку.

2 ВИБІР МЕТОДІВ РІШЕННЯ ЗАДАЧ ТА ПРОЄКТУВАННЯ

2.1 Вибір засобів реалізації задач

Для реалізації front-end частини додатку було прийнято рішення щодо використання наступних технологій:

- мови програмування Java;
- HTTP-клієнт Retrofit;
- XML-розмітки для створення зовнішнього вигляду інтерфейсу;
- інтегрованого середовища розробки Android Studio.

Для реалізації Back-end частини додатку прийнято рішення щодо використання:

- мови програмування C#;
- фреймворк Entity Framework Core для високорівневої роботи з базою даних;
- в якості бази даних - СКБД MySQL;
- інтегроване середовище розробки Microsoft Visual Studio.

2.2 Структура мобільного додатку

Враховуючи обрані засоби реалізації (дивитися пункт 2.1), структура проєкту розроблена у наступному вигляді (рис. 2.1):

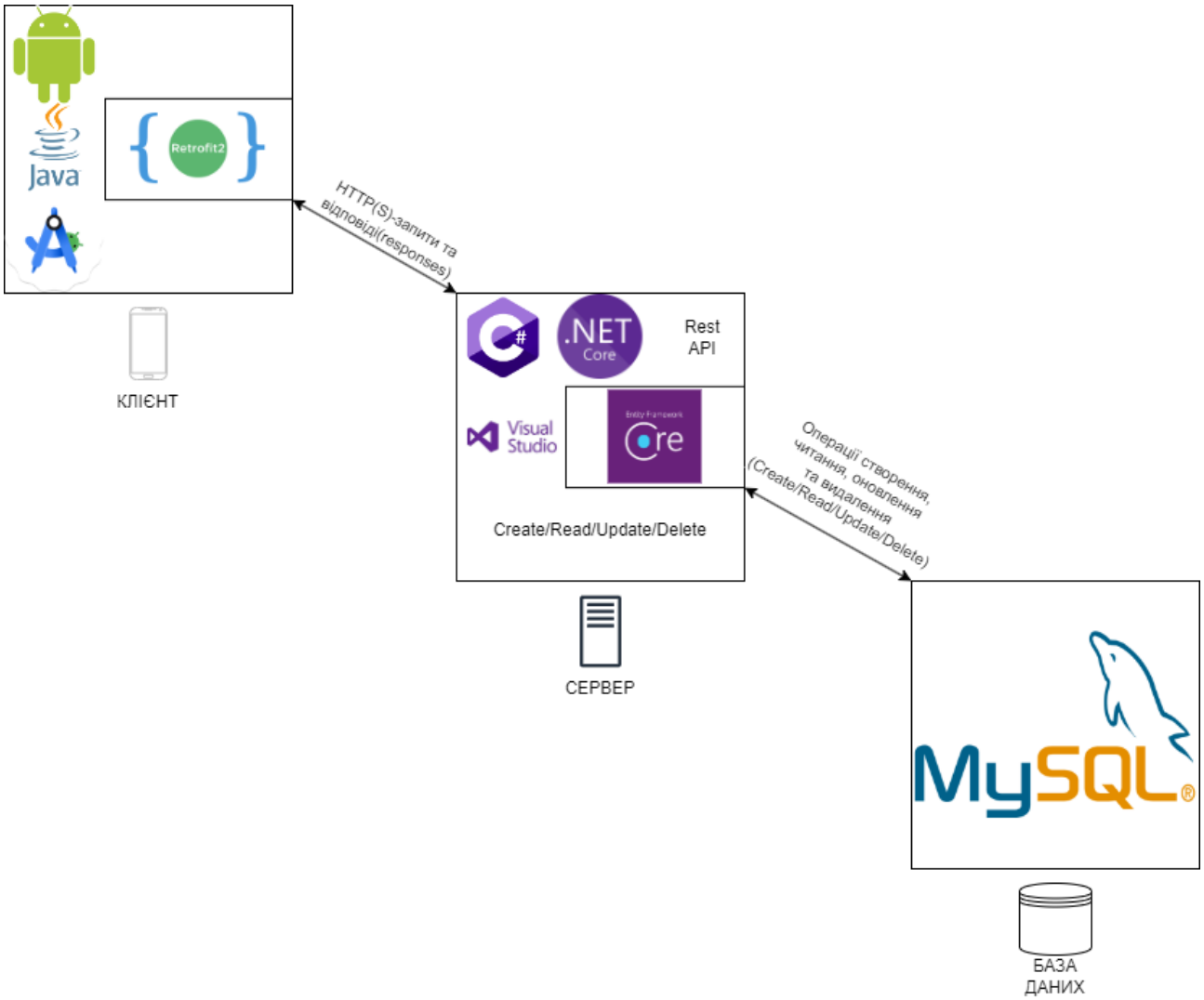


Рисунок 2.1 – Структура мобільного додатку

2.3 Діаграма варіантів використання

Діаграма варіантів використання дозволяє описати можливі сценарії взаємодії окремих зовнішніх сутностей (користувачів додатку) з системою додатку. Для мобільного додатку супроводження діяльності рієлтора створено наступну діаграму варіантів використання (рис. 2.2).

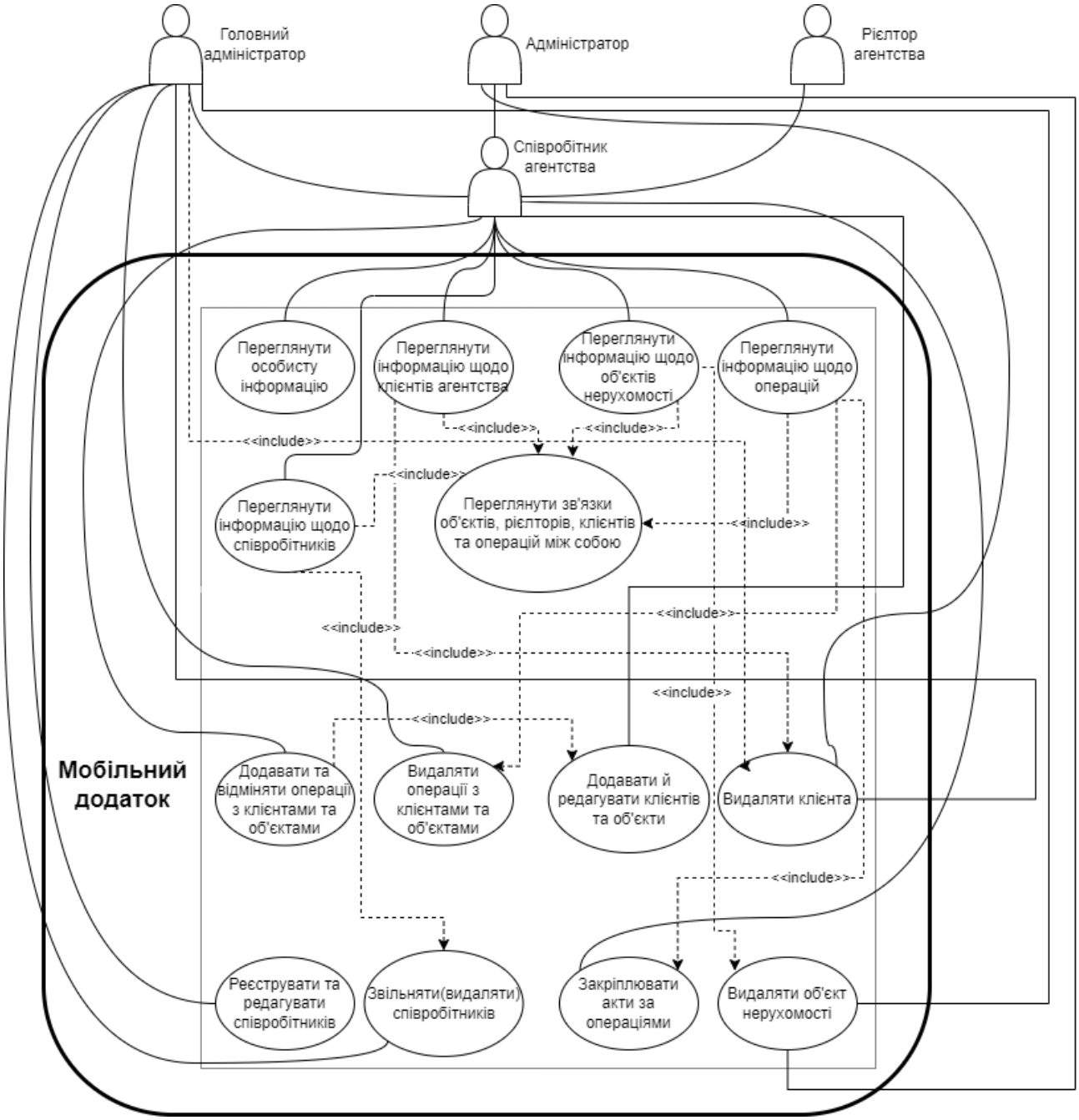


Рисунок 2.2 – Діаграма варіантів використання проєкту

2.4 Структура бази даних

На основі описаної логіки системи мобільного додатку (дивитися пункт 2.3), створюється проєкт бази даних додатку (рис. 2.3):

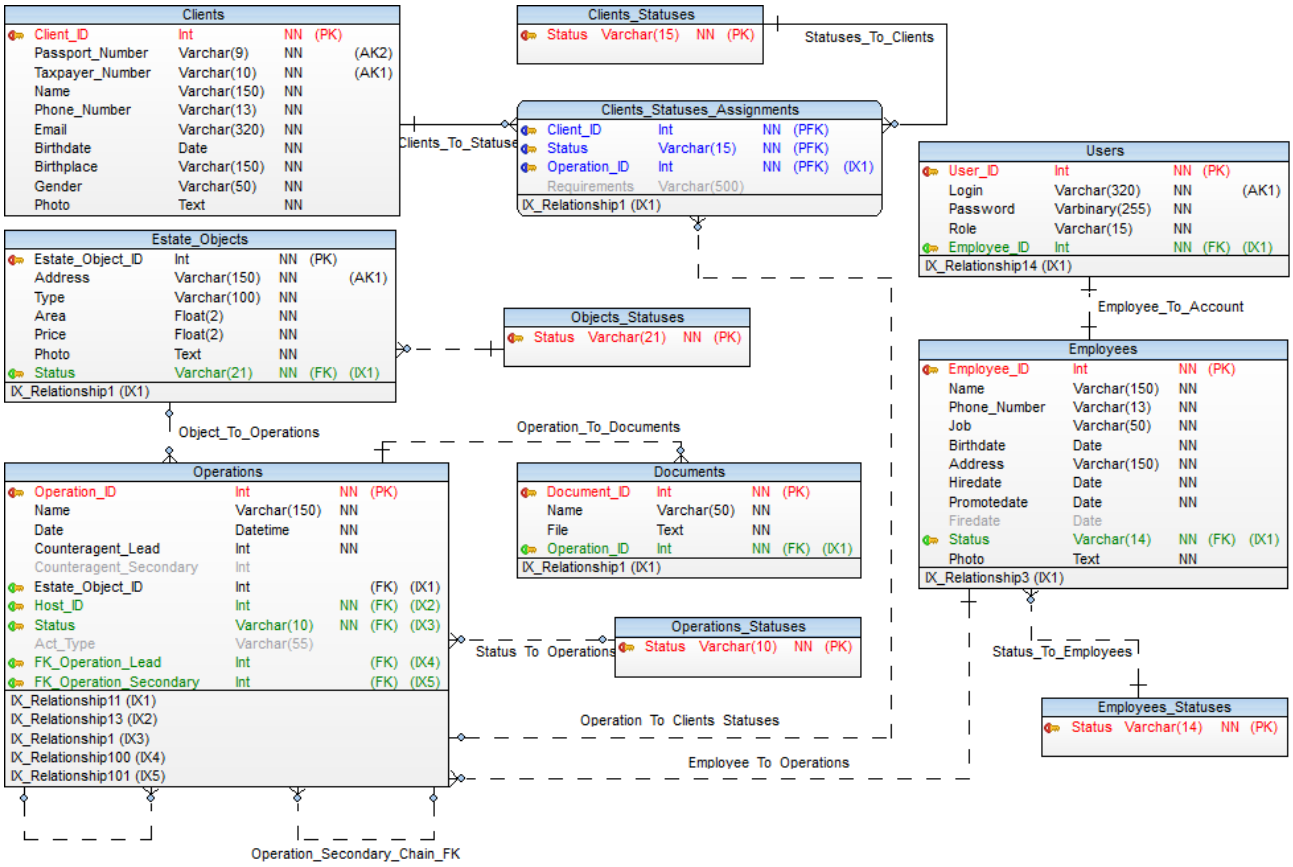


Рисунок 2.3 – Структура бази даних

Наведено опис призначення полів таблиць у базі даних (табл. 2.1 – 2.11).

Таблиця 2.1 – Опис призначення полів у таблиці клієнтів (Clients)

Поле	Призначення
Client_ID	Ідентифікатор клієнта
Passport_Number	Номер паспорту клієнта
Taxpayer_Number	Реєстраційний номер облікової картки платника податків клієнта
Name	Ім’я клієнта
Phone_Number	Номер телефону клієнта
Email	Електронна пошта клієнта
Birthdate	Дата народження клієнта
Birthplace	Місце народження клієнта
Photo	Посилання на фотографію клієнта

Таблиця 2.2 – Опис призначення полів у таблиці об’єктів нерухомості (Estate_Objects)

Поле	Призначення
Estate_Object_ID	Ідентифікатор об’єкта нерухомості
Address	Адреса об’єкта нерухомості
Type	Тип об’єкта нерухомості
Area	Площа об’єкта нерухомості
Price	Вартість об’єкта нерухомості
Photo	Фото об’єкта нерухомості
Status	Статус об’єкта нерухомості

Таблиця 2.3 – Опис призначення полів у таблиці операцій (Operations)

Поле	Призначення
Operation_ID	Ідентифікатор операції
Name	Назва операції
Date	Дата проведення операції
Counteragent_Lead	Ідентифікатор головного контрагента (продавець/орендодавець)
Counteragent_Secondary	Ідентифікатор другорядного контрагента (покупець/орендар)
Estate_Object_ID	Ідентифікатор об’єкта нерухомості
Host_ID	Ідентифікатор робітника(рієлтора), що проводить операцію
Status	Статус операції
Act_Type	Тип прив’язаного акту
FK_Operation_Lead	Посилання на попередню операцію по лінії головного контрагента
FK_Operation_Secondary	Посилання на попередню операцію по лінії другорядного контрагента

Таблиця 2.4 – Опис призначення полів у таблиці робітників (Employees)

Поле	Призначення
Employee_ID	Ідентифікатор робітника(рієлтора)
Name	Ім’я робітника(рієлтора)
Phone_Number	Номер телефону робітника(рієлтора)
Job	Посада робітника
Birthdate	Дата народження робітника(рієлтора)
Address	Місце роботи робітника(рієлтора)
Hiredate	Дата влаштування робітника(рієлтора)
Promotedate	Дата підвищення робітника(рієлтора)
Firedate	Дата звільнення робітника(рієлтора)
Status	Статус робітника(рієлтора)
Photo	Посилання на фотографію робітника(рієлтора)

Таблиця 2.5 – Опис призначення полів у таблиці акаунтів користувачів-робітників (Users)

Поле	Призначення
User_ID	Ідентифікатор акаунта
Login	Логін акаунта
Password	Хеш паролю акаунта
Role	Роль акаунта
Employee_ID	Ідентифікатор робітника, до котрого прив’язаний акаунт

Таблиця 2.6 – Опис призначення полів у таблиці призначень статусів клієнтів (Clients_Statuses_Assignments)

Поле	Призначення
Client_ID	Ідентифікатор клієнта
Status	Привласнений клієнтові статус
Operation_ID	Операція, через яку було привласнено статус
Requirements	Вимоги клієнта на продаж, купівлю або оренду, застосовується у тандемі з операціями, котрі мають відповідні привласнені акти

Таблиця 2.7 – Опис призначення полів у таблиці документів (Documents)

Поле	Призначення
Document_ID	Ідентифікатор документа
Name	Назва документа
File	Посилання на документ
Operation_ID	Операція, під час котрої було внесено документи

Таблиця 2.8 – Опис призначення полів у допоміжній таблиці статусів клієнтів (Clients_Statuses)

Поле	Призначення
Status	Текст статусу клієнта

Таблиця 2.9 – Опис призначення полів у допоміжній таблиці статусів об’єктів нерухомості (Objects_Statuses)

Поле	Призначення
Status	Текст статусу об’єкта

Таблиця 2.10 – Опис призначення полів у допоміжній таблиці статусів операцій (Operations_Statuses)

Поле	Призначення
Status	Текст статусу операції

Таблиця 2.11 – Опис призначення полів у допоміжній таблиці статусів робітників (Employees_Statuses)

Поле	Призначення
Status	Текст статусу робітника

2.5 Проєктування інтерфейсу мобільного додатку

Проєктування інтерфейсу мобільного Android-додатку для супроводження роботи рієлторів агентства нерухомоті є критично важливим етапом у розробці. Якісно спроектований інтерфейс не тільки забезпечує приємний користувацький досвід, але й сприяє досягненню цілей бізнесу. Добре спроектований інтерфейс полегшує взаємодію працівників агентства з додатком. Інтуїтивно зрозумілий та естетично привабливий дизайн підвищує задоволеність користувачів, що може призвести до збільшення ефективності роботи та поліпшення комунікації всередині команди рієлторів. Якісне проєктування інтерфейсу дозволяє досягти зрозумілості та простоти у використанні додатку. Чітко структурований інтерфейс з логічною навігацією допомагає рієлторам швидко освоїти функціонал, зменшуючи час на навчання.

Для створення проєкта інтерфейсу скористаємося сервісом Figma (режим доступу: https://www.figma.com/) (рис. 2.4). Даний сервіс дозволяє створити

візуальну репрезентацію майбутнього інтерфейсу, не використовуючи мови розмітки чи малюючи інтерфейс власноруч. Сервіс Figma містить різні заготівки під різні форми та розміри екранів пристроїв, різні фігури, шрифти для візуальної репрезентації елементів інтерфейсу, а також дозволяє проводити роботу над проєктуванням у кооперації з колегами.

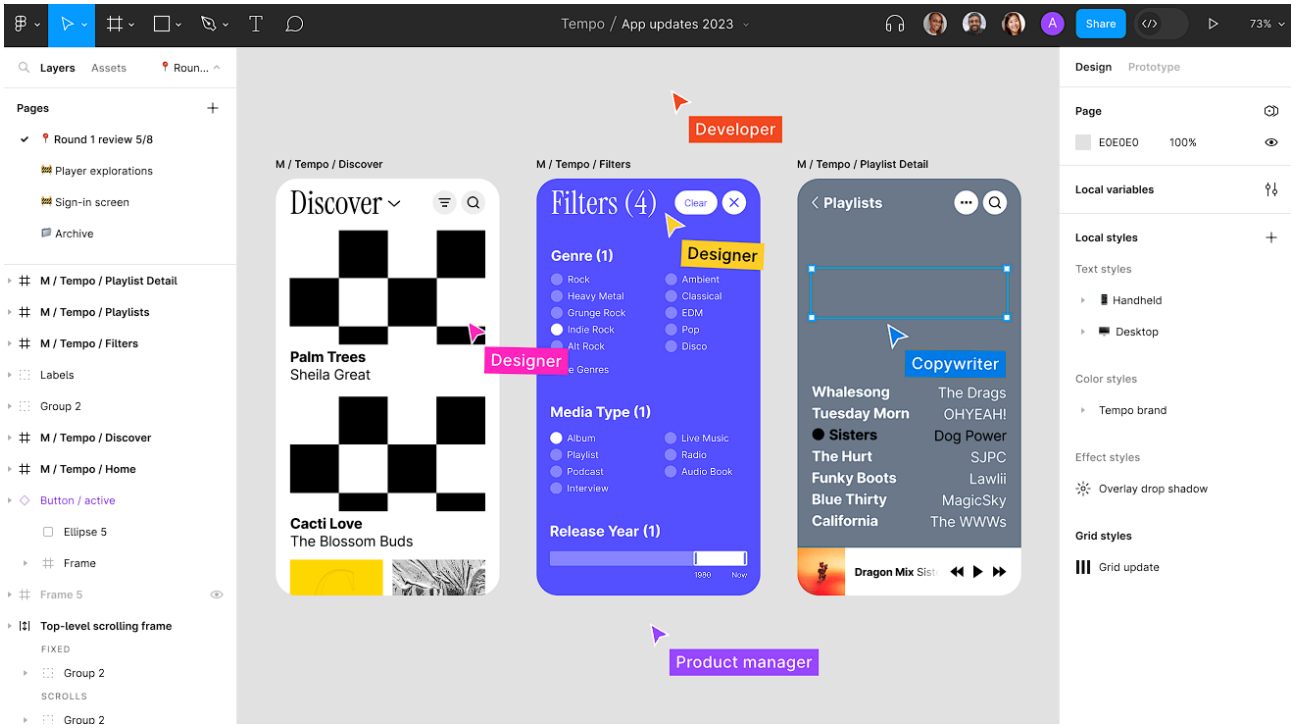


Рисунок 2.4 – Сторінка проєктування у сервісі Figma

Представимо результати проєктування інтерфейсу на прикладі кількох сторінок.

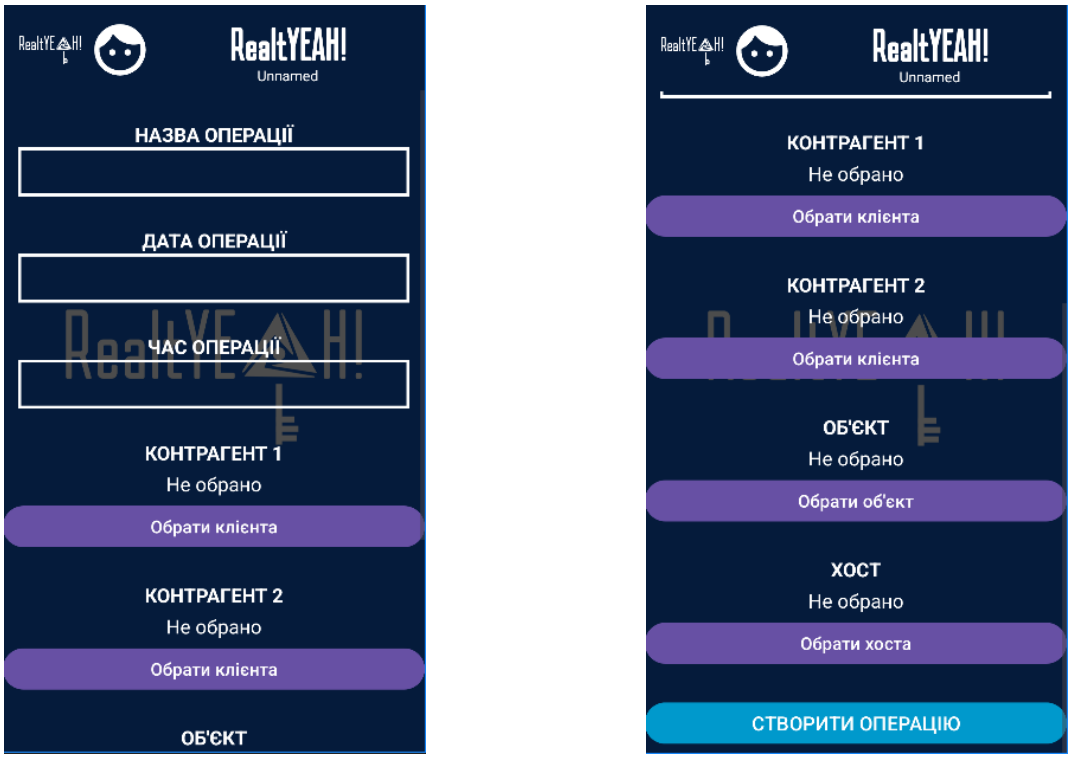
Сторінка клієнтів (рис. 2.5) є перехідною ланкою перед сторінкою пошуку клієнтів. На неї можна потрапити за умови переходу з головної сторінки або з випадаючого меню. Дана сторінка містить значок, що репрезентує людей-клієнтів, під котрим коротко описано призначення розділу клієнтів у додатку. Даний крок дозволяє зрозуміти користувачу, для яких цілей він може звернутися до розділу клієнтів, а значок з назвою розділу привертає увагу рієлторів для прочитування опису. Нижче (рис. 2.5 а–б) знаходяться кнопки переходу до окремих категорій пошуку клієнтів. Кнопки, як і сторінка, також містять окремі значки та зрозумілу назву.

а б

Рисунок 2.5 – Проєкт сторінки клієнтів: значок та опис (а), кнопки переходу до окремих категорій пошуку клієнтів (б)

Сторінка створення операції (рис. 2.6) є однією з складових найважливішого механізму додатку – обліку операцій. Операції дозволяють проводити облік всіх подій з клієнтами у ланцюжку від їх найпершого звернення в агентство до заключення угоди на купівлю/продаж або оренду/здачу в оренду. Операції також є ключовим фактором у визначенні статусів клієнтів, об’єктів, а також поточного власника об’єкта. Також операції несуть планувальну функцію – маючи поля дати та часу вони дозволяють рієлторам агентства чітко планувати графік роботи та не допускати накладання зустрічей у різних випадках.

Отже, для успішного та зручного обліку операцій необхідно про неї знати такі дані, як: назва, дата, час, клієнти-сторони угоди, об’єкт нерухомості та рієлтор, що проводить операцію(хост). Для таких даних, як дата, час та назва, виділимо окремі текстові поля введення (рис. 2.6 а), так як ці дані є атрибутами операції і не належать іншим ентіті. Для таких даних, як сторони угоди(клієнти, контрагенти), об’єкт нерухомості та хост, застосуємо кнопки переходу на сторінки пошуку відповідних ентіті (рис. 2.6 б), щоб дати змогу вибрати вже існуючі записи щодо них з бази даних, або ж створити нових. Для відображення результату вибору над відповідними кнопками додаються текстові поля, що відображатимуть імена обраних клієнтів та хоста, а також адресу обраного об’єкта. Кожний поле вибору даних має над собою підпис для розуміння рієлторами, а кнопки вибору даних ці підписи дублюють.



а б

Рисунок 2.6 – Проєкт сторінки створення операції: верхня частина з текстовими полями (а), кнопки переходу до пошуку окремих ентіті (б)

Сторінка робітника (рис. 2.7) є важливим елементом обліку рієлторів агентства. Дана сторінка повинна давати змогу побачити дані щодо рієлтора, його

статус, пов’язані операції, клієнтів, об’єктів нерухомості. Для адміністраторів також повинна бути доступною можливість редагувати дані рієлтора. Для головного адміністратора, у додаток до редагування, також доступна можливість створювати акаунт для входу в додаток для даного рієлтора, можливість його звільнення зі збереженням у базі даних, а також звільнення з повним видаленням.

Блок з даними рієлтора винесемо в окреме виділене округлим прямокутником місце на вершині сторінки. Блок міститиме текстові поля з відповідними підписами. Під блоком з даними розташуються кнопки переходу до окремих сторінок, де відображатимуться пов’язані з рієлтором операції, клієнти та об’єкти. Кожна кнопка повинна містити опис, що вона виконує, а над кнопкою цей опис дублюється більш помітним шрифтом для привернення уваги. Кнопки редагування, створення акаунту та звільнення рієлтора виділені різними кольорами та мають більші відступи між собою. Це дозволяє виділити адміністративні кнопки від загальних, котрі доступні лише адміністраторам і невидимі для звичайних рієлторів.

Рисунок 2.7 – Проєкт сторінки робітника

Зображення решти проєкту інтерфейсу наведені у додатку Б.

2.6 Серверна частина додатку

Серверна частина Android-додатку, реалізована з використанням мови C#, Entity Framework Core, Rest API та бази даних MySQL, являє собою потужний і масштабований бекенд, який забезпечує взаємодію клієнтської частини додатку з сервером. Використання C# як основної мови програмування дозволяє створити надійний і ефективний код, який легко підтримувати та розширювати. Entity Framework Core, як Object-Relational Mapping інструмент, спрощує роботу з базою даних MySQL, забезпечуючи зручний спосіб управління об'єктами та зберіганням даних на високому рівні.

Створення Rest API на базі ASP.NET Core дозволяє організувати ефективну комунікацію між Android-додатком і сервером через HTTP (HTTPS) протокол. Це забезпечує швидкий і безпечний обмін даними, використовуючи стандартизовані методи запитів, такі як GET, POST, PUT і DELETE. База даних MySQL виступає надійним сховищем для зберігання даних додатку, завдяки своїй популярності, підтримці, продуктивності та здатності обробляти великі обсяги інформації. Усі ці компоненти разом створюють міцну основу для розробки сучасного і масштабованого мобільного додатку, забезпечуючи високу продуктивність та гнучкість системи.

Так як серверна частина додатку використовуватиме Entity Framework Core та Rest API, доцільно навести діаграму класів (рис. 2.8), що описуватиме сутності у відповідності до розробленого проєкту бази даних, а також контекст, сервіси та MVC-контролери з методами, що викликатимуться з клієнтської частини через HTTP-запити.

Як відповідник до серверної частини додатку, клієнтська частина повинна містити інтерфейс, що описуватиме методи контролерів серверної частини додатку, сутності, що повністю відповідають створеним на сервері та використовуватимуться для серіалізації/десеріалізації даних у JSON-форматі, а також різні допоміжні класи та класи сторінок додатку (активіті). Дану структуру також було описано у вигляді діаграми класів (рис. 2.9).

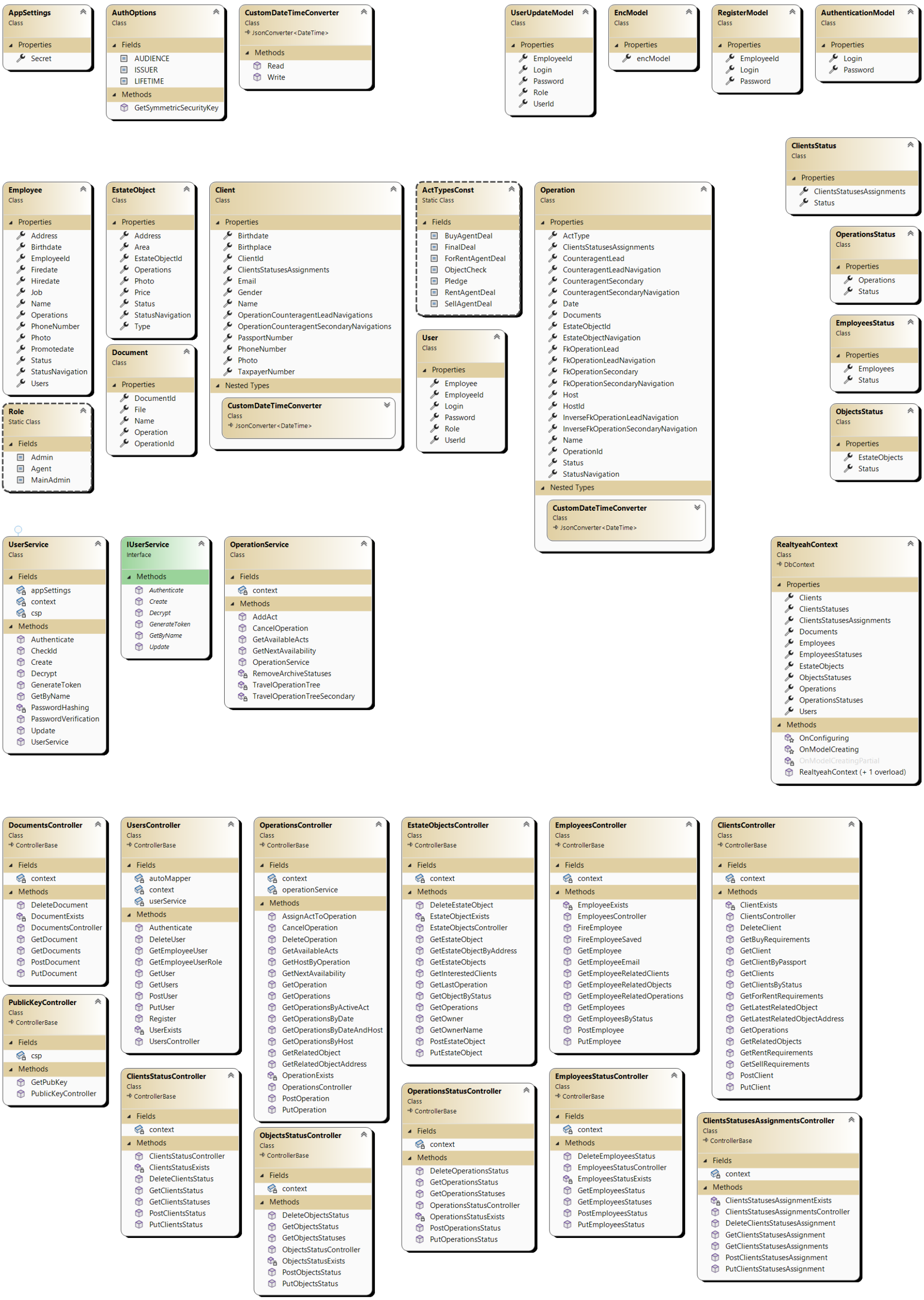


Рисунок 2.8 – Діаграма класів серверної частини додатку

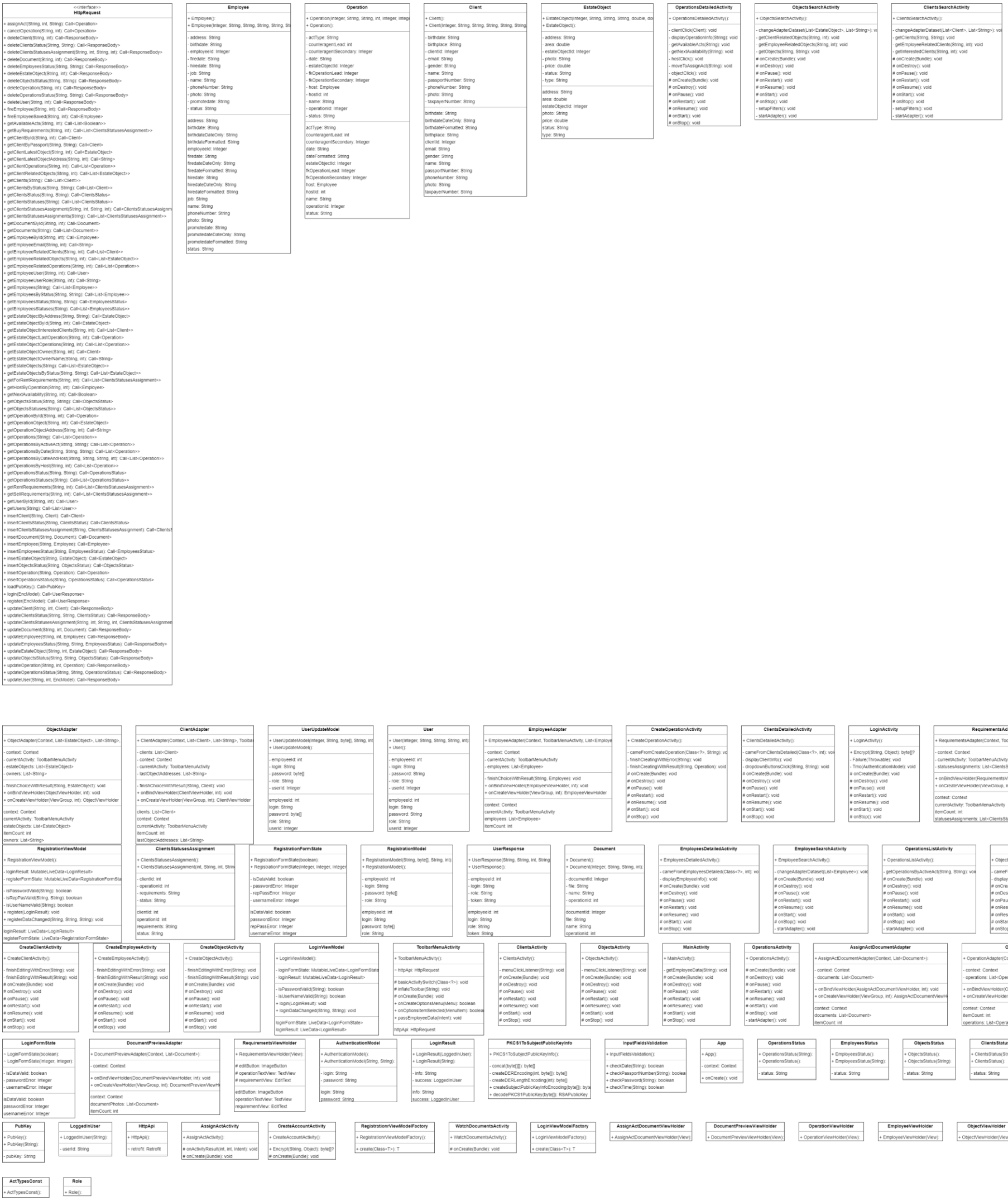


Рисунок 2.9 – Діаграма класів клієнтської частини додатку

3 ПРАКТИЧНА РЕАЛІЗАЦІЯ

3.1 Програмна реалізація

Код XML-розмітки, клієнтської та серверної частини додатку наведений у додатку А.

3.1.1 Реалізація розмітки інтерфейсу

Розробка інтерфейсу Android-додатку відбувається за допомогою написання XML-розмітки. Дана розмітка міститься у файлах з розширенням .xml, зазвичай, у директорії ресурсів, папка «layout». Розмітка вказує, які елементи знаходяться на сторінці, їх розміщення, ієрархію та залежності, зовнішній вигляд.

На початку реалізації розмітки необхідно виділити деякі спільні для всіх активіті елементи. В даному випадку, такими є верхня панель додатку (рис. 3.1), що відображається на всіх сторінках, окрім сторінки авторизації. Також загальними елементами розмітки є задній фон програми з логотипом (рис 3.2). Дані елементи розмітки винесемо в окремі файли «activity_toolbar_menu.xml», «bg_logo.xml» та підключатимемо за допомогою тегу «include».

Рисунок 3.1 – Верхня панель додатку

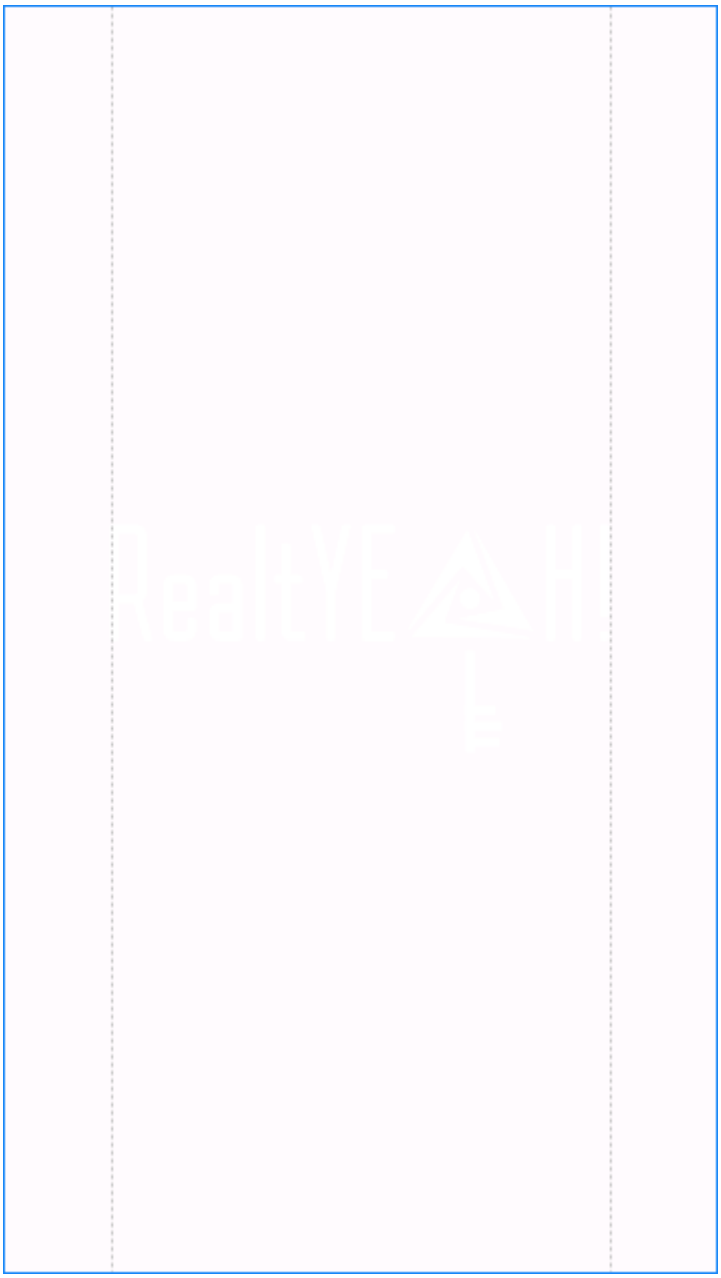


Рисунок 3.2 – Задній фон додатку з логотипом

Наступним кроком у реалізації розмітки буде створення окремих активіті додатку. Проілюструємо одну з таких активіті – головну (рис. 3.3). Дана активіті містить теги «include», котрі підключають розмітку верхньої панелі та заднього фону. Серед іншого – кнопки переходу на відповідні розділи додатку: клієнтів, об’єктів, операцій, співробітників. Дані кнопки є тегами «ConstraintLayout», що містять у собі теги «ImageView», «TextView» та «Guideline» для опису розділів, за перехід до яких вони відповідають. Також присутні теги «ImageView», «TextView», «Button» для фотографії користувача, імені користувача, виходу з

акаунта, показу особистої інформації. Натискання на кнопку показу інформації провокує відображення під нею ще одного тега – «TableLayout», тобто таблиці з основною особистою інформацією користувача.

Рисунок 3.3 – Зовнішній вигляд MainActivity

Аналогічним способом створимо розмітку інших активіті додатку. Код розмітки окремих активіті наведено у додатку А.

В рамках реалізації розмітки варто приділити увагу активіті, що містять у собі елементи, точна кількість котрих не може бути відомою заздалегідь. Такими активіті можуть бути, наприклад, пошук клієнтів (рис. 3.4). Заздалегідь нам не відомо, скільки клієнтів міститься у базі агентства, вони динамічно підтягуються

з неї. Отже, щоб відобразити кожного клієнта, необхідний контейнер, що вміє динамічно адаптовуватися під різну кількість елементів розмітки. Таким контейнером є «RecyclerView». Даний контейнер дозволяє вказати власний адаптер, що успадковується від базового адаптеру контейнера, а отже дозволяє відображати різну кількість конкретного елементу розмітки, зовнішній вигляд котрого – на розсуд, необхідний для програми.

Рисунок 3.4 – Активіті пошуку клієнтів

Перед використанням «RecyclerView», очевидно, необхідно попередньо визначити розмітку елементів, що в ній будуть відображатися. Створимо елемент

для відображення фотографії клієнта, а також деякої загальної інформації щодо нього (рис 3.5). Розмітка зроблена на прозорому фоні, використовуючи білий колір контурів та шрифтів тегів «TextView», що виділить їх на темному фоні активіті додатку (див. рис. 3.6):

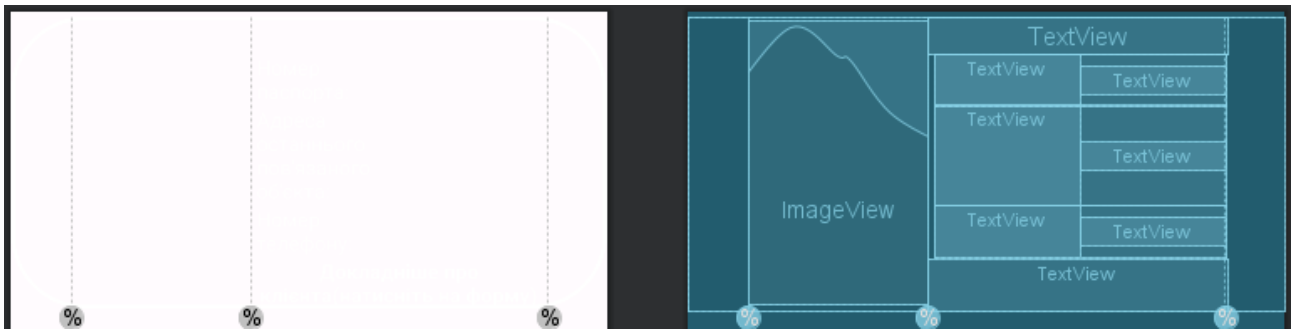


Рисунок 3.5 – Елемент розмітки для відображення у RecyclerView клієнта

Рисунок 3.6 – RecyclerView з елементом розмітки клієнта в активіті пошуку клієнтів

Аналогічним чином створимо елементи розмітки для адаптерів для інших об’єктів. Код розмітки окремих елементів наведено у додатку А.

Такі частини розмітки, як кольори, тексти заголовків(значення заздалегідь відоме для всіх), теми – задаються окремо у файлах «colors.xml», «strings.xml», «themes.xml». Далі їх можна використовувати у відповідних параметрах тегів розмітки XML, наприклад, «android:background» для вказання фонового кольору, «android:text» для вказання тексту текстових полів.

На верхній панелі додатку міститься випадаюче меню, що дозволяє перейти в різні розділи додатку поза межами головної активіті (рис 3.7).

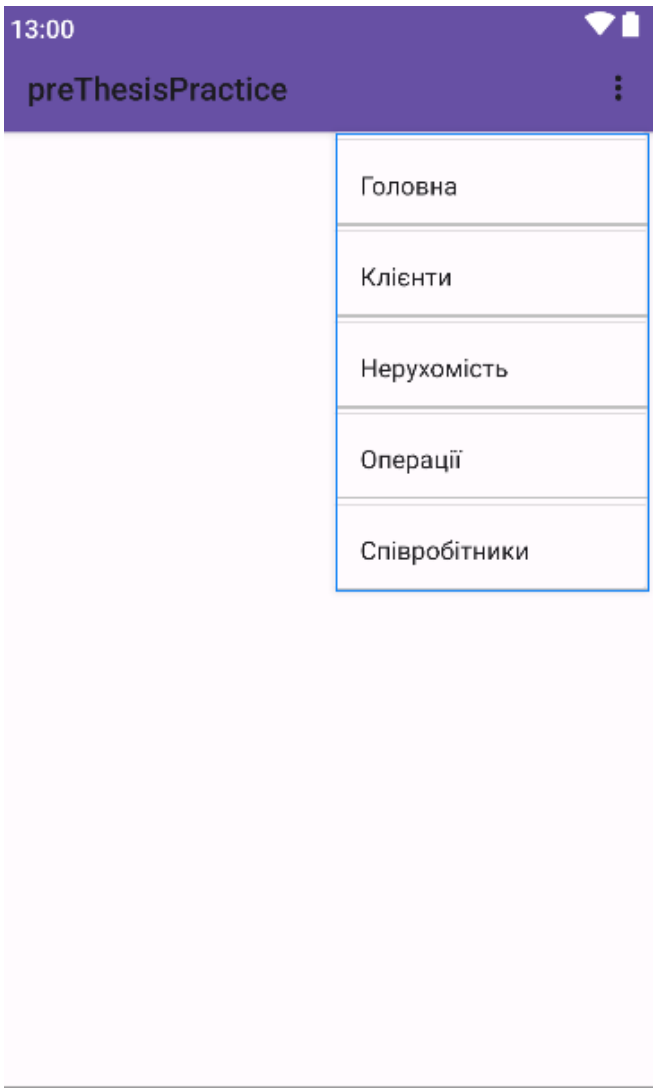


Рисунок 3.7 – Випадаюче меню верхньої панелі

3.1.2 Реалізація серверної частини додатку

Серверна частина додатку використовує мову програмування C# у кооперації з Entity Framework Core та ASP.NET MVC контролерами. Entity Framework Core є високорівневою оболонкою для взаємодії з базою даних, в той час як MVC дозволяє виконувати HTTP-запити, котрі взаємодіють з Entity Framework Core та повертають певні дані на клієнтську частину додатку.

Першим кроком буде створення entity-класів, що представляють запис даних у конкретній таблиці з конкретними колонками. За приклад візьмемо клас ClientsStatusesAssignment, котрий представляє таблицю Clients_Statuses_Assignments:

```
using System.Text.Json.Serialization;

namespace RealtYeahBackend.Entities;

public partial class ClientsStatusesAssignment
{
    public int ClientId { get; set; }

    public string Status { get; set; } = null!;

    public int OperationId { get; set; }

    public string? Requirements { get; set; }

    [JsonIgnore]
    public virtual Operation? Operation { get; set; } = null!;

    [JsonIgnore]
    public virtual Client? ClientNavigation { get; set; } = null!;

    [JsonIgnore]
    public virtual ClientsStatus? StatusNavigation { get; set; } = null!;
}
```

Даний клас містить у собі поля: ClientId (ідентифікатор клієнта), Status (статус клієнта), OperationId (операція, на котрій було привласнено статус), Requirements (побажання клієнта). Ці поля є основними та явно представлені у базі даних. Поля, типи яких можуть приймати значення null, але не повинні набувати його у базі даних, відмічаються «= null!». Крім основних полів, є необхідність вказувати навігаційні властивості, котрі є, іншими словами, уособленням Foreign Key обмежень. Так як таблиця «Clients_Statuses_Assignments» містить у собі

потрійний Primary Foreign Key, що посилається на колонки «ClientId» та «OperationId» з таблиць «Clients» та «Operations» відповідно, а також на колонку «Status» з таблиці «Clients_Statuses» – існує необхідність вказати аналогічні поля класів Operation, ClientsStatus, Client. Таким чином, маючи об'єкт класу ClientsStatusesAssignment, можливо звернутися до залежних записів без додаткових запитів до бази даних. Анотація «JsonIgnore» біля навігаційних властивостей слугує для того, щоб у випадку передачі об'єкту та його перетворення у Json не відбувався цикл об'єктів. Json намагатиметься описати, наприклад, поле «operation», клас Operation котрого, в свою чергу, також має навігаційну властивість на ClientsStatusesAssignment, а отже відбувається циклічне описування навігаційних властивостей.

Наступним кроком є написання контексту. Контекст містить опис усіх наборів даних, відповідно до кожного entity-класу. Також, контекст повинен перевантажувати метод OnModelCreating() базового класу DbContext. Даний метод описує кожне entity окремо у рамках контексту – його поля, індекси, обмеження, типи, розміри та назви колонок, якими вони є у базі даних, а також ставить їм у відповідь поля класів entity.

Наступним кроком є написання контролерів MVC. Контролери містять методи, котрі збирають певні дані з бази даних та повертають їх за HTTP-запитами, що надходять від клієнтської частини додатку. За приклад візьмемо «ClientsController.cs». Контролер містить п'ять основних методів:

- метод за запитом *HttpGet(Clients)*, котрий повертає колекцію всіх клієнтів з бази даних;
- метод за запитом *HttpGet(Clients/{id})*, котрий повертає конкретного клієнта з заданим параметром ID;
- метод за запитом *HttpPut(Clients/{id})*, котрий оновлює запис клієнта з конкретним ID у базі даних;
- метод за запитом *HttpPost(Clients)*, котрий додає новий запис клієнта до бази даних;

– метод за запитом *HttpDelete(Clients/{id}),* котрий видаляє запис клієнта з конкретним ID з бази даних.

Крім описаних методів, до контролера можна додавати власні методи. У «ClientsController.cs» серед таких є метод GetClientsByStatus(), котрий повертає усіх клієнтів з певним статусом за запитом HttpGet (Clients/statuses/{status}). Під час створення методів важливо слідкувати за тим, щоб запити не повторювалися на різних методах – виникає конфлікт HTTP-запитів. Також, за допомогою анотацій можна зробити увесь доступ до контролера лише авторизованим користувачам з певними чи будь-якими ролями, або ж зробити його таким для конкретних методів.

Як допоміжні речі слугують моделі та класи констант. Моделі дозволяють передавати дані певного entity-класу у неповному об’ємі, не з усіма полями або з додатковими, для подальшого їх маппінгу на повноцінне ентіті. Вони є проміжним рівнем для трансформування даних між рівнем entity та логікою програми. Класи констант – це класи, що містять незмінні поля з певними встановленими значеннями. Єдиним реалізованим класом констант є «ActTypesConst.cs», котрий містить у собі назви можливих актів, що прив’язуються до операцій.

3.1.3 Реалізація клієнтської частини додатку

Клієнтська частина додатку повинна забезпечити динамічність однієї з її компонент – розмітки інтерфейсу – та взаємодію з серверною частиною додатку. Це включає:

– інтерактивний UI;

– завантаження даних за запитами до сервера;

– внутрішня логіка додатку та обробка результатів.

Та інші задачі. Розглянемо пункти більш детально.

Інтерактивний UI – це виконання певних дій при взаємодії користувача з інтерфейсом(розміткою) додатку. Для забезпечення такої взаємодії у програмній реалізації використовувалися:

- Java-файли різних Activity;
- адаптери для створення динамічних списків екземплярів певних entity-класів;
- обробники подій(події натискання, довгого натискання, тощо);
- випадаючі меню та списки;
- діалогові вікна;
- спливаючі повідомлення;
- загальна верхня панель.

Завантаження даних за запитами до сервера – це виконання HttpGet-запитів до сервера в певних ситуаціях під час спрацювання певних обробників подій або під час завантажування активіті. Стосується необхідності, наприклад, завантажити з сервера колекцію всіх клієнтів з певним статусом під час вибору категорії клієнтів для пошуку в «ClientsActivity.java». Завантажена колекція може ініціалізувати масив клієнтів на клієнтській частині додатку, після чого даний масив може бути використаний в адаптерах для відображення динамічних списків.

Внутрішня логіка додатку – це забезпечення коректної роботи та взаємодії між різними елементами додатку. Включає в себе:

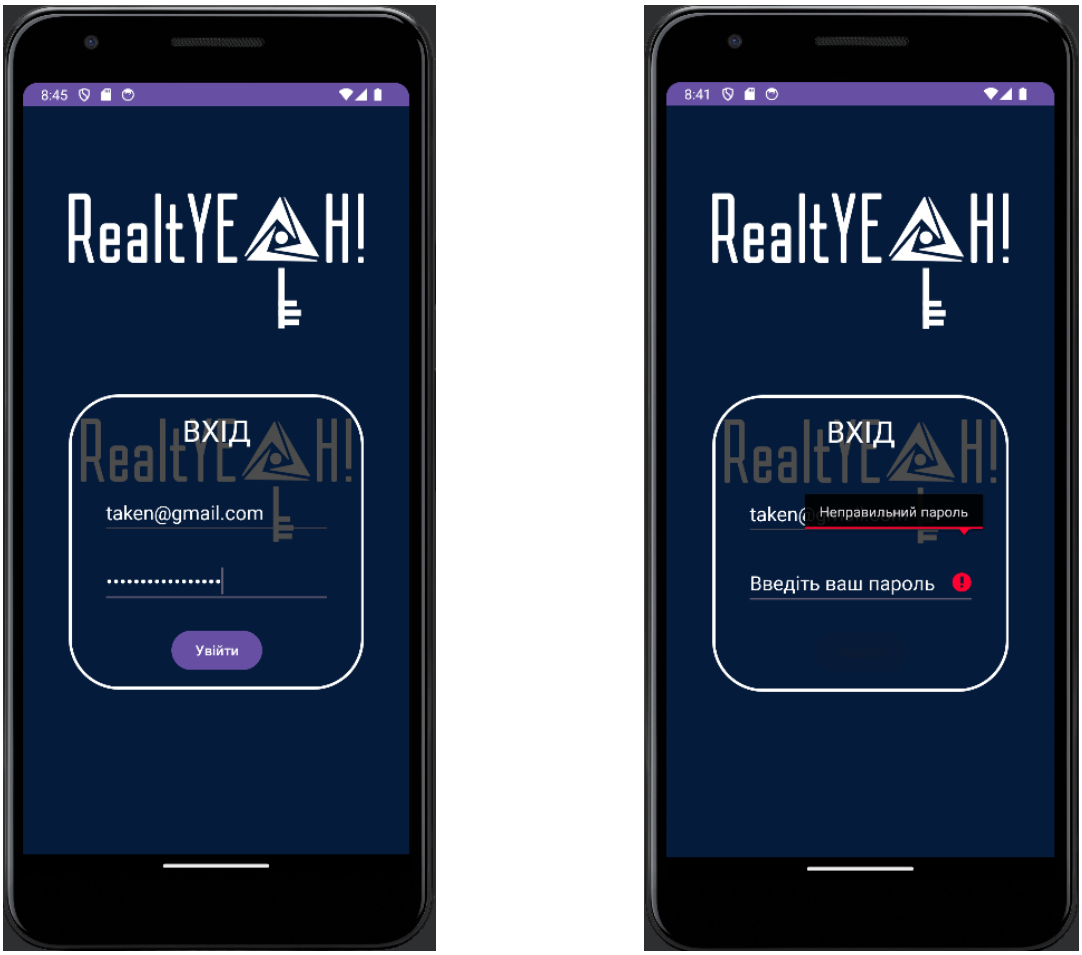
- обробка користувацького вводу;
- збірка екземплярів entity-класів та відправка їх на сервер під час їх додавання або оновлення;
- блокування певних елементів інтерфейсу в залежності від результатів запитів до сервера або певних користувацьких дій;
- блокування певних можливостей додатку для користувачів з невідповідними ролями;
- робота з Intent, переходи між активіті, обмін даних між активіті.

Код розмітки, клієнтської та серверної частин додатку наведений у додатку А.

3.2 Мануальне тестування

Для перевірки додатку скористаємося мануальним тестуванням, демонструючи роботу основних його частин – активіті.

Початкова сторінка (активіті) додатку – це активіті авторизації. Авторизуємося під користувачем «taken@gmail.com» (рис. 3.8 а). Якщо формат паролю або логіна не є правильним – отримуємо відповідні повідомлення біля полей (рис 3.8 б):



а б

Рисунок 3.8 – Активіті авторизації (а) та введення невірних даних (б)

Після успішної авторизації користувач потрапляє на головну активіті (рис. 3.9). Переглянемо основну інформацію щодо користувача за допомогою кнопки «Показати особисту інформацію» (рис. 3.9).

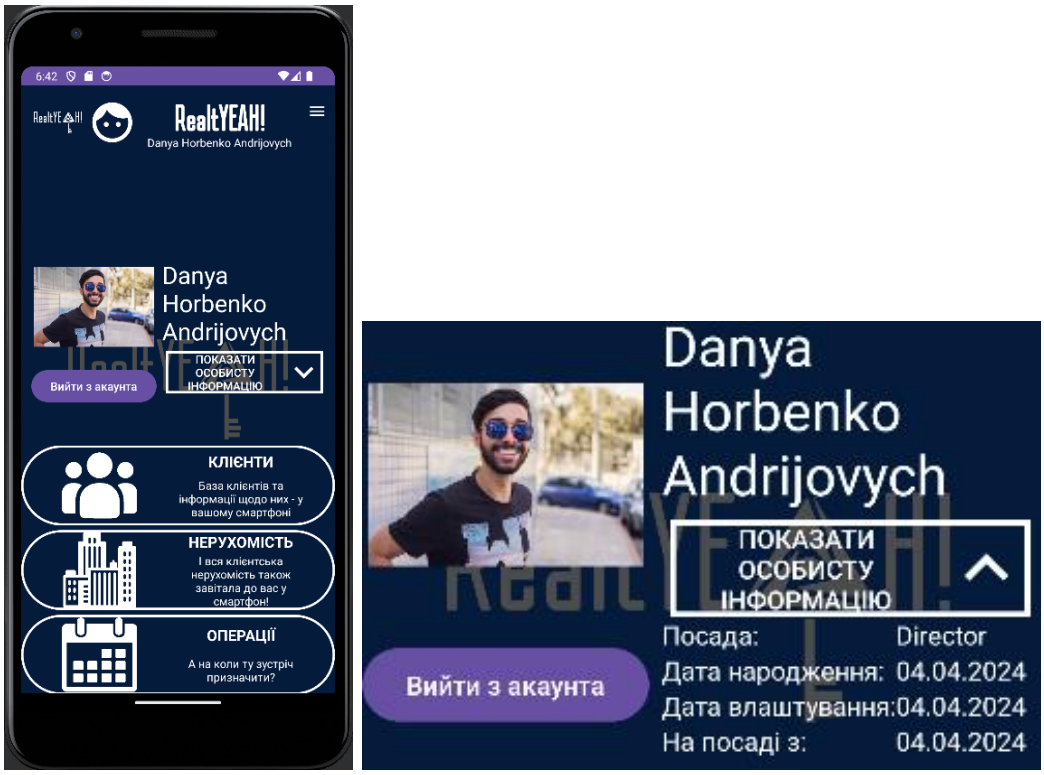
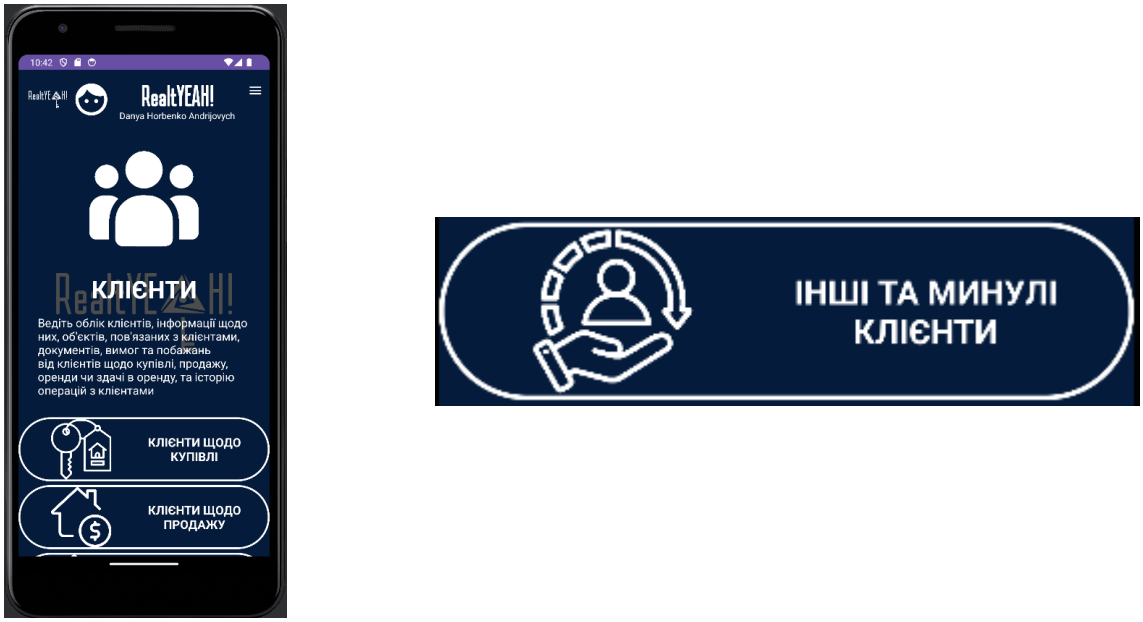


Рисунок 3.9 – Головна активіті та особиста інформація користувача

Перейдемо до розділу «Клієнти» (рис. 3.10 а). У даному розділі оберемо певну категорію клієнтів для пошуку. Наприклад, «Інші та минулі клієнти» (рис. 3.10 б):



а б

Рисунок 3.10 – Активіті розділу клієнтів (а) та категорія клієнтів «Інші та минулі» (б)

Відсортуємо клієнтів (рис. 3.11 а) за номером телефона, ввівши у пошукове поле частину номера на натиснувши відповідний перемикач (рис 3.11 б). Перейдемо на сторінку клієнта Petro Petrovych (рис. 3.11 в):

а б в

Рисунок 3.11 – Активіті пошуку клієнтів (а) з фільтруванням за телефонним номером (б) та перехід на сторінку клієнта (в)

На сторінці клієнта побачимо основну інформацію щодо обраного клієнта, а також матимемо змогу передивитися пов’язані з ним об’єкти та операції. Також, для кожної пов’язаної операції можна залишити нотатку з побажаннями та редагувати їх в будь-який момент (рис. 3.12).

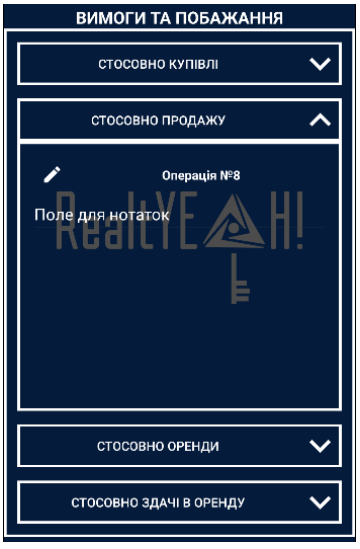
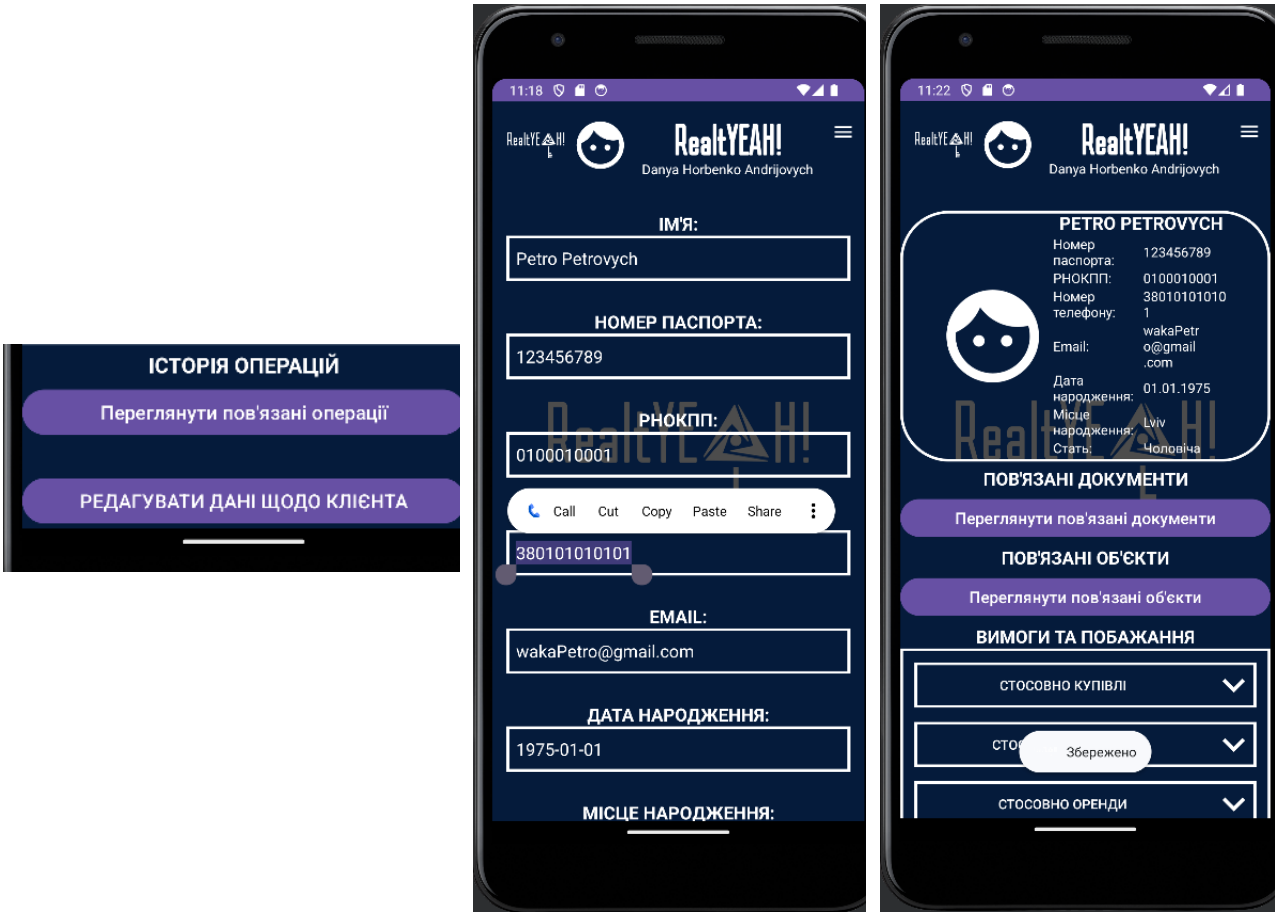


Рисунок 3.12 – Побажання клієнта

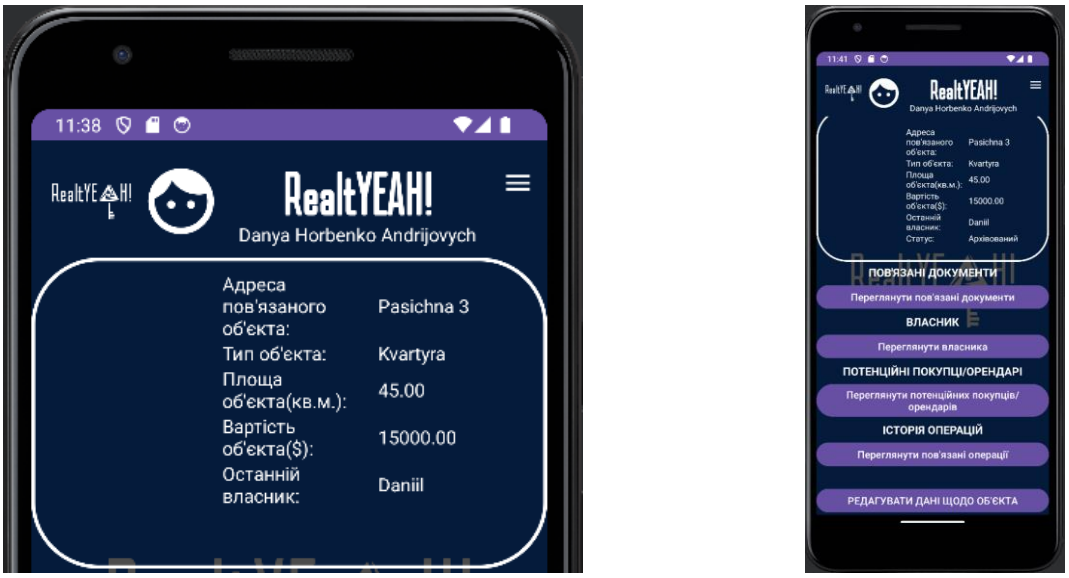
У нижній частині активіті знаходиться кнопка редагування клієнта (рис. 3.13 а). Активіті редагування клієнта дозволяє змінити значення даних клієнта, окрім його унікального ключа, якщо ці дані задовольняють умовам валідації. Змінимо номер клієнта Petro Petrovych на «380101010101» (рис. 3.13 б). Переглянемо зміни на сторінці клієнта (рис. 3.13 в).



а б в

Рисунок 3.13 – Кнопка редагування клієнта (а), активіті редагування (б) та змінений номер на активіті клієнта (в)

Переглянемо пов’язані об’єкти клієнта, натиснувши на кнопку «Переглянути пов’язані об’єкти». Опинимося на активіті пошуку об’єктів (рис. 3.14 а), звідки зможемо перейти на сторінку об’єкта, що міститься у результаті (рис. 3.14 б).



а б

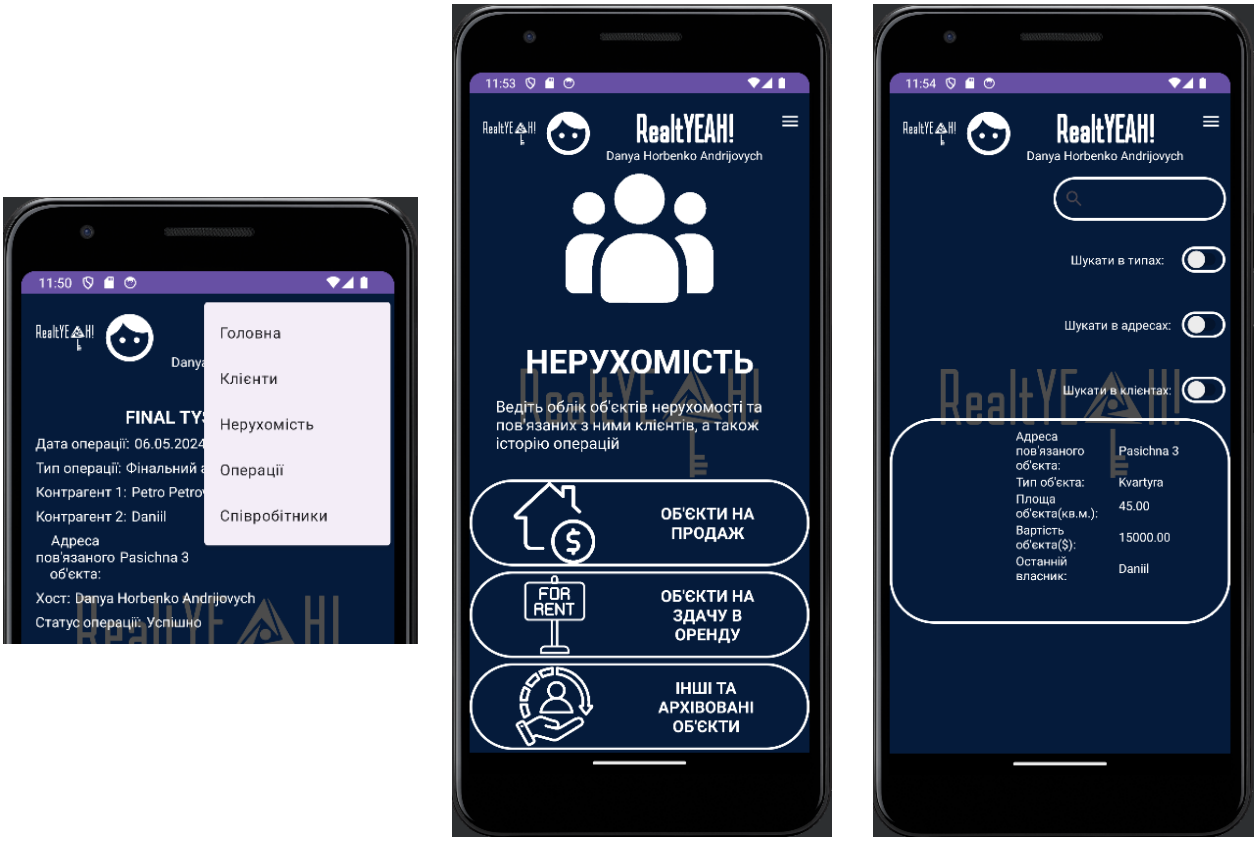
Рисунок 3.14 – Активіті пошуку пов’язаних об’єктів (а) та сторінка пов’язаного об’єкта (б)

Повернемося до активіті клієнта. Нині перейдемо на активіті пов’язаних операцій (рис. 3.15 а). Оберемо одну з операцій та перейдемо на її сторінку (рис. 3.15 б).

а б

Рисунок 3.15 – Активіті-список пов’язаних операцій (а) та сторінка пов’язаної операції (б)

Через випадаюче меню у верхній панелі (рис. 3.16 а) перейдемо до розділу об’єктів нерухомості (рис. 3.16 б). У даному розділі оберемо категорію об’єктів для пошуку(наприклад, архівовані об’єкти) та перейдемо до активіті пошуку об’єктів (рис. 3.16 в).



а б в

Рисунок 3.16 – Випадаюче меню верхньої панелі (а), розділ об’єктів нерухомості (б) та пошук об’єктів (в)

Функціонал та поведінка даного розділу аналогічні розділу клієнтів. Кнопки перегляду операцій перенаправляють на сторінку з операціями, де відображаються операції з вказаним об’єктом. Кнопки перегляду власника та потенційних клієнтів виконують функцію, аналогічну кнопці перегляду пов’язаних об’єктів – лише в даному випадку відображаються пов’язані клієнти.

Через випадаюче меню у верхній панелі перейдемо до розділу операцій (рис. 3.17).

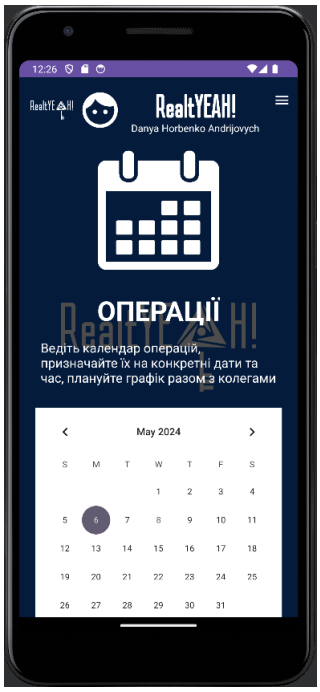


Рисунок 3.17 – Розділ операцій

У розділі операцій оберемо дату 1 травня на календарі та натиснемо кнопку «Показати операції». За бажанням користувача, справа від даної кнопки розташований чекбокс, відмічання котрого означає показ операцій лише тих, де користувач, як рієлтор, призначений у якості провідника-хостом. Переглянемо список знайдених операцій на 1 травня (рис. 3.18).

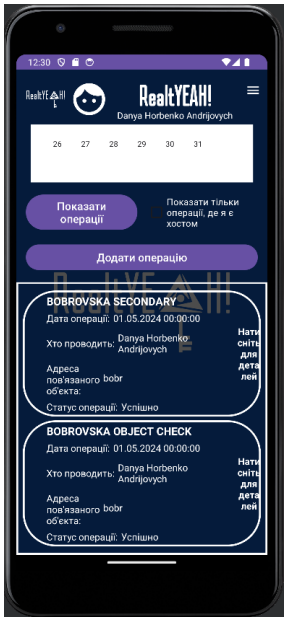
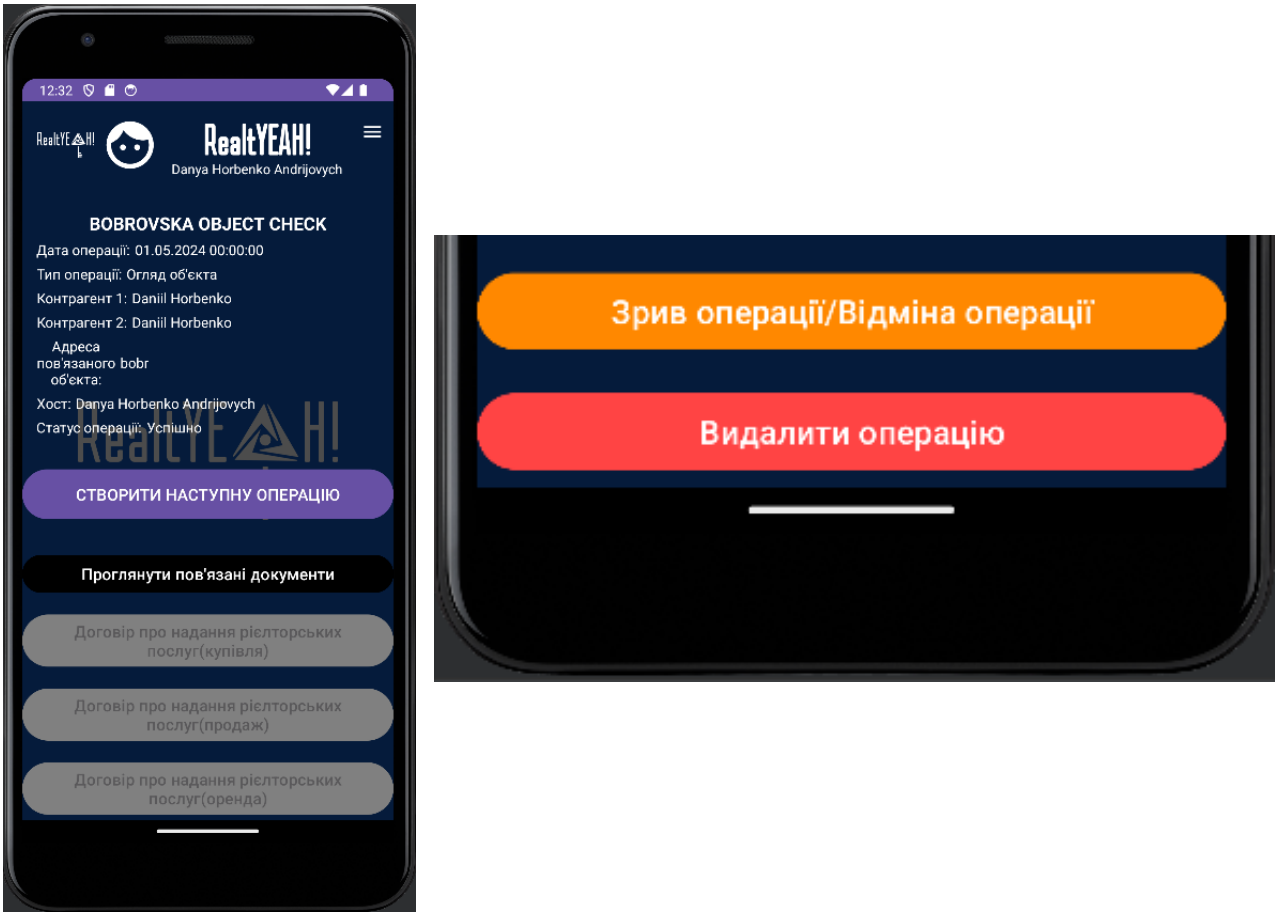


Рисунок 3.18 – Знайдені операції на 1 травня

Перейдемо на активіті другої знайденої операції та переглянемо її більш детальну інформацію (рис. 3.19 а) Дана операція є останньою у власному ланцюгу та не має прив’язаний фінальний акт, отже надається можливість використати її в ланцюгу, або ж відмінити/видалити (рис. 3.19 б). Інші кнопки є заблокованими.



а б

Рисунок 3.19 – Активіті операції «Bobrovska Object Check» (а) та кнопки відміни та видалення операції (б)

Повернемося до розділу операцій на натиснемо кнопку «Додати операцію». Створимо дві початкові операції, що не мають другого контрагента та/або об’єкта. Користувач опиняється на сторінці створення операції та вводить основні дані щодо нової операції – назву, дату та час (рис. 3.20).

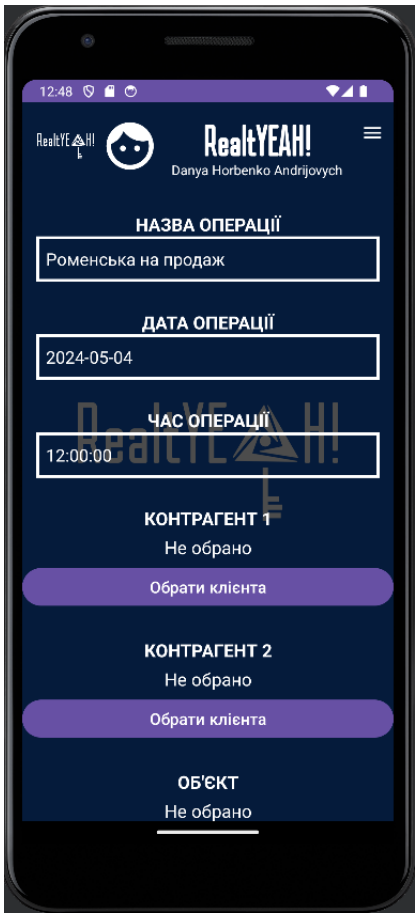
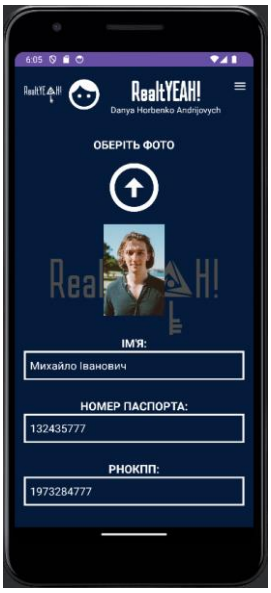


Рисунок 3.20 – Активіті створення операції

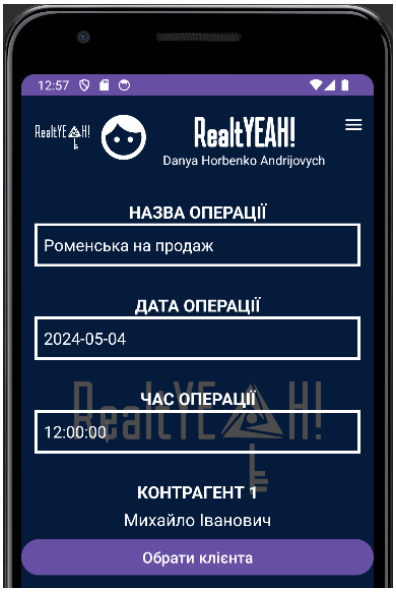
Для вибору клієнта користувач натискає кнопку «Обрати клієнта» біля поля «Контрагент 1». Користувач потрапляє на сторінку пошуку клієнтів. Якщо необхідного клієнта немає серед наявних у базі даних, користувач натискає кнопку «Додати клієнта» (рис. 3.21 а). Користувач заповнює поля з даними клієнта у правильному форматі (рис. 3.21 б) та додає клієнта до бази даних. Далі клієнта можна обрати довгим натисканням з активіті пошуку клієнтів (рис. 3.21 в). Після вибору клієнта його ім’я відображається над відповідною кнопкою вибору (рис. 3.21 г).

а



б

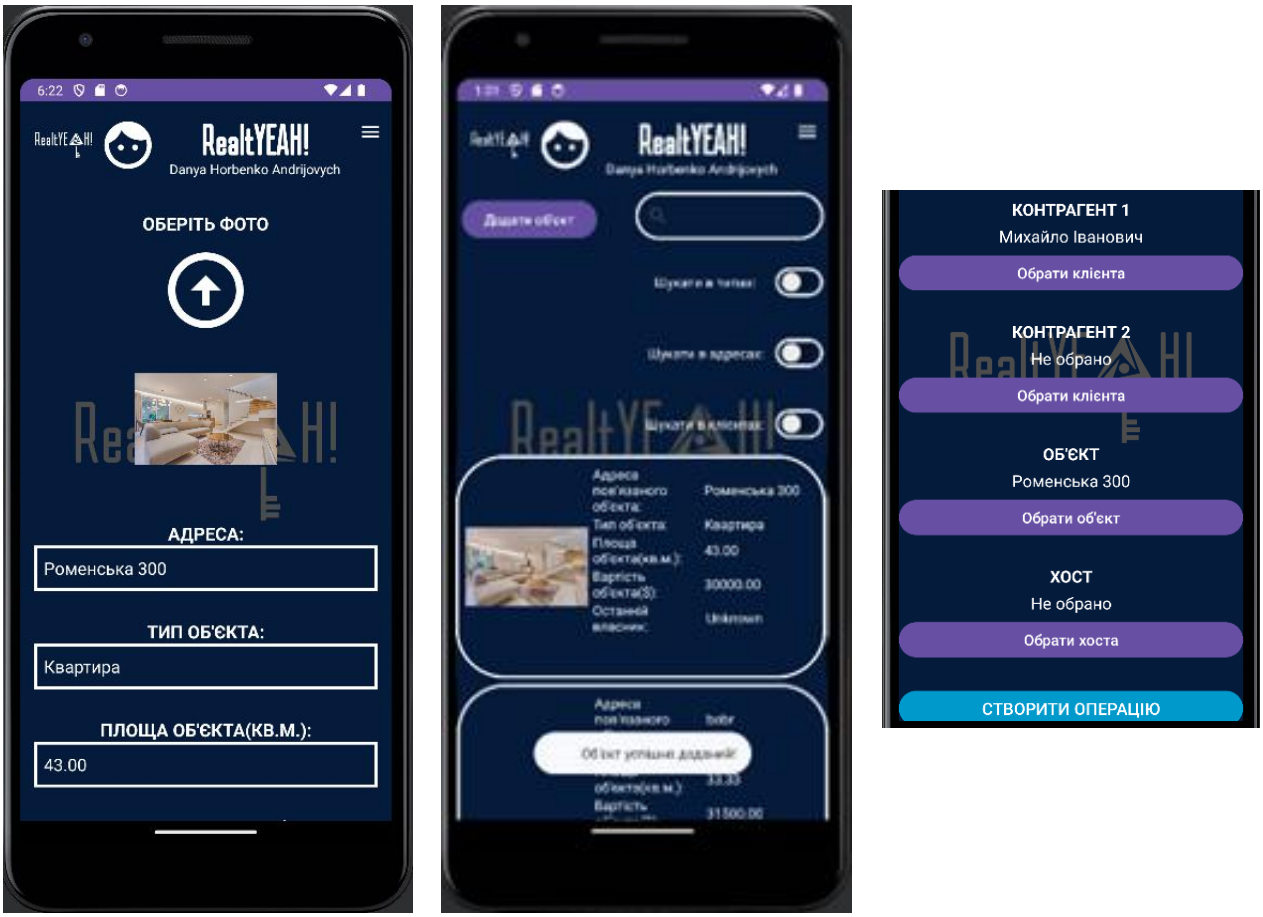
в



г

Рисунок 3.21 – Процес вибору клієнта для операції: пошук (а), додавання (б), вибір (в), відображення імені (г)

Додавання об’єкту до операції відбувається аналогічним чином, як це є з додаванням клієнта (рис. 3.22).

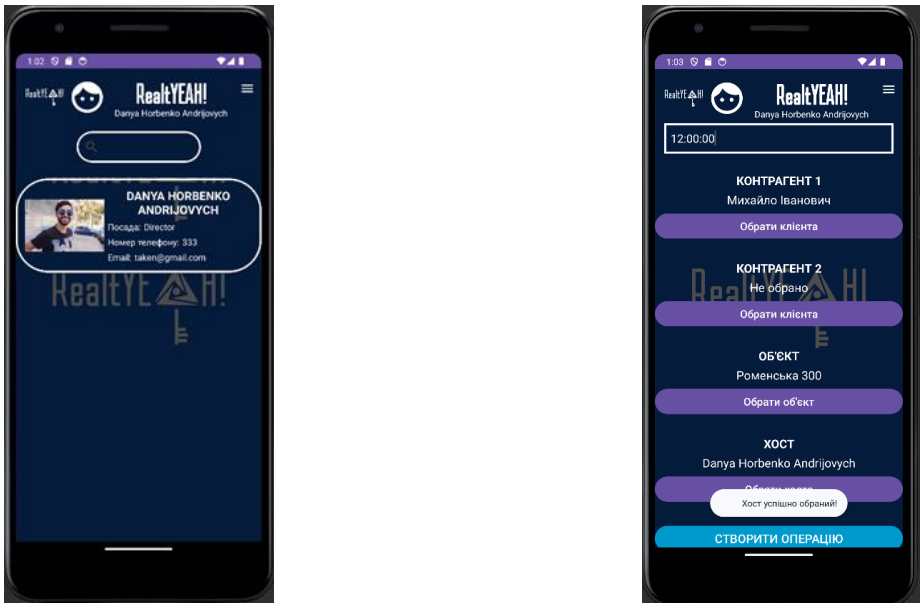


а б в

Рисунок 3.22 – Додавання нового об’єкта до операції: створення (а), вибір (б), відображення адреси (в)

Для початкових операцій, коли клієнт бажає заключити угоду з агентством щодо купівлі, продажі або оренди, вказується лише один клієнт у якості першого контрагента. Якщо клієнт бажає купити або взяти об’єкт нерухомості в оренду – об’єкт до початкової операції також не прив’язується.

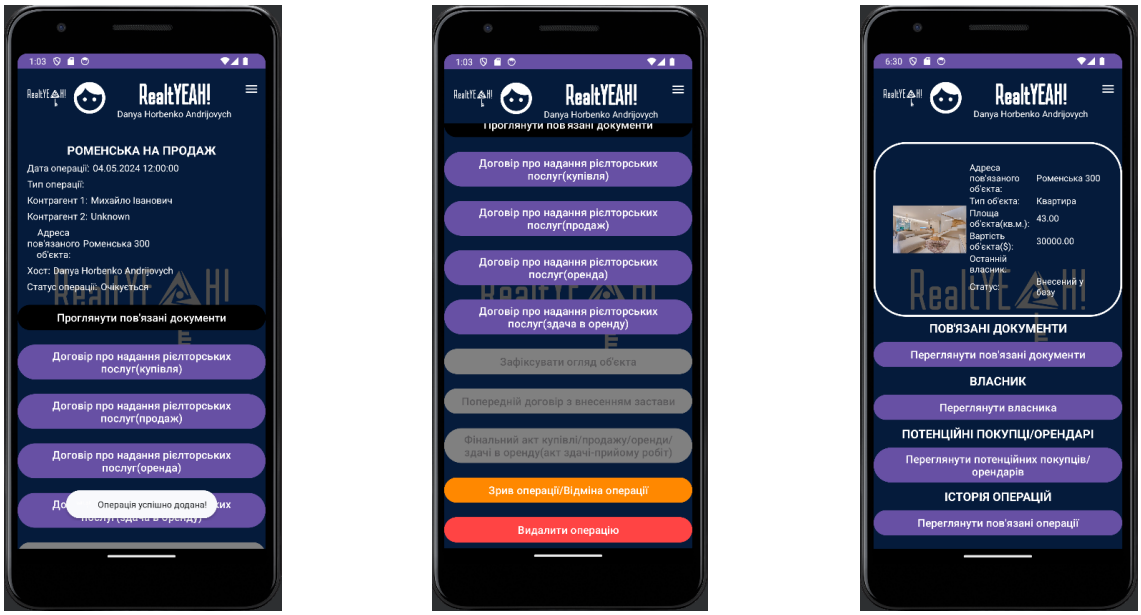
Останнім кроком створення операції є вибір рієлтора, що вважатиметься відповідальним за її проведення, тобто хоста. Користувач натискає кнопку «Обрати хоста» та обирає рієлтора з пошуку (рис. 3.23 а), після чого ім’я обраного хоста відображається над кнопкою вибору (рис. 3.23 б).



а б

Рисунок 3.23 – Вибір хоста: вибір у пошуку (а), відображення імені (б)

Користувач натискає кнопку «Створити операцію» та потрапляє на її сторінку (рис. 3.24 а). Для затвердження доступні лише 4 початкові акти та кнопки відміни/видалення операції (рис. 3.24 б). Об’єкт, що був прикріплений до операції, отримав статус «Внесений у базу» (рис. 3.24 в).



а б в

Рисунок 3.24 – Сторінка створеної операції (а): дозволені акти (б), сторінка доданого об’єкта зі статусом (в)

Створимо нову операцію аналогічним способом. В даній операції додамо нового клієнта, що бажає купити об’єкт нерухомості, а отже в операції не повинен вказуватися об’єкт (рис. 3.25).

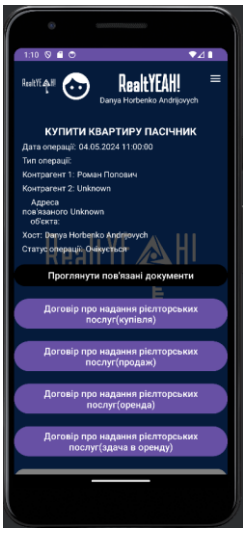
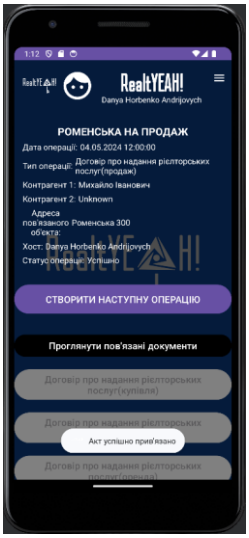
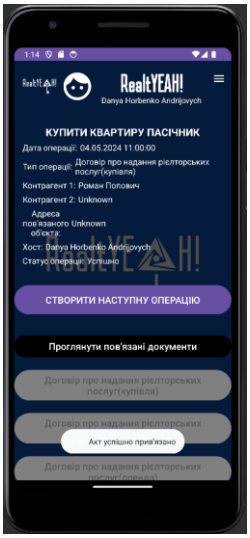
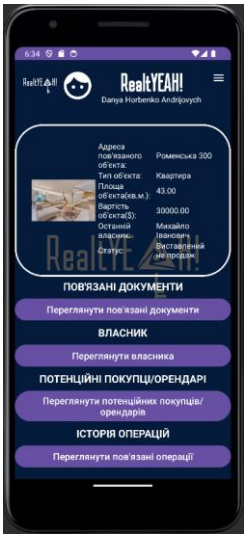


Рисунок 3.25 – Друга нова операція з клієнтом-покупцем

Прикріпимо до створених операцій відповідні акти: першій – на продаж, другій – на купівлю (рис. 3.26 а-б). Тип прикріплених актів відображається у полі «Тип операції». Об’єкт, що був прив’язаний до операції з актом на продаж, отримає статус «Виставлений на продаж» (рис. 3.26 в), а клієнти – статуси «Продавець»/«Покупець» відповідно.





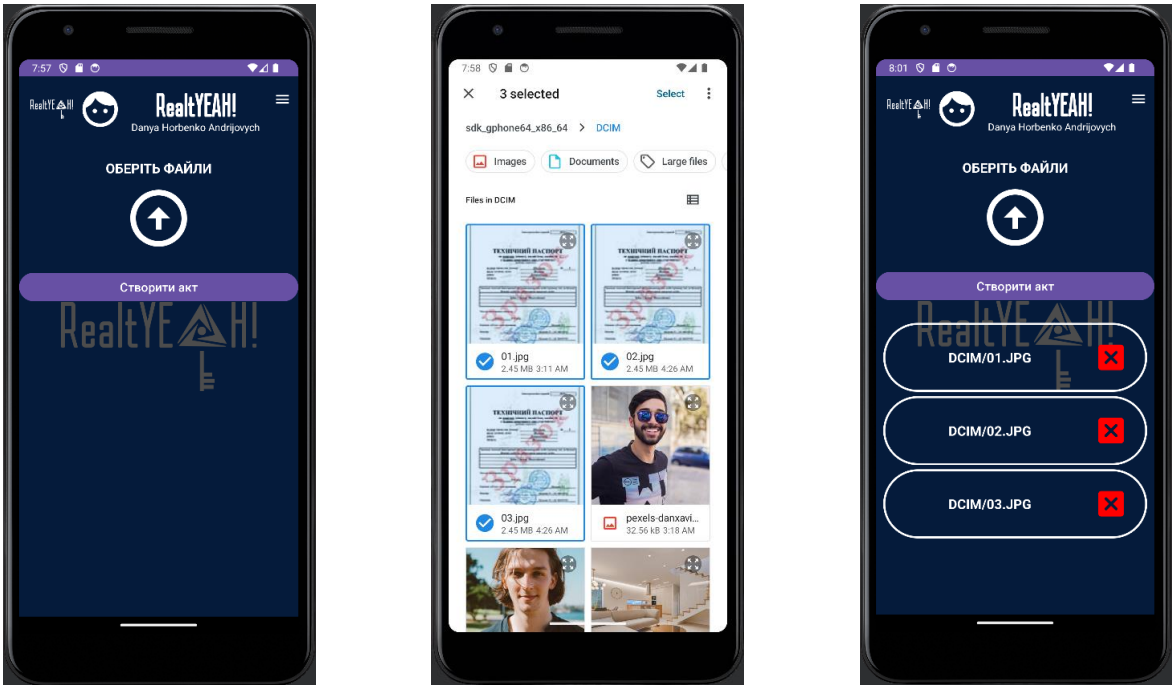


а б в

Рисунок 3.26 – Прикріплення актів до операцій: щодо продажу (а), щодо купівлі (б), статус об’єкту на продаж (в)

Під час прикріплення актів до операцій, рієлтору надається можливість завантажити документи (фотографії, текстові документи), що необхідні для проведення даної операції. Надалі, прикріплені до операції документи будуть відображатися також і на сторінках клієнтів та об’єкту, що прив’язані до даної операції, а також будуть доступні для завантажування на мобільний пристрій. На прикладі операції «Роменська на продаж» (див. рис. 3.27 а) протестуємо дану можливість.

Натискаючи кнопку необхідного акту, користувач потрапляє на сторінку створення акту (рис. 3.27 а), де, натиснувши на значок стрілки вгору, можна обрати файли з мобільного телефону (рис. 3.27 б) та завантажити їх у додаток (рис. 3.27 в).



а б в

Рисунок 3.27 – Прикріплення документів під час прив’язування актів до операції: сторінка заключення акту (а), вибір файлів з мобільного пристрою (б), підвантажені файли у додатку (в)

Підвантажені файли перед заключенням акту можна, за бажанням, прибрати, натиснувши червону кнопку-хрестик біля необхідного файлу. Виключимо зі списку останній файл (рис. 3.28).

Рисунок 3.28 – Виключення підвантажених файлів

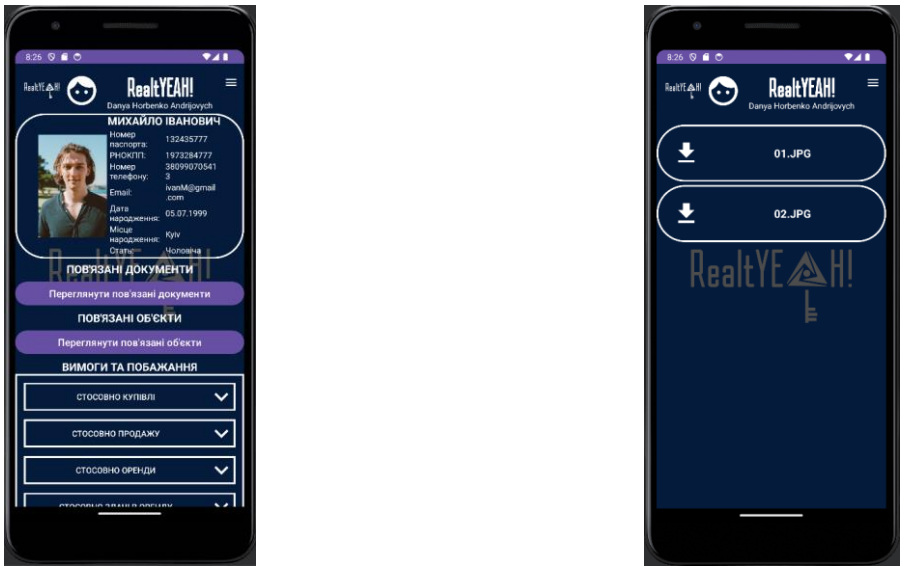
Після заключення акту, документи відображаються на сторінці операції після натискання кнопки «Проглянути пов’язані документи» (рис. 3.29 а). Дані файли можна завантажити назад на мобільний пристрій, або завантажити іншим робітникам агентства. Для завантаження необхідно натиснути кнопку стрілочки вниз біля необхідного документа. Перевіримо папку «Downloads» на наявність завантажених файлів (рис. 3.29 б).

а

б

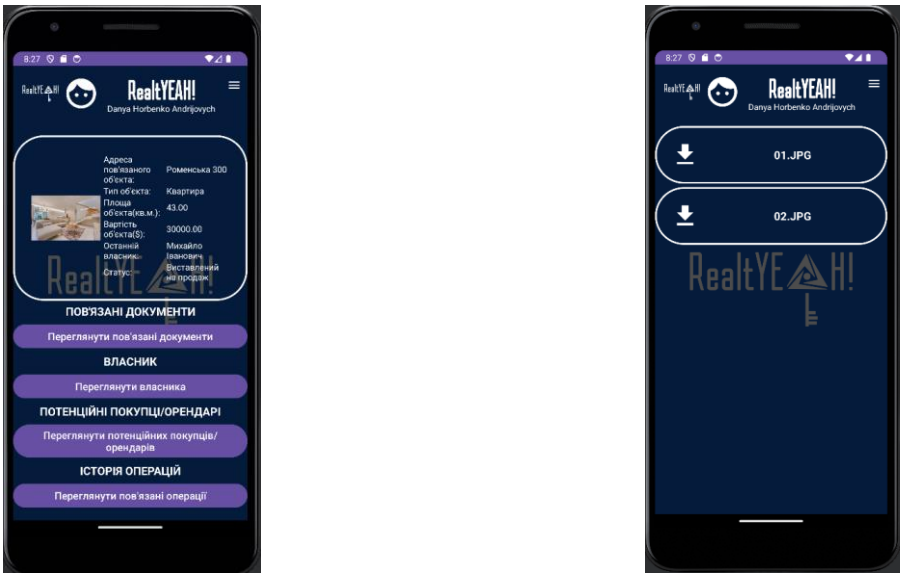
Рисунок 3.29 – Завантаження документів з додатку: наявність документів при натисканні відповідної кнопки на сторінці операції (а), завантажені документи на мобільному пристрої (б)

Також перевірено наявність прикріплених документів, якщо подивитися на них з боку клієнта (рис. 3.30) або об’єкта нерухомості (рис. 3.31).



а б

Рисунок 3.30 – Наявність документів під час їх перегляду з боку клієнта: сторінка клієнта (а), сторінка документів (б)

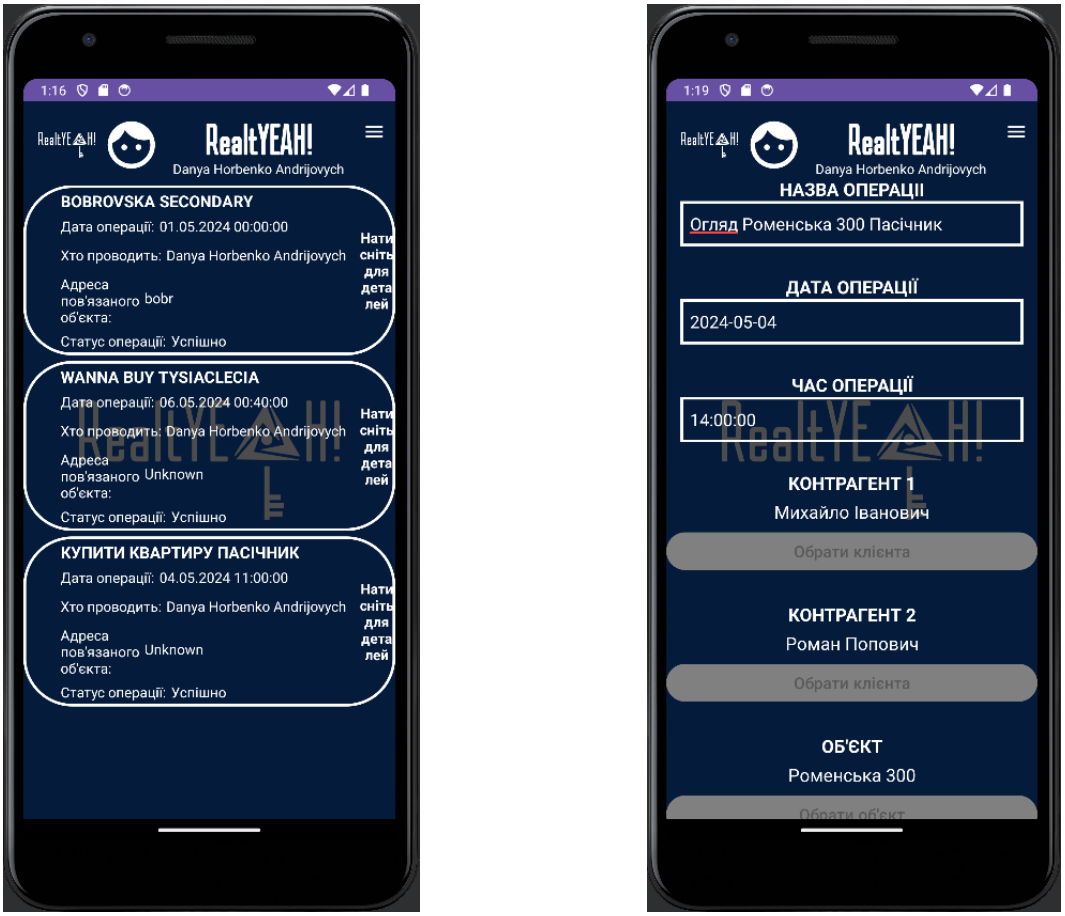


а б

Рисунок 3.31 – Наявність документів під час їх перегляду з боку об’єкта нерухомості: сторінка об’єкта (а), сторінка документів (б)

На одній з двух операцій, що містять початковий акт, з’являється кнопка «Створити наступну операцію» (див. рис. 3.26 а-б). На будь-якій з них(наведемо

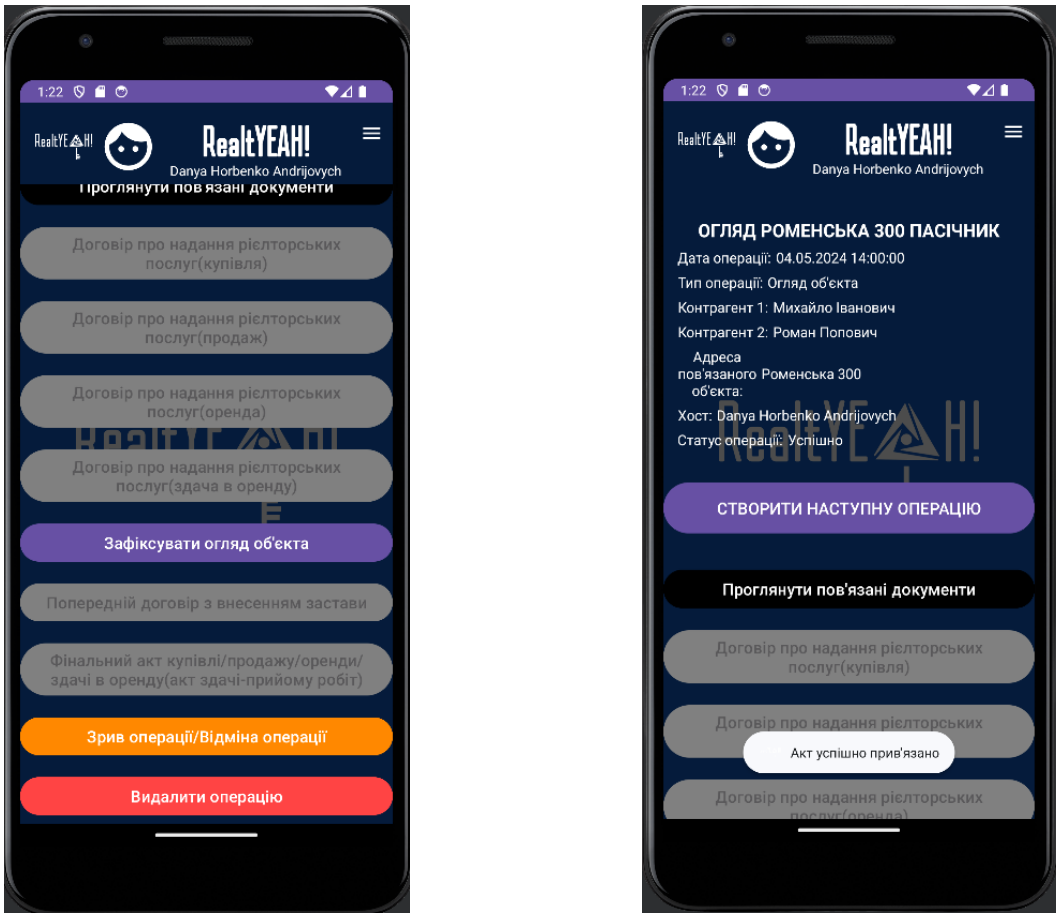
приклад з операцією «Роменська на продаж») користувач натискає дану кнопку та обирає другу операцію з початковим актом, яку треба пов’язати. Користувач обирає операцію «Купити квартиру Пасічник» (рис. 3.32 а). Система перенаправляє користувача на сторінку створення операції з заздалегідь обраними об’єктом та клієнтами (рис. 3.32 б).



а б

Рисунок 3.32 – Вибір операції для пов’язання (а) та підтягування даних з двох операцій (б)

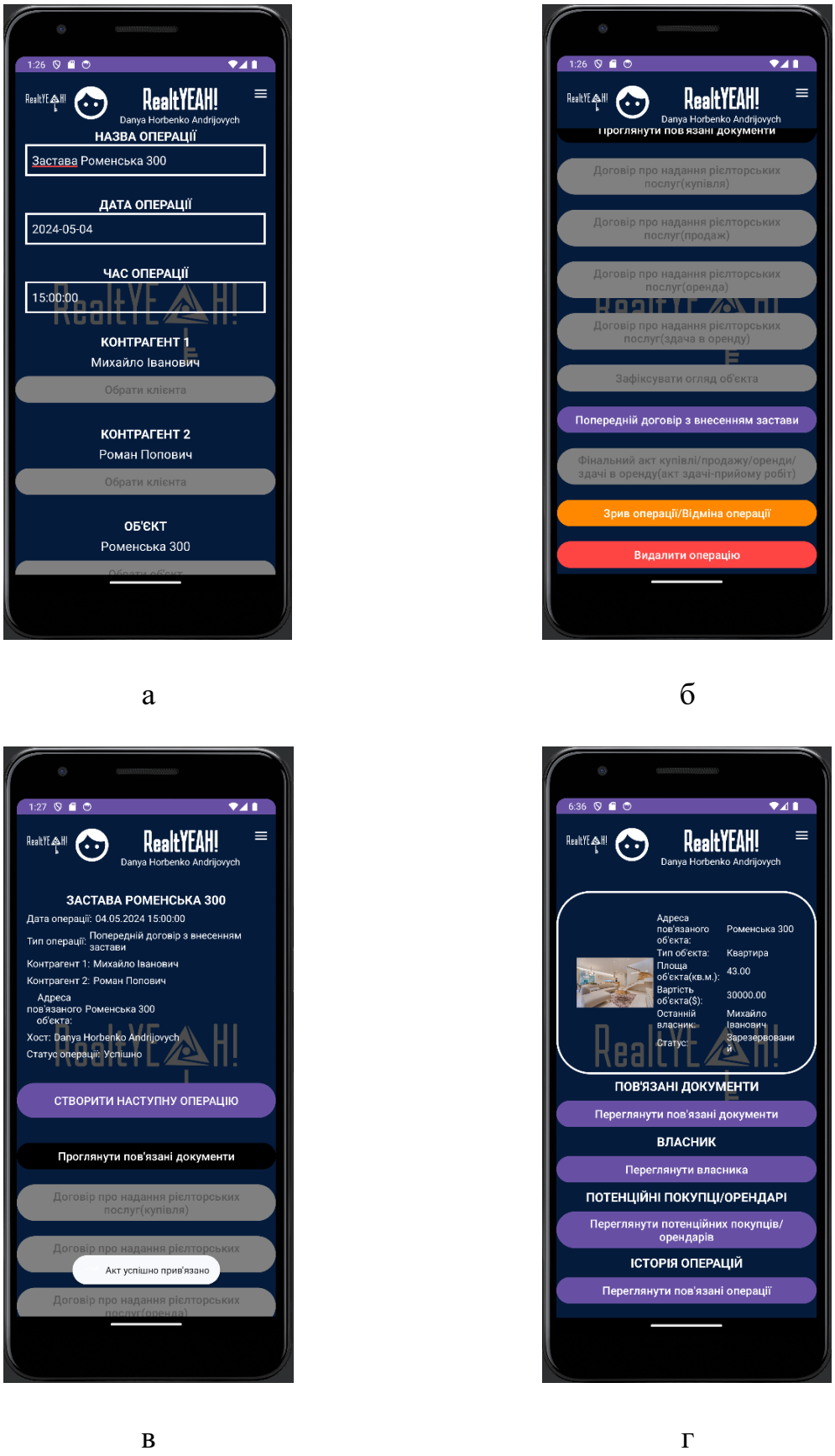
Після створення операції користувачу надається вибір актів. Так як операція була створена з двох операцій, що мають початкові акти – доступним є лише акт огляду об’єкта (рис. 3.33 а). Прикріплення даного акту дозволяє знову створити наступну операцію через кнопку «Створити наступну операцію» (рис. 3.33 б).



а б

Рисунок 3.33 – Сторінка операції «Огляд Роменська 300 Пасічник»: прикріплення акту огляду (а), кнопка створення наступної операції (б)

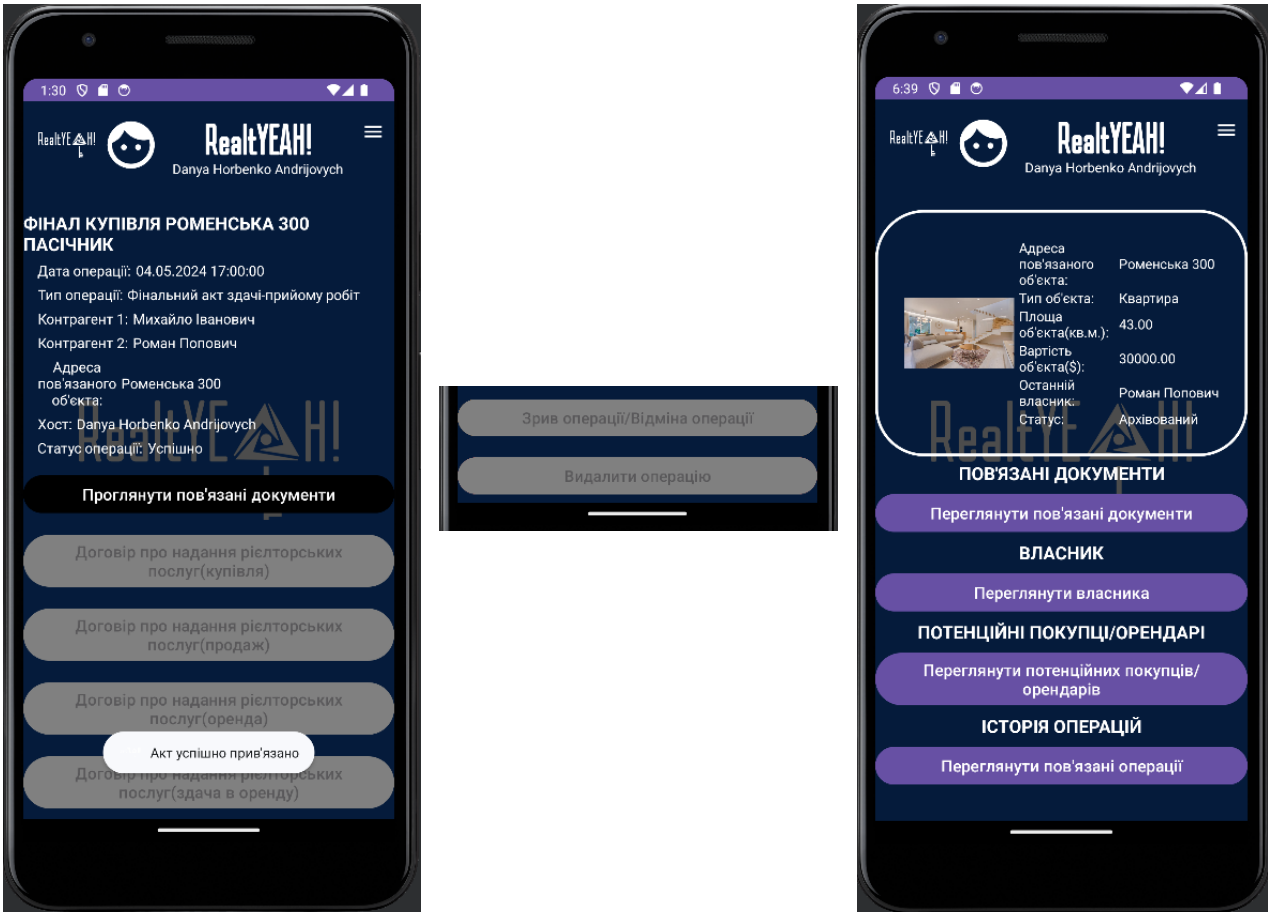
Ланцюг створення операцій продовжується аналогічним чином до фінального акту прийому-здачі послуг. Під час прикріплення акту застави об’єкту привласнюється статус «Зарезервований» (рис. 3.34).



а б в г

Рисунок 3.34 – Створення наступної операції у ланцюгу: створення операції (а), доступні акти (б), прикріплення акту застави (в), статус об’єкта після застави (г)

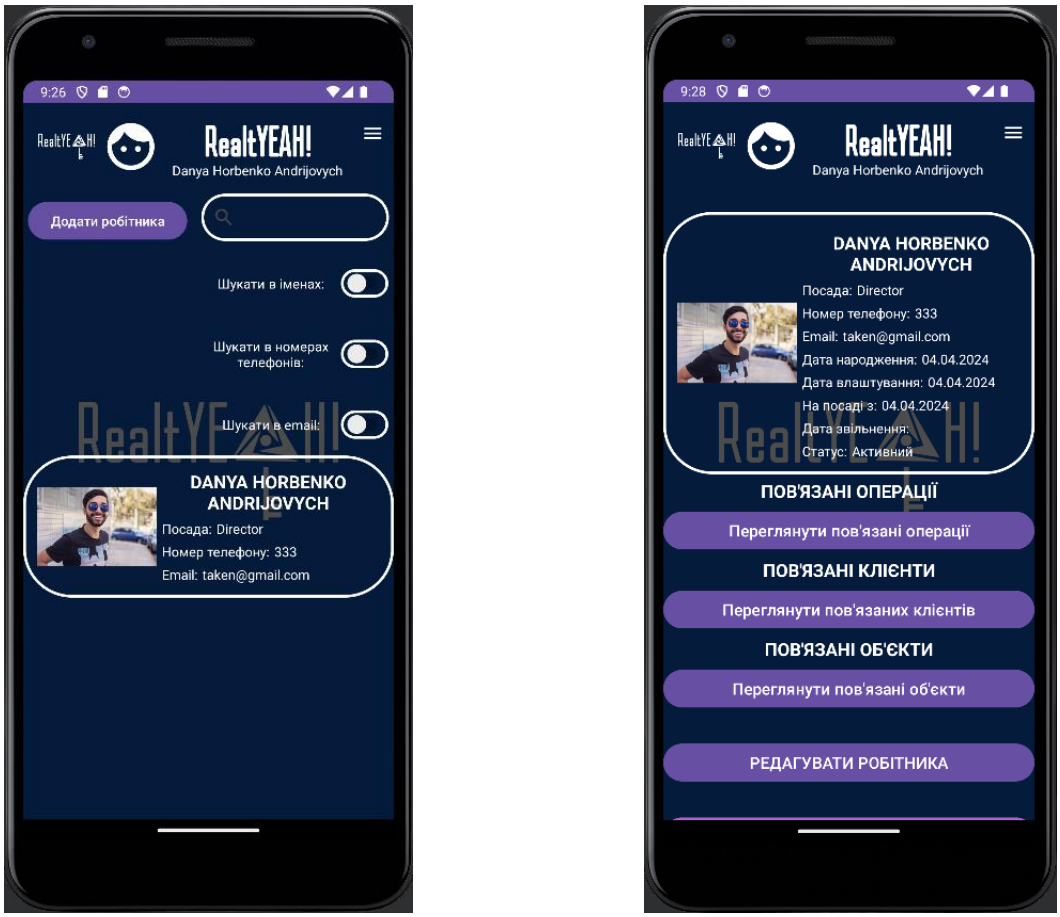
Аналогічним способом створюємо наступну операцію «Фінал купівля Роменська 300 Пасічник», яка має бути фінальною у ланцюгу. Прикріплюємо до даної операції фінальний акт здачі-прийому робіт, фіксуючи ним купівлю об’єкта нерухомості контрагентом 2 (Роман Попович) (рис. 3.35 а). Фінальний акт неможливо відмінити або видалити (рис. 3.35 б). Якщо клієнти не мають інших активних договорів – вони переміщуються в архів зі статусом «Минулий клієнт». Об’єкт переміщується в архів зі статусом «Архівований», а його власником у додатку стане відмічатися контрагент 2 (Роман Попович) (рис. 3.35 в). Архівовані клієнти та об’єкти можуть використовуватися у майбутньому, для чого необхідно створити нові початкові акти та повторити цикл створення операцій.



а б в

Рисунок 3.35 – Фіналізація ланцюга операцій: прикріплення фінального акту (а), блокування кнопок відміни та видалення (б), зміна статусу та власника об’єкта (в)

Через випадаюче меню перейдемо до розділу співробітників (рис. 3.36 а). З пошуку співробітників перейдемо до сторінки співробітника Danya Horbenko Andrijovych, котрий у даний момент є авторизованим у додатку (рис. 3.36 б).



а б

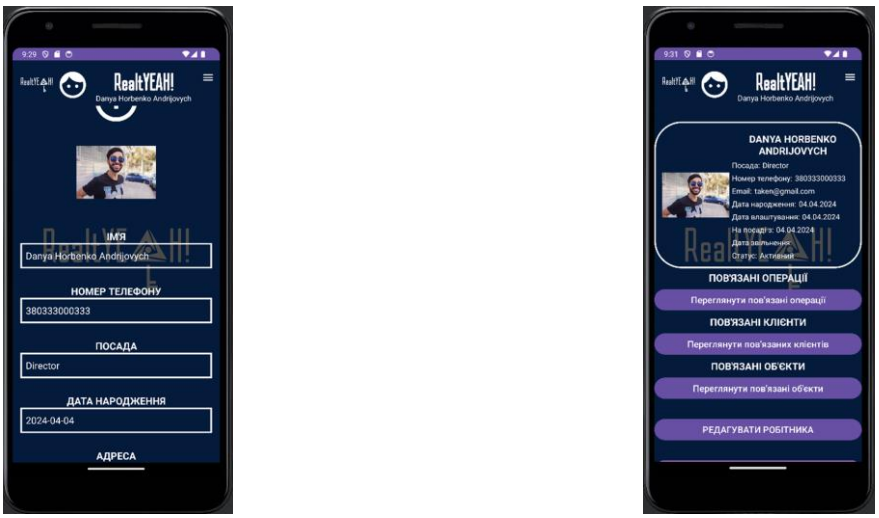
Рисунок 3.36 – Розділ співробітників: пошук співробітників (а), сторінка співробітника (б)

На сторінці співробітника присутня можливість подивитися операції (рис. 3.37 а), клієнтів (рис. 3.37 б) та об’єкти нерухомості (рис. 3.37 в), що пов’язані з даним співробітником(рієлтором). Для цього необхідно натиснути відповідні кнопки «Переглянути пов’язані операції», «Переглянути пов’язаних клієнтів» або «Переглянути пов’язані об’єкти».

а б в

Рисунок 3.37 – Сторінки пов’язаних з співробітником: операцій (а), клієнтів (б), об’єктів (в)

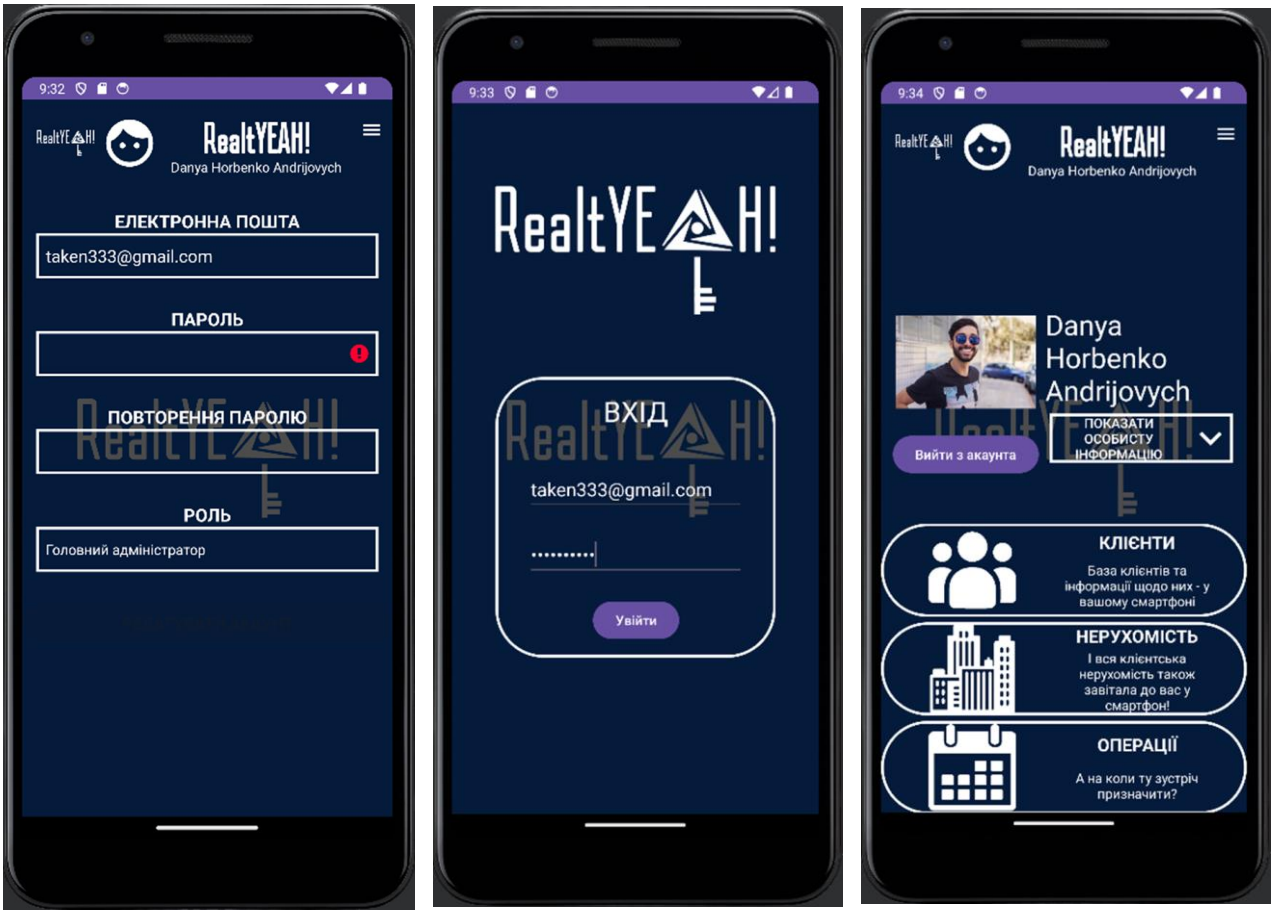
Повернемося на сторінку співробітника та відредагуємо його номер телефону. Натиснувши кнопку «Редагувати робітника», користувач з роллю адміністратора або головного адміністратора потрапляє на сторінку редагування робітника (рис. 3.38 а) та змінює номер на «380333000333» у відповідному полі, після чого натискає кнопку «Редагувати робітника». У результаті номер співробітника було змінено (рис. 3.38 б), що видно при поверненні на сторінку співробітника.



а б

Рисунок 3.38 – Редагування робітника: сторінка редагування (а), результат редагування (б)

У випадку натискання кнопки «Створити/редагувати акаунт» користувач з роллю головного адміністратора потрапляє на сторінку редагування акаунта робітника (рис. 3.39 а), де робітнику можна привласнити логін(електронна пошта), пароль та роль. Замінимо логін з «taken@gmail.com» на «taken333@gmail.com» та спробуємо заново авторизуватися у додатку з новою електронною поштою-логіном (рис. 3.39 б). Користувач потрапляє на головну сторінку додатку – авторизація є успішною (рис. 3.39 в).

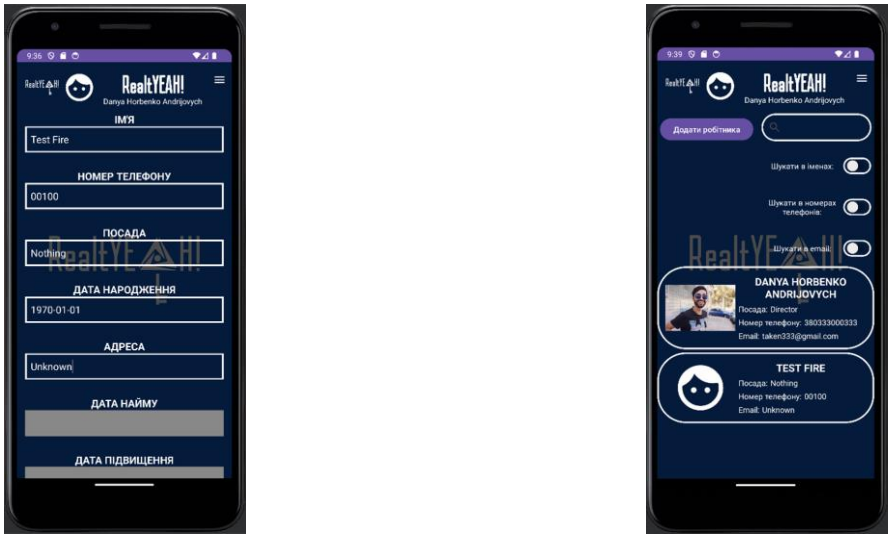


а б в

Рисунок 3.39 – Зміна даних акаунта співробітника: сторінка редагування акаунта та зміна пошти (а), авторизація з новою поштою-логіном (б), результат авторизації (в)

Користувач з роллю головного адміністратора має можливість звільнити робітника з збереженням даних, або з повним видаленням з бази даних. У першому випадку запис щодо робітника залишається у відповідній таблиці бази

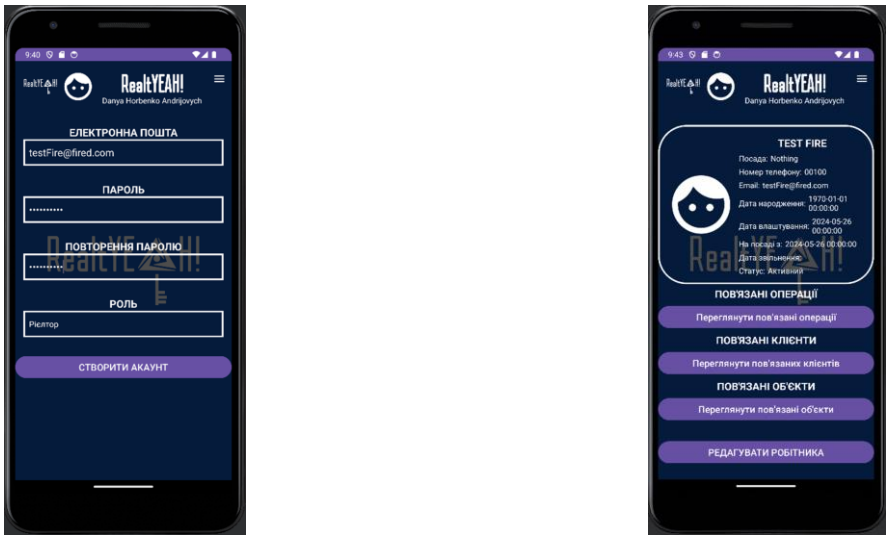
даних зі зміненим статусом на «Звільнений», у свою чергу акаунт робітника видаляється. У другому випадку й дані, й акаунт, й усі залежності каскадним принципом видаляються з бази даних. Для тестування даних можливостей скористаємося створенням нового робітника, користуючись кнопкою «Додати робітника» на сторінці пошуку робітників (див. рис. 3.36 а), після чого користувач потрапляє на сторінку створення робітника (рис. 3.40 а). Заповнивши поля тестовими даними, користувач натискає кнопку «Створити робітника» та потрапляє назад на сторінку пошуку робітників з щойно створеним робітником (рис. 3.40 б).



а б

Рисунок 3.40 – Створення робітника: сторінка створення (а), результат створення (б)

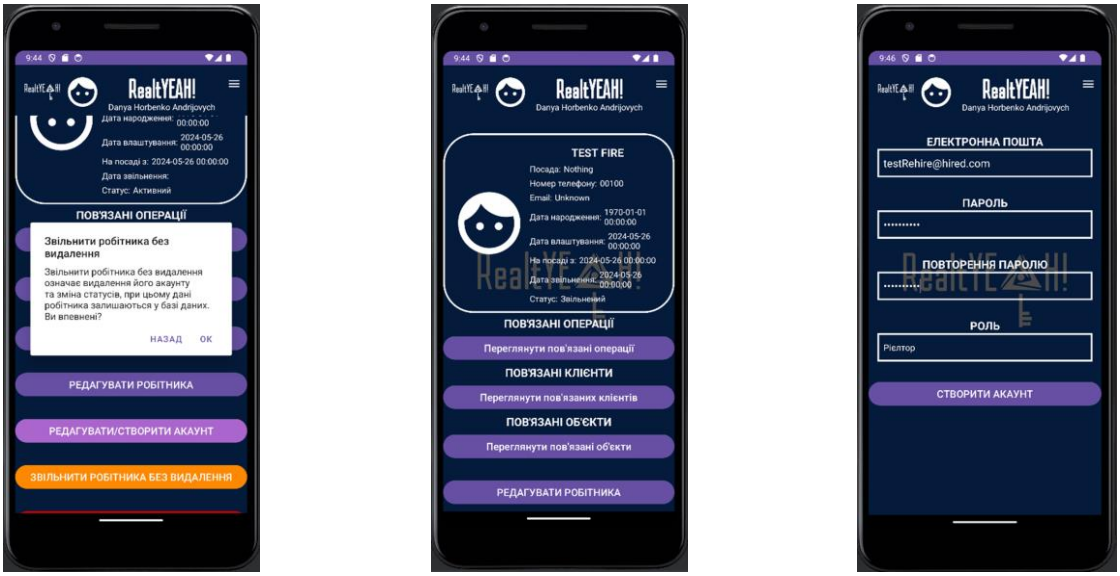
Через сторінку нового робітника перейдемо до створення його акаунту (рис. 3.41 а). Результат створення акаунта з поштою (рис. 3.41 б) дозволяє користувачу провести операції видалення робітника.



а б

Рисунок 3.41 – Створення акаунта нового робітника: введення даних акаунта (а), результат створення акаунта з поштою (б)

Звільнення робітника без видалення проводиться за натискання кнопки «Звільнити робітника без видалення» та погодження з відповідним повідомленням. У даному випадку статус робітника змінюється на «Звільнений» (рис. 3.42 а), додається дата звільнення (рис. 3.42 б), а також видаляється акаунт. Для поновлення робітника необхідно створити акаунт заново (рис. 3.42 в)



а б в

Рисунок 3.42 – Звільнення робітника без видалення: діалогове вікно з повідомленням (а), результат звільнення (б), створення нового акаунта (в)

Результатом створення нового акаунта є повернення робітника у статус активних та зникнення дати звільнення (рис. 3.43).

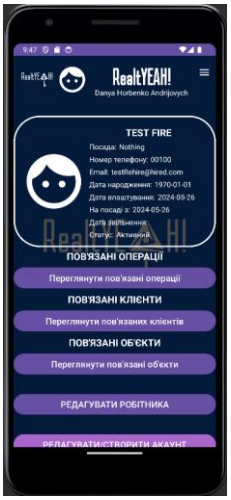
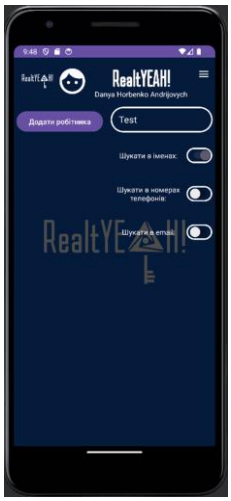


Рисунок 3.43 – Результат поновлення робітника на роботі

Звільнення робітника з видаленням означає повне видалення робітника з бази даних. Це включає його акаунт, його дані, а також всі пов’язані дитячі записи: операції, документи. Для проведення даної операції користувач натискає на кнопку «Звільнити робітника з видаленням», погоджується з повідомленням у діалоговому вікні (рис. 3.44 а). Перейшовши до сторінки пошуку робітників, можна переконатися у відсутності робітника у базі даних, виконавши пошук за частиною його імені – «test» (рис. 3.44 б).

а



б

Рисунок 3.44 – Звільнення робітника з видаленням: діалогове вікно з повідомленням (а), результат видалення (б)

ВИСНОВКИ

Під час виконання кваліфікаційної роботи бакалавра було проведено аналіз існуючих тенденцій на ринку нерухомості та виконано аналіз існуючих додатків для супроводження діяльності рієлторів. Було проведено огляд тенденцій до цифровізації та проаналізовано популярність мобільних телефонів на світовому ринку. Здійснено огляд основних засобів програмування для операційної системи Android, їх статистики використання. Розглянуто популярні реляційні середовища керування базами даних, а також сучасні архітектури серверних додатків. На основі опрацьованої інформації, було виконано наступні задачі:

Вибір засобів реалізації задач. Було здійснено вибір засобів для реалізації мобільного додатку. Розглядалися різні інструменти програмування, реляційні бази даних, та технології для створення серверної частини додатку.

Розробка структури системи мобільного додатку. Було розроблено структуру системи мобільного додатку, що включала в себе визначення основних модулів та компонентів додатку, їх взаємодію та ієрархію.

Розробка логіки роботи додатку. Проведено детальну розробку логіки роботи додатку, включаючи алгоритми для ведення обліку клієнтів та їх об’єктів нерухомості, планування та облік операцій, пов'язаних з клієнтами та об’єктами нерухомості.

Розробка структури бази даних. Розроблено структуру бази даних для зберігання інформації про клієнтів, об’єкти нерухомості, операції, співробітників та документи. Визначено необхідні таблиці, їх поля та взаємозв’язки.

Розробка проєкту інтерфейсу мобільного додатку. Розроблено проєкт інтерфейсу мобільного додатку, враховуючи зручність користування, інтуїтивність та естетичний вигляд. Були створені макети основних екранів та визначено їх функціональність.

Проєктування серверного API. Здійснено проєктування серверного API для забезпечення взаємодії між мобільним додатком та сервером. Визначено необхідні кінцеві точки, методи та формати даних для обміну інформацією.

Програмна реалізація мобільного додатку. Було здійснено програмну реалізацію мобільного додатку відповідно до розроблених проєктів. Реалізація включала всі вказані у постановці задачі можливості: ведення обліку клієнтів, об’єктів нерухомості, операцій, пов'язування документів з операціями та клієнтами, управління базою співробітників та можливість входу в додаток під своїм акаунтом.

Тестування мобільного додатку. Проведене тестування реалізації додатку довело стабільність та коректність роботи функціоналу.

Отримане інформаційне та програмне забезпечення мобільного додатку супроводження діяльності рієлторів відповідає умовам технічного завдання та задовольняє потреби рієлторів агентства нерухомості у частковому перенесенні робочого процесу з режиму офлайн у режим онлайн, що значно підвищує ефективність їхньої діяльності та зручність управління інформацією.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Кознова О. Шахрайство з нерухомістю: що треба перевіряти під час купівлі квартири | Think brave. Liga Zakon. URL: https://biz.ligazakon.net/news/202505_shakhraystvo-z-nerukhomstyu-shcho-treba-perevryati-pd-chas-kupvl-kvartiri (дата звернення: 05.05.2024).

2. Семеняка В. «Проблемна» забудова: повернення проінвестованих коштів чи боротьба за власність – PRAVO.UA. URL: https://pravo.ua/problemna-zabudova-povernennia-proinvestovanykh-koshtiv-chy-borotba-za-vlasnist/ (дата звернення: 05.05.2024).

3. Як стати рієлтором: обов'язки, зарплата - офіційний блог olx.ua. Офіційний блог OLX.ua. URL: https://blog.olx.ua/31479/hto-takij-riieltor-obov-jazki-zarplatnja/ (дата звернення: 05.05.2024).

4. Степанець П. Оприлюднено найпоширеніші схеми шахрайства на вторинному ринку: як обманюють при купівлі квартири - УНІАН. URL: https://www.unian.ua/economics/other/chastina-ugod-na-vtorinnomu-rinku-zakinchuyutsya-shahraystvom-yak-obmanyuyut-ukrajinciv-ostanni-novini-11449807.html (дата звернення: 05.05.2024).

5. Татаренко О., Свирида Р., Власюк І. Які найпоширеніші шахрайства на ринку оренди житла. Пояснює експертка. Суспільне | Новини. URL: https://suspilne.media/rivne/370699-aki-najposirenisi-sahrajstva-na-rinku-orendi-zitla-poasnue-ekspertka/ (дата звернення: 05.05.2024).

6. Як Україна під час війни стала світовим лідером із цифровізації держуправління. Головні новини з України сьогодні - Kyiv Post. URL: https://www.kyivpost.com/uk/post/22145 (дата звернення: 05.05.2024).

7. Shewale R. 20 android statistics for 2024 (market share & users). DemandSage. URL: https://www.demandsage.com/android-statistics/ (дата звернення: 05.05.2024).

8. Як керувати оголошеннями співробітників агентства нерухомості. Довідковий центр DIM.RIA.com. URL: https://help.ria.com/index.php?/Knowledgebase/Article/View/808/134#start_content (дата звернення: 05.05.2024).

9. Як приєднати нового рієлтора до реєстрації Агентства нерухомості?. Довідковий центр DIM.RIA.com. URL: https://help.ria.com/index.php?/Knowledgebase/Article/View/743/134#start_content (дата звернення: 05.05.2024).

10. Lowe D. Java All-In-One for Dummies. Wiley & Sons, Incorporated, John, 2023. 890 с.

11. Stack overflow developer survey 2023. Stack Overflow. URL: https://survey.stackoverflow.co/2023/#most-popular-technologies-language (дата звернення: 05.05.2024).

12. Böllhoff P. Kotlin vs Java: strengths, weaknesses and when to use which. K&C. URL: https://kruschecompany.com/kotlin-vs-java/ (дата звернення: 05.05.2024).

13. Учасники проектів Вікімедіа. Kotlin – вікіпедія. Вікіпедія. URL: https://uk.wikipedia.org/wiki/Kotlin (дата звернення: 05.05.2024).

14. Griffiths I. Programming C# 10. Sebastopol : O'Reilly Media, 2022. 1110 с.

15. DB-Engines Ranking - popularity ranking of database management systems. DB-Engines. URL: https://db-engines.com/en/ranking (дата звернення: 05.05.2024).

16. What is MySQL?. Oracle | Cloud Applications and Cloud Platform. URL: https://www.oracle.com/mysql/what-is-mysql/ (дата звернення: 05.05.2024).

17. Ramesh R. Oracle, MySQL, PostgreSQL, and Teradata – What's the difference?. LinkedIn. URL: https://www.linkedin.com/pulse/oracle-mysql-postgresql-teradata-whats-difference-make-data-useful/ (дата звернення: 04.05.2024).

18. PostgreSQL: about. PostgreSQL: The world's most advanced open source database. URL: https://www.postgresql.org/about/#:~:text=PostgreSQL%20comes%20with%20many%20features,source,%20PostgreSQL%20is%20highly%20extensible. (date of access: 04.05.2024).

19. Davidson T. Why use postgresql for your next project? | clean commit. Website Development and Conversion Rate Optimization for eCommerce Brands | Clean Commit. URL: https://cleancommit.io/blog/why-use-postgresql-for-your-next-project/ (дата звернення: 04.05.2024).

20. Strauss D. Getting Started with Visual Studio 2022: Learning and Implementing New Features. Apress L. P., 2022. 313 с.

21. Bodnar J. MySQL tutorial - introduction to MySQL database. ZetCode - Go, C#, Python, Java, JavaScript programming. URL: https://zetcode.com/mysql/ (дата звернення: 04.05.2024).

22. Joshi B. Beginning Database Programming Using ASP.NET Core 3: With MVC, Razor Pages, Web API, jQuery, Angular, SQL Server, and NoSQL. Apress, 2019. 481 с.

23. Byrne G. Target C#: Simple Hands-On Programming with Visual Studio 2022. Apress L. P., 2022. 1619 с.

24. Smyth N. Android Studio Flamingo Essentials. 2023. 792 с.

25. Java® Platform, Standard Edition & Java Development Kit Version 20 API Specification. docs.oracle.com. URL: https://docs.oracle.com/en/java/javase/20/docs/api/index.html (дата звернення: 04.05.2024).

ДОДАТОК А
ЛІСТИНГ ПРОГРАМИ

Повний код front-end частини додатку доступний на репозиторії GitHub (режим доступу: https://github.com/Dan-Johns-MSB/RealtYeahFront.git).

Повний код front-end частини додатку доступний на репозиторії GitHub (режим доступу: https://github.com/Dan-Johns-MSB/RealtYeahBackend.git).

А.1 Файл «activity_operations_detailed.xml»

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#105CC5"
    tools:context=".activities.OperationsActivity">

    <include
        android:id="@+id/backgroundLogo"
        layout="@layout/bg_logo"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintBottom_toBottomOf="parent"/>

    <include
        android:id="@+id/toolbarContainer"
        layout="@layout/activity_toolbar_menu"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintBottom_toTopOf="@+id/scrollViewContainer"/>

    <ScrollView
        android:id="@+id/scrollViewContainer"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:background="#B3000000"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/toolbarContainer">

        <androidx.constraintlayout.widget.ConstraintLayout
            android:id="@+id/scrollViewLayout"
            android:layout_width="match_parent"
```

```
            android:layout_height="wrap_content" >

            <TextView
                android:id="@+id/detailedOpName"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_marginTop="30dp"
                android:text="@string/unknown_value"
                android:textAllCaps="true"
                android:textColor="@color/white"
                android:textSize="19sp"
                android:textStyle="bold"

app:layout_constraintBottom_toTopOf="@+id/detailedOpInfoContainer"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintTop_toTopOf="parent" />

            <androidx.constraintlayout.widget.ConstraintLayout
                android:id="@+id/detailedOpInfoContainer"
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:paddingStart="15dp"
                app:layout_constraintTop_toBottomOf="@+id/detailedOpName"

app:layout_constraintBottom_toTopOf="@+id/createNextOperationButton"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintEnd_toEndOf="parent">

                <androidx.constraintlayout.widget.ConstraintLayout
                    android:id="@+id/detailedOpDateContainer"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_marginTop="5dp"
                    app:layout_constraintTop_toTopOf="parent"

app:layout_constraintBottom_toTopOf="@+id/detailedOpTypeContainer"
                    app:layout_constraintStart_toStartOf="parent"
                    app:layout_constraintEnd_toStartOf="parent">

                    <TextView
                        android:id="@+id/detailedOpDateTitle"
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:text="@string/detailed_op_date_title"
                        android:textAlignment="center"
                        android:textColor="@color/white"
                        android:textSize="15sp"
                        app:layout_constraintTop_toTopOf="parent"
                        app:layout_constraintBottom_toBottomOf="parent"
                        app:layout_constraintStart_toStartOf="parent"
                        />

                    <TextView
                        android:id="@+id/detailedOpDate"
                        android:layout_width="0dp"
                        android:layout_height="wrap_content"
                        android:layout_marginStart="5dp"
                        android:text="@string/unknown_value"
                        android:textColor="@color/white"
                        android:textSize="15sp"
                        app:layout_constraintBottom_toBottomOf="parent"
```

```
                        app:layout_constraintEnd_toEndOf="parent"

app:layout_constraintStart_toEndOf="@+id/detailedOpDateTitle"
                        app:layout_constraintTop_toTopOf="parent" />
                </androidx.constraintlayout.widget.ConstraintLayout>

                <androidx.constraintlayout.widget.ConstraintLayout
                    android:id="@+id/detailedOpTypeContainer"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_marginTop="5dp"

app:layout_constraintTop_toBottomOf="@+id/detailedOpDateContainer"

app:layout_constraintBottom_toTopOf="@+id/detailedOpClientContainer1"
                    app:layout_constraintStart_toStartOf="parent"
                    app:layout_constraintEnd_toStartOf="parent">

                    <TextView
                        android:id="@+id/detailedOpTypeTitle"
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:text="@string/detailed_op_type_title"
                        android:textAlignment="center"
                        android:textColor="@color/white"
                        android:textSize="15sp"
                        app:layout_constraintTop_toTopOf="parent"
                        app:layout_constraintBottom_toBottomOf="parent"
                        app:layout_constraintStart_toStartOf="parent"/>

                    <TextView
                        android:id="@+id/detailedOpType"
                        android:layout_width="0dp"
                        android:layout_height="wrap_content"
                        android:text="@string/unknown_value"
                        android:textColor="@color/white"
                        android:textSize="15sp"
                        android:layout_marginStart="5dp"
                        app:layout_constraintTop_toTopOf="parent"
                        app:layout_constraintBottom_toBottomOf="parent"

app:layout_constraintStart_toEndOf="@+id/detailedOpTypeTitle"
                        app:layout_constraintEnd_toEndOf="parent"/>
                </androidx.constraintlayout.widget.ConstraintLayout>

                <androidx.constraintlayout.widget.ConstraintLayout
                    android:id="@+id/detailedOpClientContainer1"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_marginTop="5dp"

app:layout_constraintTop_toBottomOf="@+id/detailedOpTypeContainer"

app:layout_constraintBottom_toTopOf="@+id/detailedOpClientContainer2"
                    app:layout_constraintStart_toStartOf="parent"
                    app:layout_constraintEnd_toStartOf="parent">

                    <TextView
                        android:id="@+id/detailedOpClientTitle1"
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:text="@string/detailed_op_client_title_1"
```

```
                        android:textAlignment="center"
                        android:textColor="@color/white"
                        android:textSize="15sp"
                        app:layout_constraintTop_toTopOf="parent"
                        app:layout_constraintBottom_toBottomOf="parent"
                        app:layout_constraintStart_toStartOf="parent"/>

                    <TextView
                        android:id="@+id/detailedOpClient1"
                        android:layout_width="0dp"
                        android:layout_height="wrap_content"
                        android:layout_marginStart="5dp"
                        android:text="@string/unknown_value"
                        android:textColor="@color/white"
                        android:textSize="15sp"
                        app:layout_constraintTop_toTopOf="parent"
                        app:layout_constraintBottom_toBottomOf="parent"

app:layout_constraintStart_toEndOf="@+id/detailedOpClientTitle1"
                        app:layout_constraintEnd_toEndOf="parent"/>
                </androidx.constraintlayout.widget.ConstraintLayout>

                <androidx.constraintlayout.widget.ConstraintLayout
                    android:id="@+id/detailedOpClientContainer2"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_marginTop="5dp"

app:layout_constraintTop_toBottomOf="@+id/detailedOpClientContainer1"

app:layout_constraintBottom_toTopOf="@+id/detailedOpObjectContainer"
                    app:layout_constraintStart_toStartOf="parent"
                    app:layout_constraintEnd_toStartOf="parent">

                    <TextView
                        android:id="@+id/detailedOpClientTitle2"
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:text="@string/detailed_op_client_title_2"
                        android:textAlignment="center"
                        android:textColor="@color/white"
                        android:textSize="15sp"
                        app:layout_constraintTop_toTopOf="parent"
                        app:layout_constraintBottom_toBottomOf="parent"
                        app:layout_constraintStart_toStartOf="parent"/>

                    <TextView
                        android:id="@+id/detailedOpClient2"
                        android:layout_width="0dp"
                        android:layout_height="wrap_content"
                        android:layout_marginStart="5dp"
                        android:text="@string/unknown_value"
                        android:textColor="@color/white"
                        android:textSize="15sp"
                        app:layout_constraintTop_toTopOf="parent"
                        app:layout_constraintBottom_toBottomOf="parent"

app:layout_constraintStart_toEndOf="@+id/detailedOpClientTitle2"
                        app:layout_constraintEnd_toEndOf="parent"/>
                </androidx.constraintlayout.widget.ConstraintLayout>

                <androidx.constraintlayout.widget.ConstraintLayout
```

```
                    android:id="@+id/detailedOpObjectContainer"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_marginTop="5dp"

app:layout_constraintTop_toBottomOf="@+id/detailedOpClientContainer2"

app:layout_constraintBottom_toTopOf="@+id/detailedOpRealtorContainer"
                    app:layout_constraintStart_toStartOf="parent"
                    app:layout_constraintEnd_toStartOf="parent">

                    <TextView
                        android:id="@+id/detailedOpObjectTitle"
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:text="@string/detailed_op_object_title"
                        android:textAlignment="center"
                        android:textColor="@color/white"
                        android:textSize="15sp"
                        app:layout_constraintTop_toTopOf="parent"
                        app:layout_constraintBottom_toBottomOf="parent"
                        app:layout_constraintStart_toStartOf="parent"/>

                    <TextView
                        android:id="@+id/detailedOpObject"
                        android:layout_width="0dp"
                        android:layout_height="wrap_content"
                        android:layout_marginStart="5dp"
                        android:text="@string/unknown_value"
                        android:textColor="@color/white"
                        android:textSize="15sp"
                        app:layout_constraintBottom_toBottomOf="parent"

app:layout_constraintStart_toEndOf="@+id/detailedOpObjectTitle"
                        app:layout_constraintTop_toTopOf="parent"
                        app:layout_constraintEnd_toEndOf="parent"/>
                </androidx.constraintlayout.widget.ConstraintLayout>

                <androidx.constraintlayout.widget.ConstraintLayout
                    android:id="@+id/detailedOpRealtorContainer"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_marginTop="5dp"

app:layout_constraintTop_toBottomOf="@+id/detailedOpObjectContainer"

app:layout_constraintBottom_toTopOf="@+id/detailedOpStatusContainer"
                    app:layout_constraintStart_toStartOf="parent"
                    app:layout_constraintEnd_toStartOf="parent">

                    <TextView
                        android:id="@+id/detailedOpRealtorTitle"
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:text="@string/detailed_op_realtor_title"
                        android:textAlignment="center"
                        android:textColor="@color/white"
                        android:textSize="15sp"
                        app:layout_constraintTop_toTopOf="parent"
                        app:layout_constraintBottom_toBottomOf="parent"
                        app:layout_constraintStart_toStartOf="parent"/>
```

```
                    <TextView
                        android:id="@+id/detailedOpRealtor"
                        android:layout_width="0dp"
                        android:layout_height="wrap_content"
                        android:layout_marginStart="5dp"
                        android:text="@string/unknown_value"
                        android:textColor="@color/white"
                        android:textSize="15sp"
                        app:layout_constraintBottom_toBottomOf="parent"
                        app:layout_constraintEnd_toEndOf="parent"

app:layout_constraintStart_toEndOf="@+id/detailedOpRealtorTitle"
                        app:layout_constraintTop_toTopOf="parent" />
                </androidx.constraintlayout.widget.ConstraintLayout>

                <androidx.constraintlayout.widget.ConstraintLayout
                    android:id="@+id/detailedOpStatusContainer"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_marginTop="5dp"

app:layout_constraintTop_toBottomOf="@+id/detailedOpRealtorContainer"
                    app:layout_constraintBottom_toBottomOf="parent"
                    app:layout_constraintStart_toStartOf="parent"
                    app:layout_constraintEnd_toStartOf="parent">

                    <TextView
                        android:id="@+id/detailedOpStatusTitle"
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:text="@string/detailed_op_status_title"
                        android:textAlignment="center"
                        android:textColor="@color/white"
                        android:textSize="15sp"
                        app:layout_constraintTop_toTopOf="parent"
                        app:layout_constraintBottom_toBottomOf="parent"
                        app:layout_constraintStart_toStartOf="parent"/>

                    <TextView
                        android:id="@+id/detailedOpStatus"
                        android:layout_width="0dp"
                        android:layout_height="wrap_content"
                        android:layout_marginStart="5dp"
                        android:text="@string/operation_status_wait"
                        android:textColor="@color/white"
                        android:textSize="15sp"
                        app:layout_constraintTop_toTopOf="parent"
                        app:layout_constraintBottom_toBottomOf="parent"

app:layout_constraintStart_toEndOf="@+id/detailedOpStatusTitle"
                        app:layout_constraintEnd_toEndOf="parent"/>
                </androidx.constraintlayout.widget.ConstraintLayout>
            </androidx.constraintlayout.widget.ConstraintLayout>

            <Button
                android:id="@+id/createNextOperationButton"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginTop="30dp"
                android:layout_marginBottom="15dp"
                android:padding="15dp"
                android:text="@string/operations_create_next_button"
```

```
                android:textAllCaps="true"
                android:textSize="17sp"
                android:visibility="gone"
                android:enabled="false"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintEnd_toEndOf="parent"

app:layout_constraintTop_toBottomOf="@+id/detailedOpInfoContainer"
                app:layout_constraintBottom_toTopOf="@+id/watchDocsOpButton"/>

            <Button
                android:id="@+id/watchDocsOpButton"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginTop="15dp"
                android:backgroundTint="@android:color/black"
                android:text="@string/operations_watch_docs_button"
                android:textSize="17sp"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintEnd_toEndOf="parent"

app:layout_constraintTop_toBottomOf="@+id/createNextOperationButton"
                app:layout_constraintBottom_toTopOf="@+id/beginBuyOpButton"/>

            <Button
                android:id="@+id/beginBuyOpButton"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginTop="15dp"
                android:text="@string/operations_begin_buy_button"
                android:textSize="17sp"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintTop_toBottomOf="@+id/watchDocsOpButton"
                app:layout_constraintBottom_toTopOf="@+id/beginSellOpButton"/>

            <Button
                android:id="@+id/beginSellOpButton"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginTop="15dp"
                android:text="@string/operations_begin_sell_button"
                android:textSize="17sp"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintTop_toBottomOf="@+id/beginBuyOpButton"
                app:layout_constraintBottom_toTopOf="@+id/beginRentOpButton"/>

            <Button
                android:id="@+id/beginRentOpButton"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginTop="15dp"
                android:text="@string/operations_begin_rent_button"
                android:textSize="17sp"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintTop_toBottomOf="@+id/beginSellOpButton"

app:layout_constraintBottom_toTopOf="@+id/beginForRentOpButton"/>

            <Button
```

```
        android:id="@+id/beginForRentOpButton"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="15dp"
        android:text="@string/operations_begin_for_rent_button"
        android:textSize="17sp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/beginRentOpButton"
        app:layout_constraintBottom_toTopOf="@+id/objectCheckOpButton"/>

    <Button
        android:id="@+id/objectCheckOpButton"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="15dp"
        android:text="@string/operations_object_check_button"
        android:textSize="17sp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/beginForRentOpButton"
        app:layout_constraintBottom_toTopOf="@+id/pledgeOpButton"/>

    <Button
        android:id="@+id/pledgeOpButton"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="15dp"
        android:text="@string/operations_pledge_button"
        android:textSize="17sp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/objectCheckOpButton"
        app:layout_constraintBottom_toTopOf="@+id/finalOpButton"/>

    <Button
        android:id="@+id/finalOpButton"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="15dp"
        android:text="@string/operations_final_button"
        android:textSize="17sp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/pledgeOpButton"
        app:layout_constraintBottom_toTopOf="@+id/cancelOpButton"/>

    <Button
        android:id="@+id/cancelOpButton"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="15dp"
        android:backgroundTint="@android:color/holo_orange_dark"
        android:text="@string/operations_cancel_op_button"
        android:textSize="17sp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/finalOpButton"
        app:layout_constraintBottom_toTopOf="@+id/deleteOpButton"/>

    <Button
        android:id="@+id/deleteOpButton"
```

```
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginTop="15dp"
                android:layout_marginBottom="5dp"
                android:backgroundTint="@android:color/holo_red_light"
                android:text="@string/operations_delete_op_button"
                android:textSize="17sp"
                app:layout_constraintBottom_toBottomOf="parent"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintTop_toBottomOf="@+id/cancelOpButton" />
        </androidx.constraintlayout.widget.ConstraintLayout>
    </ScrollView>
</androidx.constraintlayout.widget.ConstraintLayout>
```

A.2 Файл «OperationsDetailedActivity.java»

```
package com.example.prethesispractice.activities;

import android.app.AlertDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

import com.example.prethesispractice.R;
import com.example.prethesispractice.entities.Client;
import com.example.prethesispractice.entities.Employee;
import com.example.prethesispractice.entities.EstateObject;
import com.example.prethesispractice.entities.Operation;
import com.example.prethesispractice.models.constants.ActTypesConst;
import com.google.gson.Gson;

import java.io.IOException;
import java.util.ArrayList;
import java.util.List;

import okhttp3.ResponseBody;
import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;

public class OperationsDetailedActivity extends ToolbarMenuActivity {
    private TextView operationName;
    private TextView operationDate;
    private TextView operationType;
    private TextView firstClientView;
    private TextView secondClientView;
    private TextView objectAddress;
    private TextView hostName;
    private TextView operationStatus;
    private Button createNextButton;
    private Button watchDocsButton;
    private Operation operation;
    private Client firstClient;
    private Client secondClient;
    private EstateObject estateObject;
```

```
    private Employee host;
    private ArrayList<Boolean> availableActs;
    private ArrayList<Button> actButtons;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_operations_detailed);
        inflateToolbar(null);

        operationName = findViewById(R.id.detailedOpName);
        operationDate = findViewById(R.id.detailedOpDate);
        operationType = findViewById(R.id.detailedOpType);
        firstClientView = findViewById(R.id.detailedOpClient1);
        secondClientView = findViewById(R.id.detailedOpClient2);
        objectAddress = findViewById(R.id.detailedOpObject);
        hostName = findViewById(R.id.detailedOpRealtor);
        operationStatus = findViewById(R.id.detailedOpStatus);

        String chosenOperation = getIntent().getStringExtra("chosen_operation");
        if (chosenOperation == null || chosenOperation.isBlank()) {
            Toast.makeText(getApplicationContext(), "No operation data to
display", Toast.LENGTH_LONG).show();
            finish();
        }

        Gson jsonConverter = new Gson();
        operation = jsonConverter.fromJson(chosenOperation, Operation.class);

        final String token = getIntent().getStringExtra("token");
        displayOperationInfo(token);

        createNextButton = findViewById(R.id.createNextOperationButton);
        getNextAvailability(token);

        actButtons = new ArrayList<Button>();
        actButtons.add(findViewById(R.id.beginBuyOpButton));
        actButtons.add(findViewById(R.id.beginSellOpButton));
        actButtons.add(findViewById(R.id.beginRentOpButton));
        actButtons.add(findViewById(R.id.beginForRentOpButton));
        actButtons.add(findViewById(R.id.objectCheckOpButton));
        actButtons.add(findViewById(R.id.pledgeOpButton));
        actButtons.add(findViewById(R.id.finalOpButton));
        actButtons.add(findViewById(R.id.cancelOpButton));
        actButtons.add(findViewById(R.id.deleteOpButton));

        getAvailableActs(token);

        firstClientView.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                clientClick(firstClient);
            }
        });
        secondClientView.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                clientClick(secondClient);
            }
        });
        objectAddress.setOnClickListener(new View.OnClickListener() {
            @Override
```

```
            public void onClick(View view) {
                objectClick();
            }
        });
        hostName.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                //hostClick();
            }
        });

        actButtons.get(0).setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                moveToAssignAct(ActTypesConst.BUY_AGENT_DEAL);
            }
        });
        actButtons.get(1).setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                moveToAssignAct(ActTypesConst.SELL_AGENT_DEAL);
            }
        });
        actButtons.get(2).setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                moveToAssignAct(ActTypesConst.RENT_AGENT_DEAL);
            }
        });
        actButtons.get(3).setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                moveToAssignAct(ActTypesConst.FOR_RENT_AGENT_DEAL);
            }
        });
        actButtons.get(4).setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                moveToAssignAct(ActTypesConst.OBJECT_CHECK);
            }
        });
        actButtons.get(5).setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                moveToAssignAct(ActTypesConst.PLEDGE);
            }
        });
        actButtons.get(6).setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                moveToAssignAct(ActTypesConst.FINAL_DEAL);
            }
        });
        actButtons.get(7).setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                AlertDialog.Builder forCancel = new
AlertDialog.Builder(OperationsDetailedActivity.this);
                forCancel.setTitle(R.string.operations_cancel_op_button);
                forCancel.setMessage(R.string.cancel_op_message);
                forCancel.setPositiveButton(R.string.alert_dialog_ok, new
DialogInterface.OnClickListener() {
```

```
                    public void onClick(DialogInterface dialog, int id) {
                        httpApi.cancelOperation("Bearer " + token,
operation.getOperationId()).enqueue(new Callback<Operation>() {
                            @Override
                            public void onResponse(Call<Operation> call,
Response<Operation> response) {
                                if (response.isSuccessful()) {
                                    Toast.makeText(getApplicationContext(),
"Операцію успішно відмінено/зірвано!", Toast.LENGTH_LONG).show();
                                    operation = response.body();

                                    getAvailableActs(token);
                                    displayOperationInfo(token);
                                    getNextAvailability(token);
                                } else {
                                    try {
                                        Toast.makeText(getApplicationContext(),
"Сталася помилка під час відміни операції("
                                                + response.errorBody().string()
+ ")", Toast.LENGTH_LONG).show();
                                    } catch (IOException e) {
                                        Toast.makeText(getApplicationContext(),
"Сталася помилка під час відміни операції", Toast.LENGTH_LONG).show();
                                    }
                                }
                            }

                            @Override
                            public void onFailure(Call<Operation> call,
Throwable throwable) {
                                Toast.makeText(getApplicationContext(), "Сталася
помилка під час відміни операції("
                                        + throwable.getMessage() + ")",
Toast.LENGTH_LONG).show();
                            }
                        });
                    }
                });
                forCancel.setNegativeButton(R.string.alert_dialog_goBack, new
DialogInterface.OnClickListener() {
                    public void onClick(DialogInterface dialog, int id) {
                        dialog.dismiss();
                    }
                });

                AlertDialog alertDialog = forCancel.create();
                alertDialog.show();
            }
        });
        actButtons.get(8).setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                AlertDialog.Builder forDelete = new
AlertDialog.Builder(OperationsDetailedActivity.this);
                forDelete.setTitle(R.string.operations_delete_op_button);
                forDelete.setMessage(R.string.delete_op_message);
                forDelete.setPositiveButton(R.string.alert_dialog_ok, new
DialogInterface.OnClickListener() {
                    public void onClick(DialogInterface dialog, int id) {
                        httpApi.deleteOperation("Bearer " + token,
operation.getOperationId()).enqueue(new Callback<ResponseBody>() {
                            @Override
```

```
                            public void onResponse(Call<ResponseBody> call,
Response<ResponseBody> response) {
                                if (response.isSuccessful()) {
                                    Toast.makeText(getApplicationContext(),
"Операцію успішно видалено!", Toast.LENGTH_LONG).show();
                                } else {
                                    try {
                                        Toast.makeText(getApplicationContext(),
"Сталася помилка під час видалення операції("
                                                + response.errorBody().string()
+ ")", Toast.LENGTH_LONG).show();
                                    } catch (IOException e) {
                                        Toast.makeText(getApplicationContext(),
"Сталася помилка під час видалення операції", Toast.LENGTH_LONG).show();
                                    }
                                }
                            }

                            @Override
                            public void onFailure(Call<ResponseBody> call,
Throwable throwable) {
                                Toast.makeText(getApplicationContext(), "Сталася
помилка під час видалення операції("
                                        + throwable.getMessage() + ")",
Toast.LENGTH_LONG).show();
                            }
                        });
                    }
                });
                forDelete.setNegativeButton(R.string.alert_dialog_goBack, new
DialogInterface.OnClickListener() {
                    public void onClick(DialogInterface dialog, int id) {
                        dialog.dismiss();
                    }
                });

                AlertDialog alertDialog = forDelete.create();
                alertDialog.show();
            }
        });

        watchDocsButton = findViewById(R.id.watchDocsOpButton);
        watchDocsButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                basicActivitySwitch(WatchDocumentsActivity.class);
            }
        });
    }

    @Override
    protected void onStart() {
        super.onStart();
    }

    @Override
    protected void onRestart() {
        super.onRestart();
    }

    @Override
    protected void onResume() {
```

```
        super.onResume();
    }

    @Override
    protected void onPause() {
        super.onPause();
    }

    @Override
    protected void onStop() {
        super.onStop();
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
    }

    private void getAvailableActs(String token) {
        httpApi.getAvailableActs("Bearer " + token,
operation.getOperationId()).enqueue(new Callback<List<Boolean>>() {
            @Override
            public void onResponse(Call<List<Boolean>> call,
Response<List<Boolean>> response) {
                if (response.isSuccessful()) {
                    availableActs = new ArrayList<Boolean>(response.body());

                    for (Button actButton : actButtons) {

actButton.setEnabled(availableActs.get(actButtons.indexOf(actButton)));

                        if (!availableActs.get(actButtons.indexOf(actButton))) {

actButton.setBackgroundTintList(getApplicationContext().getColorStateList(R.colo
r.gray));
                        }
                    }
                }
            }

            @Override
            public void onFailure(Call<List<Boolean>> call, Throwable t) {
                Toast.makeText(getApplicationContext(), t.getMessage(),
Toast.LENGTH_LONG).show();
            }
        });
    }

    private void clientClick(Client client) {
        Intent moveToClient = new Intent(getApplicationContext(),
ClientsDetailedActivity.class);

        Gson jsonConverter = new Gson();
        String chosenClient = jsonConverter.toJson(client);
        moveToClient.putExtra("chosen_client", chosenClient);

        passEmployeeData(moveToClient);
    }

    private void objectClick() {
        if (estateObject != null) {
```

```
            Intent moveToObject = new Intent(getApplicationContext(),
ObjectsDetailedActivity.class);

            Gson jsonConverter = new Gson();
            String chosenObject = jsonConverter.toJson(estateObject);
            moveToObject.putExtra("chosen_object", chosenObject);

            passEmployeeData(moveToObject);
        }
    }

    /*private void hostClick() {
        Intent moveToHost = new Intent(getApplicationContext(),
EmployeesDetailedActivity.class);

        Gson jsonConverter = new Gson();
        String chosenHost = jsonConverter.toJson(host);
        moveToHost.putExtra("chosen_employee", chosenHost);

        passEmployeeData(moveToHost);
    }*/

    private void displayOperationInfo(String token) {
        if (operation == null) {
            Toast.makeText(getApplicationContext(), "No operation data to
display", Toast.LENGTH_LONG).show();
            finish();
        }

        operationName.setText(operation.getName());
        operationDate.setText(operation.getDateFormatted());
        operationType.setText(operation.getActType());
        operationStatus.setText(operation.getStatus());

        httpApi.getClientById("Bearer " + token,
operation.getCounteragentLead())
            .enqueue(new Callback<Client>() {
                @Override
                public void onResponse(Call<Client> call, Response<Client>
response) {
                    if (response.isSuccessful()) {
                        firstClient = response.body();
                        firstClientView.setText(firstClient.getName());
                    }
                }

                @Override
                public void onFailure(Call<Client> call, Throwable t) {
                    Toast.makeText(getApplicationContext(), t.getMessage(),
Toast.LENGTH_LONG).show();
                }
            });

        if (operation.getCounteragentSecondary() == null) {
            secondClientView.setText(R.string.unknown_value);
        } else {
            httpApi.getClientById("Bearer " + token,
operation.getCounteragentSecondary())
                .enqueue(new Callback<Client>() {
                    @Override
                    public void onResponse(Call<Client> call, Response<Client>
response) {
```

```
                        if (response.isSuccessful()) {
                            secondClient = response.body();
                            secondClientView.setText(secondClient.getName());
                        }
                    }

                    @Override
                    public void onFailure(Call<Client> call, Throwable t) {
                        Toast.makeText(getApplicationContext(), t.getMessage(),
Toast.LENGTH_LONG).show();
                    }
                });
        }

        if (operation.getEstateObjectId() == null) {
            objectAddress.setText(R.string.unknown_value);
        } else {
            httpApi.getEstateObjectById("Bearer " + token,
operation.getEstateObjectId())
                .enqueue(new Callback<EstateObject>() {
                    @Override
                    public void onResponse(Call<EstateObject> call,
Response<EstateObject> response) {
                        if (response.isSuccessful()) {
                            estateObject = response.body();
                            objectAddress.setText(estateObject.getAddress());
                        }
                    }

                    @Override
                    public void onFailure(Call<EstateObject> call, Throwable t)
{
                        Toast.makeText(getApplicationContext(), t.getMessage(),
Toast.LENGTH_LONG).show();
                    }
                });
        }

        httpApi.getHostByOperation("Bearer " + token,
operation.getOperationId()).enqueue(new Callback<Employee>() {
            @Override
            public void onResponse(Call<Employee> call, Response<Employee>
response) {
                if (response.isSuccessful()) {
                    host = response.body();
                    hostName.setText(host.getName());
                }
            }

            @Override
            public void onFailure(Call<Employee> call, Throwable t) {
                Toast.makeText(getApplicationContext(), t.getMessage(),
Toast.LENGTH_LONG).show();
            }
        });
    }

    private void moveToAssignAct(String actType) {
        /*Intent moveToApplyDocuments = new Intent(getApplicationContext(),
AssignActActivity.class);
        moveToApplyDocuments.putExtra("act_type", actType);
```

```
        passEmployeeData(moveToApplyDocuments);*/

        final String token = getIntent().getStringExtra("token");

        httpApi.assignAct("Bearer " + token, operation.getOperationId(),
actType).enqueue(new Callback<Operation>() {
            @Override
            public void onResponse(Call<Operation> call, Response<Operation>
response) {
                if (response.isSuccessful()) {
                    Toast.makeText(getApplicationContext(), "Акт успішно
прив'язано", Toast.LENGTH_LONG).show();
                    operation.setActType(actType);
                    operation.setStatus("Успішно");

                    getAvailableActs(token);
                    displayOperationInfo(token);
                    getNextAvailability(token);
                } else {
                    try {
                        Toast.makeText(getApplicationContext(), "Акт неможливо
прив'язати(" + response.errorBody().string() + ")", Toast.LENGTH_LONG).show();
                    } catch (IOException e) {
                        throw new RuntimeException(e);
                    }
                }
            }

            @Override
            public void onFailure(Call<Operation> call, Throwable t) {
                Toast.makeText(getApplicationContext(), "Помилка під час
прив'язування акту(" + t.getMessage() + ")", Toast.LENGTH_LONG).show();
            }
        });
    }

    private void getNextAvailability(String token) {
        httpApi.getNextAvailability("Bearer " + token,
operation.getOperationId()).enqueue(new Callback<Boolean>() {
            @Override
            public void onResponse(Call<Boolean> call, Response<Boolean>
response) {
                if (response.isSuccessful() && response.body()) {
                    createNextButton.setOnClickListener(new
View.OnClickListener() {
                        @Override
                        public void onClick(View view) {
                            Gson jsonConverter = new Gson();

                            if (operation.getCounteragentSecondary() != null) {
                                Intent toCreateChainOperation = new
Intent(getApplicationContext(), CreateOperationActivity.class);
                                toCreateChainOperation.putExtra("prev_lead",
jsonConverter.toJson(operation));

toCreateChainOperation.putExtra("prev_secondary",
jsonConverter.toJson(operation));

                                passEmployeeData(toCreateChainOperation);
                            } else {
                                Intent toPickSecondOperation = new
Intent(getApplicationContext(), OperationsListActivity.class);
```

```
                                toPickSecondOperation.putExtra("prev_lead",
jsonConverter.toJson(operation));

                                passEmployeeData(toPickSecondOperation);
                            }
                        }
                    });

                    createNextButton.setEnabled(true);
                    createNextButton.setVisibility(Button.VISIBLE);
                } else {
                    createNextButton.setEnabled(false);
                    createNextButton.setVisibility(Button.GONE);
                    Toast.makeText(getApplicationContext(), "Не вдалося
визначити можливість використання операції у ланцюгу",
Toast.LENGTH_LONG).show();
                }
            }

            @Override
            public void onFailure(Call<Boolean> call, Throwable throwable) {
                Toast.makeText(getApplicationContext(), "Не вдалося визначити
можливість використання операції у ланцюгу("
                        + throwable.getMessage() + ")",
Toast.LENGTH_LONG).show();
            }
        });
    }
}
```

A.3 Файл «activity_create_operation.xml»

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#105CC5"
    tools:context=".activities.OperationsActivity">

    <include
        android:id="@+id/backgroundLogo"
        layout="@layout/bg_logo"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintBottom_toBottomOf="parent"/>

    <include
        android:id="@+id/toolbarContainer"
        layout="@layout/activity_toolbar_menu"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toTopOf="parent"
```

```
        app:layout_constraintBottom_toTopOf="@+id/scrollViewContainer"/>

    <ScrollView
        android:id="@+id/scrollViewContainer"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:background="#B3000000"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/toolbarContainer">

        <androidx.constraintlayout.widget.ConstraintLayout
            android:id="@+id/scrollViewLayout"
            android:layout_width="match_parent"
            android:layout_height="wrap_content">

            <TextView
                android:id="@+id/createOpNameTitle"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_marginTop="30dp"
                android:text="@string/create_op_name_title"
                android:textAlignment="center"
                android:textAllCaps="true"
                android:textColor="@color/white"
                android:textSize="19sp"
                android:textStyle="bold"
                app:layout_constraintBottom_toTopOf="@+id/createOpName"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintTop_toTopOf="parent" />

            <EditText
                android:id="@+id/createOpName"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_marginHorizontal="15dp"
                android:background="@drawable/standart_rectangle"
                android:ems="10"
                android:inputType="text"
                android:minHeight="48dp"
                android:paddingStart="10dp"
                android:paddingEnd="10dp"
                android:textColor="@color/white"
                app:layout_constraintBottom_toTopOf="@+id/createOpDateTitle"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintTop_toBottomOf="@+id/createOpNameTitle" />

            <TextView
                android:id="@+id/createOpDateTitle"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_marginTop="30dp"
                android:text="@string/create_op_date_title"
                android:textAlignment="center"
                android:textAllCaps="true"
                android:textColor="@color/white"
                android:textSize="19sp"
                android:textStyle="bold"
                app:layout_constraintBottom_toTopOf="@+id/createOpDate"
```

```
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintTop_toBottomOf="@+id/createOpName" />

            <EditText
                android:id="@+id/createOpDate"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_marginHorizontal="15dp"
                android:background="@drawable/standart_rectangle"
                android:ems="10"
                android:inputType="date"
                android:minHeight="48dp"
                android:paddingStart="10dp"
                android:paddingEnd="10dp"
                android:textColor="@color/white"
                app:layout_constraintBottom_toTopOf="@+id/createOpTimeTitle"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintTop_toBottomOf="@+id/createOpDateTitle" />

            <TextView
                android:id="@+id/createOpTimeTitle"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_marginTop="30dp"
                android:text="@string/create_op_time_title"
                android:textAlignment="center"
                android:textAllCaps="true"
                android:textColor="@color/white"
                android:textSize="19sp"
                android:textStyle="bold"
                app:layout_constraintBottom_toTopOf="@+id/createOpTime"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintTop_toBottomOf="@+id/createOpDate" />

            <EditText
                android:id="@+id/createOpTime"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_marginHorizontal="15dp"
                android:background="@drawable/standart_rectangle"
                android:ems="10"
                android:inputType="time"
                android:minHeight="48dp"
                android:paddingStart="10dp"
                android:paddingEnd="10dp"
                android:textColor="@color/white"

app:layout_constraintBottom_toTopOf="@+id/createOpFirstClientTitle"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintTop_toBottomOf="@+id/createOpTimeTitle" />

            <TextView
                android:id="@+id/createOpFirstClientTitle"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_marginTop="30dp"
                android:text="@string/create_op_client_title_1"
                android:textAlignment="center"
```

```
                android:textAllCaps="true"
                android:textColor="@color/white"
                android:textSize="19sp"
                android:textStyle="bold"

app:layout_constraintBottom_toTopOf="@+id/createOpFirstClientName"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintTop_toBottomOf="@+id/createOpTime" />

            <TextView
                android:id="@+id/createOpFirstClientName"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_marginTop="5dp"
                android:text="@string/not_chosen"
                android:textAlignment="center"
                android:textColor="@color/white"
                android:textSize="19sp"
                app:layout_constraintBottom_toTopOf="@+id/pickFirstClientButton"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintStart_toStartOf="parent"

app:layout_constraintTop_toBottomOf="@+id/createOpFirstClientTitle" />

            <Button
                android:id="@+id/pickFirstClientButton"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginTop="5dp"
                android:text="@string/pick_client_button"
                android:textSize="17sp"

app:layout_constraintBottom_toTopOf="@+id/createOpSecondClientTitle"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintStart_toStartOf="parent"

app:layout_constraintTop_toBottomOf="@+id/createOpFirstClientName" />

            <TextView
                android:id="@+id/createOpSecondClientTitle"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_marginTop="30dp"
                android:text="@string/create_op_client_title_2"
                android:textAlignment="center"
                android:textAllCaps="true"
                android:textColor="@color/white"
                android:textSize="19sp"
                android:textStyle="bold"

app:layout_constraintBottom_toTopOf="@+id/createOpSecondClientName"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintTop_toBottomOf="@+id/pickFirstClientButton"
/>

            <TextView
                android:id="@+id/createOpSecondClientName"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_marginTop="5dp"
```

```
                android:text="@string/not_chosen"
                android:textAlignment="center"
                android:textColor="@color/white"
                android:textSize="19sp"

app:layout_constraintBottom_toTopOf="@+id/pickSecondClientButton"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintStart_toStartOf="parent"

app:layout_constraintTop_toBottomOf="@+id/createOpSecondClientTitle" />

            <Button
                android:id="@+id/pickSecondClientButton"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginTop="5dp"
                android:text="@string/pick_client_button"
                android:textSize="17sp"
                app:layout_constraintBottom_toTopOf="@+id/createOpObjectTitle"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintStart_toStartOf="parent"

app:layout_constraintTop_toBottomOf="@+id/createOpSecondClientName" />

            <TextView
                android:id="@+id/createOpObjectTitle"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_marginTop="30dp"
                android:text="@string/create_op_object_title"
                android:textAlignment="center"
                android:textAllCaps="true"
                android:textColor="@color/white"
                android:textSize="19sp"
                android:textStyle="bold"
                app:layout_constraintBottom_toTopOf="@+id/createOpObjectAddress"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintStart_toStartOf="parent"

app:layout_constraintTop_toBottomOf="@+id/pickSecondClientButton" />

            <TextView
                android:id="@+id/createOpObjectAddress"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_marginTop="5dp"
                android:text="@string/not_chosen"
                android:textAlignment="center"
                android:textColor="@color/white"
                android:textSize="19sp"
                app:layout_constraintBottom_toTopOf="@+id/pickObjectButton"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintTop_toBottomOf="@+id/createOpObjectTitle"
/>

            <Button
                android:id="@+id/pickObjectButton"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginTop="5dp"
                android:text="@string/pick_object_button"
```

```
                android:textSize="17sp"
                app:layout_constraintBottom_toTopOf="@+id/createOpHostTitle"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintTop_toBottomOf="@+id/createOpObjectAddress"
/>

            <TextView
                android:id="@+id/createOpHostTitle"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_marginTop="30dp"
                android:text="@string/create_op_host_title"
                android:textAlignment="center"
                android:textAllCaps="true"
                android:textColor="@color/white"
                android:textSize="19sp"
                android:textStyle="bold"
                app:layout_constraintBottom_toTopOf="@+id/createOpHostName"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintTop_toBottomOf="@+id/pickObjectButton" />

            <TextView
                android:id="@+id/createOpHostName"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_marginTop="5dp"
                android:text="@string/not_chosen"
                android:textAlignment="center"
                android:textColor="@color/white"
                android:textSize="19sp"
                app:layout_constraintBottom_toTopOf="@+id/pickHostButton"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintTop_toBottomOf="@+id/createOpHostTitle" />

            <Button
                android:id="@+id/pickHostButton"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginTop="5dp"
                android:text="@string/pick_host_button"
                android:textSize="17sp"
                app:layout_constraintBottom_toTopOf="@+id/createOpButton"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintTop_toBottomOf="@+id/createOpHostName" />

            <Button
                android:id="@+id/createOpButton"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginTop="30dp"
                android:layout_marginBottom="5dp"
                android:text="@string/create_op_button"
                android:textSize="19sp"
                android:textAllCaps="true"
                android:backgroundTint="@android:color/holo_blue_dark"
                app:layout_constraintBottom_toBottomOf="parent"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintStart_toStartOf="parent"
```

```
                app:layout_constraintTop_toBottomOf="@+id/pickHostButton" />
        </androidx.constraintlayout.widget.ConstraintLayout>
    </ScrollView>
</androidx.constraintlayout.widget.ConstraintLayout>
```

A.4 Файл «CreateOperationActivity.java»

```
package com.example.prethesispractice.activities;

import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import androidx.activity.result.ActivityResultLauncher;
import androidx.activity.result.contract.ActivityResultContracts;

import com.example.prethesispractice.R;
import com.example.prethesispractice.entities.Client;
import com.example.prethesispractice.entities.Employee;
import com.example.prethesispractice.entities.EstateObject;
import com.example.prethesispractice.entities.Operation;
import com.example.prethesispractice.helpers.InputFieldsValidation;
import com.google.gson.Gson;

import java.io.IOException;

import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;

public class CreateOperationActivity extends ToolbarMenuActivity {
    private EditText operationName;
    private EditText operationDate;
    private EditText operationTime;
    private TextView leadCounteragentNameView;
    private Button pickFirstClientButton;
    private TextView secondaryCounteragentNameView;
    private Button pickSecondClientButton;
    private TextView objectAddressView;
    private Button pickObjectButton;
    private TextView hostNameView;
    private Button pickHostButton;
    private Button createButton;
    private Client counteragentLead;
    private Client counteragentSecondary;
    private EstateObject estateObject;
    private Employee host;
    private Operation leadOperation;
    private Operation secondaryOperation;
    private final ActivityResultLauncher<Intent> chooseLeadLauncher =
registerForActivityResult(
            new ActivityResultContracts.StartActivityForResult(),
            result -> {
                if (result.getResultCode() == RESULT_OK) {
```

```
                    String chosenClient =
result.getData().getStringExtra("result_client");

                    Gson jsonConverter = new Gson();
                    counteragentLead = jsonConverter.fromJson(chosenClient,
Client.class);

leadCounteragentNameView.setText(counteragentLead.getName());
                }
            });
    private final ActivityResultLauncher<Intent> chooseSecondaryLauncher =
registerForActivityResult(
            new ActivityResultContracts.StartActivityForResult(),
            result -> {
                if (result.getResultCode() == RESULT_OK) {
                    String chosenClient =
result.getData().getStringExtra("result_client");

                    Gson jsonConverter = new Gson();
                    counteragentSecondary = jsonConverter.fromJson(chosenClient,
Client.class);

secondaryCounteragentNameView.setText(counteragentSecondary.getName());
                }
            });
    private final ActivityResultLauncher<Intent> chooseObjectLauncher =
registerForActivityResult(
            new ActivityResultContracts.StartActivityForResult(),
            result -> {
                if (result.getResultCode() == RESULT_OK) {
                    String chosenObject =
result.getData().getStringExtra("result_object");

                    Gson jsonConverter = new Gson();
                    estateObject = jsonConverter.fromJson(chosenObject,
EstateObject.class);
                    objectAddressView.setText(estateObject.getAddress());
                }
            });
    private final ActivityResultLauncher<Intent> chooseHostLauncher =
registerForActivityResult(
            new ActivityResultContracts.StartActivityForResult(),
            result -> {
                if (result.getResultCode() == RESULT_OK) {
                    String chosenHost =
result.getData().getStringExtra("result_employee");

                    Gson jsonConverter = new Gson();
                    host = jsonConverter.fromJson(chosenHost, Employee.class);
                    hostNameView.setText(host.getName());
                }
            });

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_create_operation);
        inflateToolbar(null);

        String previousLeadOperation = getIntent().getStringExtra("prev_lead");
        String previousSecondaryOperation =
getIntent().getStringExtra("prev_secondary");
```

```
        final String token = getIntent().getStringExtra("token");

        if (token != null && !token.isBlank()) {
            operationName = findViewById(R.id.createOpName);
            operationDate = findViewById(R.id.createOpDate);
            operationTime = findViewById(R.id.createOpTime);
            leadCounteragentNameView =
findViewById(R.id.createOpFirstClientName);
            pickFirstClientButton = findViewById(R.id.pickFirstClientButton);
            secondaryCounteragentNameView =
findViewById(R.id.createOpSecondClientName);
            pickSecondClientButton = findViewById(R.id.pickSecondClientButton);
            objectAddressView = findViewById(R.id.createOpObjectAddress);
            pickObjectButton = findViewById(R.id.pickObjectButton);
            hostNameView = findViewById(R.id.createOpHostName);
            pickHostButton = findViewById(R.id.pickHostButton);
            createButton = findViewById(R.id.createOpButton);

            if (previousLeadOperation != null &&
!previousLeadOperation.isBlank()) {
                Gson jsonConverter = new Gson();
                leadOperation = jsonConverter.fromJson(previousLeadOperation,
Operation.class);

                httpApi.getClientById("Bearer " + token,
leadOperation.getCounteragentLead()).enqueue(new Callback<Client>() {
                    @Override
                    public void onResponse(Call<Client> call, Response<Client>
response) {
                        if (response.isSuccessful()) {
                            counteragentLead = response.body();


leadCounteragentNameView.setText(counteragentLead.getName());
                            pickFirstClientButton.setEnabled(false);

pickFirstClientButton.setBackgroundTintList(getApplicationContext().getColorStat
eList(R.color.gray));
                        } else {
                            Toast.makeText(getApplicationContext(), "Помилка під
час підтягування головного " +
                                    "контрагента з ланцюга операцій",
Toast.LENGTH_LONG).show();
                            finish();
                        }
                    }

                    @Override
                    public void onFailure(Call<Client> call, Throwable
throwable) {
                        Toast.makeText(getApplicationContext(), "Помилка під час
підтягування головного " +
                                        "контрагента з ланцюга операцій(" +
throwable.getMessage() + ")", Toast.LENGTH_LONG).show();
                        finish();
                    }
                });
                httpApi.getEstateObjectById("Bearer " + token,
leadOperation.getEstateObjectId()).enqueue(new Callback<EstateObject>() {
                    @Override
                    public void onResponse(Call<EstateObject> call,
Response<EstateObject> response) {
```

```
                        if (response.isSuccessful()) {
                            estateObject = response.body();


objectAddressView.setText(estateObject.getAddress());
                            pickObjectButton.setEnabled(false);

pickObjectButton.setBackgroundTintList(getApplicationContext().getColorStateList
(R.color.gray));
                        } else {
                            Toast.makeText(getApplicationContext(), "Помилка під
час підтягування об'єкта з ланцюга операцій", Toast.LENGTH_LONG).show();
                            finish();
                        }
                    }

                    @Override
                    public void onFailure(Call<EstateObject> call, Throwable
throwable) {
                        Toast.makeText(getApplicationContext(), "Помилка під час
підтягування об'єкта з ланцюга операцій("
                                        + throwable.getMessage() + ")",
Toast.LENGTH_LONG).show();
                        finish();
                    }
                });
            }

            if (previousSecondaryOperation != null &&
!previousSecondaryOperation.isBlank()) {
                Gson jsonConverter = new Gson();
                secondaryOperation =
jsonConverter.fromJson(previousSecondaryOperation, Operation.class);

                httpApi.getClientById("Bearer " + token,

secondaryOperation.getCounteragentSecondary() == null
                                        ? secondaryOperation.getCounteragentLead()
                                        :
secondaryOperation.getCounteragentSecondary())
                        .enqueue(new Callback<Client>() {
                    @Override
                    public void onResponse(Call<Client> call, Response<Client>
response) {
                        if (response.isSuccessful()) {
                            counteragentSecondary = response.body();


secondaryCounteragentNameView.setText(counteragentSecondary.getName());
                            pickSecondClientButton.setEnabled(false);

pickSecondClientButton.setBackgroundTintList(getApplicationContext().getColorSta
teList(R.color.gray));
                        } else {
                            Toast.makeText(getApplicationContext(), "Помилка під
час підтягування другого " +
                                            "контрагента з ланцюга операцій",
Toast.LENGTH_LONG).show();
                            finish();
                        }
                    }
```

```
                    @Override
                    public void onFailure(Call<Client> call, Throwable
throwable) {
                        Toast.makeText(getApplicationContext(), "Помилка під час
підтягування другого " +
                                "контрагента з ланцюга операцій(" +
throwable.getMessage() + ")", Toast.LENGTH_LONG).show();
                        finish();
                    }
                });
            }

            createButton.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View view) {
                    Integer operationId = null;
                    String name = operationName.getText().toString();
                    String date = operationDate.getText().toString();
                    String time = operationTime.getText().toString();
                    int counteragentLeadId = 0;
                    Integer counteragentSecondaryId = null;
                    Integer estateObjectId = null;
                    int hostId = 0;
                    String status = "Очікується";
                    String actType = null;
                    Integer fkOperationLead = null;
                    Integer fkOperationSecondary = null;

                    boolean checkPassFlag = true;
                    if (counteragentLead == null || host == null) {
                        Toast.makeText(getApplicationContext(), "Оберіть
обов'язкові поля"
                                        + "(перший контрагент, хост)",
Toast.LENGTH_LONG).show();
                        checkPassFlag = false;
                    }
                    if (name.isBlank()) {
                        operationName.setError("Назва операції не може бути
порожньою");
                        checkPassFlag = false;
                    }
                    if (!InputFieldsValidation.checkDate(date)) {
                        operationDate.setError("Дата повинна відповідати формату
рік(4 цифри)-місяць(2 цифри)-день(2 цифри)");
                        checkPassFlag = false;
                    }
                    if (!InputFieldsValidation.checkTime(time)) {
                        operationTime.setError("Час повинен відповідати формату
година(2 цифри):хвилини(2 цифри):секунди(2 цифри)");
                        checkPassFlag = false;
                    }

                    if (counteragentLead == null) {
                        Toast.makeText(getApplicationContext(), "Оберіть
головного(першого) контрагента", Toast.LENGTH_LONG).show();
                        checkPassFlag = false;
                    } else {
                        counteragentLeadId = counteragentLead.getClientId();
                    }

                    if (counteragentSecondary != null) {
```

```
                        counteragentSecondaryId =
counteragentSecondary.getClientId();
                    }

                    if ((leadOperation == null && secondaryOperation != null)
                            || (leadOperation != null && secondaryOperation ==
null)) {
                        Toast.makeText(getApplicationContext(), "Не було
підтягнуто усі минулі операції", Toast.LENGTH_LONG).show();
                        checkPassFlag = false;
                    } else if (leadOperation != null && secondaryOperation !=
null) {
                        fkOperationLead = leadOperation.getOperationId();
                        fkOperationSecondary =
secondaryOperation.getOperationId();
                    }

                    if (estateObject != null) {
                        estateObjectId = estateObject.getEstateObjectId();
                    }

                    if (host == null) {
                        Toast.makeText(getApplicationContext(), "Оберіть хоста",
Toast.LENGTH_LONG).show();
                        checkPassFlag = false;
                    } else {
                        hostId = host.getEmployeeId();
                    }

                    if (checkPassFlag) {
                        Operation newOperation = new Operation(operationId,
name, date + " " + time,

counteragentLeadId, counteragentSecondaryId,
                                                                estateObjectId,
hostId, null, status,
                                                                actType,
fkOperationLead, fkOperationSecondary);

                        httpApi.insertOperation("Bearer " + token,
newOperation).enqueue(new Callback<Operation>() {
                            @Override
                            public void onResponse(Call<Operation> call,
Response<Operation> response) {
                                if (response.isSuccessful()) {
                                    Operation returnOperation = response.body();
                                    finishCreatingWithResult("Операція успішно
додана!", returnOperation);
                                } else {
                                    finishCreatingWithError("Помилка при
збереженні нової операції. Спробуйте знову");
                                }
                            }

                            @Override
                            public void onFailure(Call<Operation> call,
Throwable throwable) {
                                finishCreatingWithError("Помилка при збереженні
нової операції("
                                                        + throwable.getMessage()
+ "). Спробуйте знову");
                            }
```

```
                        });
                    }
                }
            });
        }

        pickFirstClientButton = findViewById(R.id.pickFirstClientButton);
        pickSecondClientButton = findViewById(R.id.pickSecondClientButton);
        pickObjectButton = findViewById(R.id.pickObjectButton);
        pickHostButton = findViewById(R.id.pickHostButton);

        pickFirstClientButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                cameFromCreateOperation(ClientsSearchActivity.class, "lead");
            }
        });
        pickSecondClientButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                cameFromCreateOperation(ClientsSearchActivity.class,
"secondary");
            }
        });
        pickObjectButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                cameFromCreateOperation(ObjectsSearchActivity.class, "objects");
            }
        });
        pickHostButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                cameFromCreateOperation(EmployeeSearchActivity.class, "hosts");
            }
        });
    }

    @Override
    protected void onStart() {
        super.onStart();
    }

    @Override
    protected void onRestart() {
        super.onRestart();
    }

    @Override
    protected void onResume() {
        super.onResume();
    }

    @Override
    protected void onPause() {
        super.onPause();
    }

    @Override
    protected void onStop() {
        super.onStop();
    }
```

```
    @Override
    protected void onDestroy() {
        super.onDestroy();
    }

    private void cameFromCreateOperation(Class<?> activityClass, String
launcherType) {
        Intent moveToActivity = new Intent(getApplicationContext(),
activityClass);
        moveToActivity.putExtra("login", getIntent().getStringExtra("login"));
        moveToActivity.putExtra("role", getIntent().getStringExtra("role"));
        moveToActivity.putExtra("token", getIntent().getStringExtra("token"));
        moveToActivity.putExtra("employee_id",
getIntent().getIntExtra("employee_id", 0));

        moveToActivity.putExtra("FROM_CREATE_OPERATION", true);

        String employeeNameExtra = getIntent().getStringExtra("employeeName");

        if (employeeNameExtra != null) {
            moveToActivity.putExtra("employeeName", employeeNameExtra);
        }

        if (launcherType.equals("lead")) {
            chooseLeadLauncher.launch(moveToActivity);
        } else if (launcherType.equals("secondary")) {
            chooseSecondaryLauncher.launch(moveToActivity);
        } else if (launcherType.equals("objects")) {
            chooseObjectLauncher.launch(moveToActivity);
        } else {
            chooseHostLauncher.launch(moveToActivity);
        }

    }

    private void finishCreatingWithResult(String message, Operation operation) {
        Gson jsonConverter = new Gson();
        String insertedOperation = jsonConverter.toJson(operation);

        Intent moveToOperationDetailed = new Intent(getApplicationContext(),
OperationsDetailedActivity.class);
        moveToOperationDetailed.putExtra("chosen_operation", insertedOperation);

        Toast.makeText(getApplicationContext(), message,
Toast.LENGTH_LONG).show();
        passEmployeeData(moveToOperationDetailed);
        finish();
    }

    private void finishCreatingWithError(String message) {
        Toast.makeText(getApplicationContext(), message,
Toast.LENGTH_LONG).show();
        finish();
    }
}
```

A.5 Файл «OperationsController.cs»

```
using System;
```

```
using System.Collections.Generic;
using System.Globalization;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Authorization;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using MySqlX.XDevAPI;
using RealtYeahBackend.Entities;
using RealtYeahBackend.Models;
using RealtYeahBackend.Services;

namespace RealtYeahBackend.Controllers
{
    [Authorize]
    [Route("api/[controller]")]
    [ApiController]
    [RequireHttps]
    public class OperationsController : ControllerBase
    {
        private readonly RealtyeahContext context;
        private readonly OperationService operationService;

        public OperationsController(RealtyeahContext context,
OperationService operationService)
        {
            this.context = context;
            this.operationService = operationService;
        }

        // GET: api/Operations
        [HttpGet]
        public async Task<ActionResult<IEnumerable<Operation>>>
GetOperations()
        {
            if (context.Operations == null)
            {
                return NotFound();
            }
            return await context.Operations.ToListAsync();
```

```
        }

        // GET: api/Operations/5
        [HttpGet("{id}")]
        public async Task<ActionResult<Operation>> GetOperation(int id)
        {
            if (context.Operations == null)
            {
                return NotFound();
            }
            var operation = await context.Operations.FindAsync(id);

            if (operation == null)
            {
                return NotFound();
            }

            return operation;
        }

        // PUT: api/Operations/5
        //     To     protect     from     overposting     attacks,     see
https://go.microsoft.com/fwlink/?linkid=2123754
        [Authorize(Roles = Role.Admin + "," + Role.MainAdmin)]
        [HttpPut("{id}")]
        public  async  Task<IActionResult>  PutOperation(int  id,  Operation
operation)
        {
            if (id != operation.OperationId)
            {
                return BadRequest();
            }

            context.Entry(operation).State = EntityState.Modified;

            try
            {
                await context.SaveChangesAsync();
            }
            catch (DbUpdateConcurrencyException)
            {
```

```
                    if (!OperationExists(id))
                    {
                        return NotFound();
                    }
                    else
                    {
                        throw;
                    }
                }

                return NoContent();
            }

            // POST: api/Operations
            //     To     protect     from     overposting     attacks,     see
https://go.microsoft.com/fwlink/?linkid=2123754
            [HttpPost]
            public async Task<ActionResult<Operation>> PostOperation(Operation
operation)
            {
                if (context.Operations == null)
                {
                    return  Problem("Entity  set  'RealtyeahContext.Operations'
is null.");
                }

                operation.Status = "Очікується";
                context.Operations.Add(operation);
                await context.SaveChangesAsync();

                if (operation.CounteragentSecondary == null)
                {
                    context.ClientsStatusesAssignments.Add(new
ClientsStatusesAssignment
                    {
                        ClientId = operation.CounteragentLead,
                        Status = "Внесений у базу",
                        OperationId = operation.OperationId
                    });
                }
```

```
                return CreatedAtAction("GetOperation", new { id =
operation.OperationId }, operation);
            }

            // DELETE: api/Operations/5
            [HttpDelete("{id}")]
            public async Task<IActionResult> DeleteOperation(int id)
            {
                if (context.Operations == null)
                {
                    return NotFound();
                }
                var operation = await context.Operations.FindAsync(id);
                if (operation == null)
                {
                    return NotFound();
                }
                else if (!string.IsNullOrWhiteSpace(operation.ActType) &&
!operation.Status.Equals("Неуспішно"))
                {
                    return BadRequest("Can't delete a successful operation with
already assigned act");
                }

                context.Operations.Remove(operation);
                await context.SaveChangesAsync();

                return NoContent();
            }

            [HttpPut("{id}/assign/{actType}")]
            public IActionResult AssignActToOperation(int id, string actType)
            {
                try
                {
                    Operation operationWithAct = operationService.AddAct(id,
actType);
                    return Ok(operationWithAct);
                }
                catch (ArgumentException ex)
                {
```

```
                return BadRequest(ex.Message);
            }
        }

        [HttpPut("{id}/cancel")]
        public IActionResult CancelOperation(int id)
        {
            try
            {
                Operation cancelledOperation = operationService.CancelOperation(id);
                return Ok(cancelledOperation);
            }
            catch (ArgumentException ex)
            {
                return BadRequest(ex.Message);
            }
            catch (NullReferenceException ex)
            {
                return BadRequest(ex.Message);
            }
        }

        [HttpGet("bd/{start}/to/{end}")]
        public async Task<ActionResult<IList<Operation>>> GetOperationsByDate(string start, string end)
        {
            if (context.Operations == null)
            {
                return NotFound();
            }

            IList<Operation> operations = await context.Operations.Where(operation => operation.Date.Date >= DateTime.ParseExact(start, "ddMMyyyy", CultureInfo.InvariantCulture).Date

&& operation.Date.Date <= DateTime.ParseExact(end, "ddMMyyyy", CultureInfo.InvariantCulture).Date).ToListAsync();

            return Ok(operations);
        }
```

```
            [AllowAnonymous]
            [HttpGet("bd/{start}/to/{end}/host/{id}")]
            public async Task<ActionResult<IList<Operation>>>
GetOperationsByDateAndHost(string start, string end, int id)
            {
                if (context.Operations == null)
                {
                    return NotFound();
                }

                IList<Operation> operations = await
context.Operations.Where(operation => operation.Date.Date >=
DateTime.ParseExact(start, "ddMMyyyy", CultureInfo.InvariantCulture).Date

&& operation.Date.Date <= DateTime.ParseExact(end, "ddMMyyyy",
CultureInfo.InvariantCulture).Date

&& operation.HostId == id).ToListAsync();

                return Ok(operations);
            }

            [HttpGet("host/{id}")]
            public async Task<ActionResult<IList<Operation>>>
GetOperationsByHost(int id)
            {
                if (context.Operations == null)
                {
                    return NotFound();
                }

                IList<Operation> operations = await
context.Operations.Where(operation => operation.HostId == id).ToListAsync();

                return Ok(operations);
            }

            [HttpGet("{id}/host")]
            public async Task<ActionResult<Employee>> GetHostByOperation(int
id)
```

```
        {
            if (context.Operations == null)
            {
                return NotFound();
            }

            Employee host = await context.Operations.Where(operation =>
operation.OperationId == id)
                .Select(operation =>
operation.Host).FirstOrDefaultAsync();

            if (host == null)
            {
                return NotFound();
            }

            return Ok(host);
        }

        [HttpGet("{id}/Object")]
        public async Task<ActionResult<EstateObject>> GetRelatedObject(int
id)
        {
            if (context.Operations == null)
            {
                return NotFound();
            }

            EstateObject estateObject = await context.Operations.Where(op
=> op.OperationId == id)
                .Select(op =>
op.EstateObjectNavigation).FirstOrDefaultAsync();

            if (estateObject == null)
            {
                return NotFound();
            }

            return Ok(estateObject);
        }
```

```
        [HttpGet("{id}/ObjectAddress")]
        public async Task<ActionResult<string>> GetRelatedObjectAddress(int
id)
        {
            if (context.Operations == null)
            {
                return NotFound();
            }

            Operation operation = await context.Operations.Where(operation
=> operation.OperationId == id)
                .Include(operation                                          =>
operation.EstateObjectNavigation).FirstOrDefaultAsync();

            if (operation == null)
            {
                return NotFound();
            }

            EstateObject estateObject = operation.EstateObjectNavigation;

            if (estateObject == null)
            {
                return NotFound();
            }

            return Ok("\"" + estateObject.Address + "\"");
        }

        [HttpGet("{id}/availableActs")]
        public ActionResult<List<bool>> GetAvailableActs(int id)
        {
            if (context.Operations == null)
            {
                return NotFound();
            }

            Operation  operation  =  context.Operations.Where(operation  =>
operation.OperationId == id)
                .Include(operation => operation.FkOperationLeadNavigation)
```

```
                    .Include(operation => operation.FkOperationSecondaryNavigation)
                    .Include(operation => operation.InverseFkOperationLeadNavigation)
                    .Include(operation => operation.InverseFkOperationSecondaryNavigation)
                    .FirstOrDefault();

                if (operation == null)
                {
                    return NotFound();
                }

                List<bool> availableActs = operationService.GetAvailableActs(operation);

                return Ok(availableActs);
            }

            [HttpGet("{id}/availableForNext")]
            public ActionResult<bool> GetNextAvailability(int id)
            {
                if (context.Operations == null)
                {
                    return NotFound();
                }

                Operation operation = context.Operations.Find(id);

                if (operation == null)
                {
                    return NotFound();
                }

                return Ok(operationService.GetNextAvailability(operation));
            }

            [HttpGet("actTypeActive/{actType}")]
            public async Task<ActionResult<IList<Operation>>> GetOperationsByActiveAct(string actType)
            {
```

```
                if (context.Operations == null)
                {
                    return NotFound();
                }

                List<Operation>          operations          =          await
context.Operations.Where(operation => actType.Equals(operation.ActType)

&& operation.Status.Equals("Успішно")).ToListAsync();
                List<Operation> availableOperations = new List<Operation>();

                foreach (Operation operation in operations)
                {
                    if (operationService.GetNextAvailability(operation))
                    {
                        availableOperations.Add(operation);
                    }
                }

                return Ok(availableOperations);
            }

            private bool OperationExists(int id)
            {
                return    (context.Operations?.Any(e    =>    e.OperationId    ==
id)).GetValueOrDefault();
            }
        }
}
```

A.6 Файл «OperationService.cs»

```
using RealtYeahBackend.Entities;
using RealtYeahBackend.Models.Constants;
using Microsoft.EntityFrameworkCore;

namespace RealtYeahBackend.Services
{
    public class OperationService
    {
        private readonly RealtyeahContext context;

        public OperationService(RealtyeahContext context)
        {
            this.context = context;
        }
```

```
        public Operation AddAct(int operationID, string actType)
        {
            Operation operation = context.Operations.Where(operation =>
operation.OperationId == operationID)
                .Include(operation => operation.CounteragentLeadNavigation)
                .ThenInclude(counteragent =>
counteragent.ClientsStatusesAssignments)
                .ThenInclude(status => status.Operation)
                .Include(operation => operation.CounteragentSecondaryNavigation)
                .ThenInclude(counteragent =>
counteragent.ClientsStatusesAssignments)
                .ThenInclude(status => status.Operation)
                .Include(operation => operation.EstateObjectNavigation)
                .FirstOrDefault();

            if (operation == null)
            {
                throw new ArgumentException("No operation with such ID");
            }
            else if (!string.IsNullOrWhiteSpace(operation.ActType))
            {
                throw new ArgumentException("Act has been already assigned to
this operation");
            }

            string counteragentLeadStatus = "";
            string counteragentSecondaryStatus = "";
            string objectStatus = "";
            if (ActTypesConst.BuyAgentDeal.Equals(actType))
            {
                operation = RemoveArchiveStatuses(operation);

                counteragentLeadStatus =
context.ClientsStatuses.Find("Покупець").Status;
            }
            else if (ActTypesConst.SellAgentDeal.Equals(actType))
            {
                operation = RemoveArchiveStatuses(operation);

                counteragentLeadStatus =
context.ClientsStatuses.Find("Продавець").Status;
                objectStatus = context.ObjectsStatuses.Find("Виставлений на
продаж").Status;
            }
            else if (ActTypesConst.RentAgentDeal.Equals(actType))
            {
                operation = RemoveArchiveStatuses(operation);

                counteragentLeadStatus =
context.ClientsStatuses.Find("Орендар").Status;
            }
            else if (ActTypesConst.ForRentAgentDeal.Equals(actType))
            {
                operation = RemoveArchiveStatuses(operation);

                counteragentLeadStatus =
context.ClientsStatuses.Find("Орендодавець").Status;
                objectStatus = context.ObjectsStatuses.Find("Виставлений на
оренду").Status;
            }
            else if (ActTypesConst.Pledge.Equals(actType))
```

```
            {
                objectStatus =
context.ObjectsStatuses.Find("Зарезервований").Status;
            }
            else if (ActTypesConst.FinalDeal.Equals(actType))
            {
                objectStatus =
context.ObjectsStatuses.Find("Архівований").Status;

                ClientsStatusesAssignment leadStatusAssignment =
operation.CounteragentLeadNavigation.ClientsStatusesAssignments

.Where(status => status.Operation.EstateObjectId == operation.EstateObjectId)

.FirstOrDefault();
                ClientsStatusesAssignment secondaryStatusAssignment =
operation.CounteragentSecondaryNavigation.ClientsStatusesAssignments

.Where(status => ActTypesConst.BuyAgentDeal.Equals(status.Operation.ActType)

|| ActTypesConst.RentAgentDeal.Equals(status.Operation.ActType))

.FirstOrDefault();

operation.CounteragentLeadNavigation.ClientsStatusesAssignments.Remove(leadStatu
sAssignment);

operation.CounteragentSecondaryNavigation.ClientsStatusesAssignments.Remove(seco
ndaryStatusAssignment);

                if
(operation.CounteragentLeadNavigation.ClientsStatusesAssignments.Count <= 0)
                {
                    counteragentLeadStatus = "Минулий клієнт";
                }
                if
(operation.CounteragentSecondaryNavigation.ClientsStatusesAssignments.Count <=
0)
                {
                    counteragentSecondaryStatus = "Минулий клієнт";
                }
            }

            if (string.IsNullOrWhiteSpace(counteragentLeadStatus)
                && string.IsNullOrWhiteSpace(counteragentSecondaryStatus)
                && string.IsNullOrWhiteSpace(objectStatus)
                && !ActTypesConst.ObjectCheck.Equals(actType))
            {
                throw new ArgumentException("Attempt to assign a non-existing
act type \"" + actType + "\"");
            }

            operation.ActType = actType;
            if (!string.IsNullOrWhiteSpace(counteragentLeadStatus))
            {
                operation.ClientsStatusesAssignments.Add(new
ClientsStatusesAssignment
                {
                    ClientId = operation.CounteragentLead,
                    Status = counteragentLeadStatus,
                    OperationId = operation.OperationId
                });
```

```
            }
            if (!string.IsNullOrWhiteSpace(counteragentSecondaryStatus))
            {
                operation.ClientsStatusesAssignments.Add(new
ClientsStatusesAssignment
                {
                    ClientId = operation.CounteragentSecondary.Value,
                    Status = counteragentSecondaryStatus,
                    OperationId = operation.OperationId
                });
            }
            if (!string.IsNullOrWhiteSpace(objectStatus))
            {
                operation.EstateObjectNavigation.Status = objectStatus;
            }

            operation.Status = "Успішно";

            context.SaveChanges();

            return operation;
        }

        public Operation CancelOperation(int operationID)
        {
            Operation operation = context.Operations.Where(operation =>
operation.OperationId == operationID)
                .Include(operation => operation.EstateObjectNavigation)
                .Include(operation => operation.CounteragentLeadNavigation)
                .ThenInclude(counteragent =>
counteragent.ClientsStatusesAssignments)
                .FirstOrDefault();

            if (operation == null)
            {
                throw new ArgumentException("No operation with such ID");
            }
            else if (string.IsNullOrWhiteSpace(operation.ActType))
            {
                operation.Status =
context.OperationsStatuses.Find("Неуспішно").Status;
            }
            else
            {
                if (ActTypesConst.Pledge.Equals(operation.ActType))
                {
                    Operation backupOperation = context.Operations.Where(op =>
op.CounteragentLead == operation.CounteragentLead
                                                                          &&
op.EstateObjectId == operation.EstateObjectId
                                                                          &&
!(ActTypesConst.Pledge.Equals(op.ActType) ||
ActTypesConst.FinalDeal.Equals(op.ActType)))

.OrderByDescending(op => op.Date)
                                                                .Include(op =>
op.EstateObjectNavigation)

.FirstOrDefault();

                    if (backupOperation == null)
                    {
```

```
                        throw new NullReferenceException("Can't find any
suitable backup operation");
                    }

                    operation.EstateObjectNavigation.Status =
backupOperation.EstateObjectNavigation.Status;
                }
                else if (ActTypesConst.BuyAgentDeal.Equals(operation.ActType)
                        || ActTypesConst.SellAgentDeal.Equals(operation.ActType)
                        || ActTypesConst.RentAgentDeal.Equals(operation.ActType)
                        ||
ActTypesConst.ForRentAgentDeal.Equals(operation.ActType))
                {
                    operation.EstateObjectNavigation.Status =
context.ObjectsStatuses.Find("Архівований").Status;
                    ClientsStatusesAssignment status =
operation.CounteragentLeadNavigation.ClientsStatusesAssignments
                                                        .Where(status =>
status.OperationId == operation.OperationId)
                                                        .FirstOrDefault();

operation.CounteragentLeadNavigation.ClientsStatusesAssignments.Remove(status);

                    if
(operation.CounteragentLeadNavigation.ClientsStatusesAssignments.Count <= 0)
                    {

operation.CounteragentLeadNavigation.ClientsStatusesAssignments.Add(new
ClientsStatusesAssignment { ClientId = operation.CounteragentLead,

Status = context.ClientsStatuses.Find("Минулий клієнт").Status,

OperationId = operation.OperationId });
                    }
                }

                operation.Status =
context.OperationsStatuses.Find("Неуспішно").Status;
            }

            context.SaveChanges();

            return operation;
        }

        private Operation RemoveArchiveStatuses(Operation operation)
        {
            List<ClientsStatusesAssignment> leadStatusAssignments =
operation.CounteragentLeadNavigation.ClientsStatusesAssignments
                .Where(status => status.Status.Equals("Внесений у базу")
                                || status.Status.Equals("Минулий
клієнт")).ToList();

            if (leadStatusAssignments != null)
            {
                foreach (ClientsStatusesAssignment assignment in
leadStatusAssignments)
                {

operation.CounteragentLeadNavigation.ClientsStatusesAssignments.Remove(assignmen
t);
                }
```

```
            }

            return operation;
        }

        public List<bool> GetAvailableActs(Operation operation)
        {
            List<bool> availableActs = new List<bool> { false, false, false,
false, false, false, false, false, false };

            if (ActTypesConst.FinalDeal.Equals(operation.ActType))
            {
                return availableActs;
            }

            if ((operation.InverseFkOperationLeadNavigation == null ||
operation.InverseFkOperationLeadNavigation.Count == 0)
                 && (operation.InverseFkOperationSecondaryNavigation == null ||
operation.InverseFkOperationSecondaryNavigation.Count == 0))
            {
                availableActs[7] = !operation.Status.Equals("Неуспішно");

                availableActs[8] = true;
            }
            else if (operation.InverseFkOperationLeadNavigation != null
                     && operation.InverseFkOperationLeadNavigation.Where(op =>
op.Status.Equals("Неуспішно")).Count() ==
operation.InverseFkOperationLeadNavigation.Count
                     && operation.InverseFkOperationSecondaryNavigation != null
                     && operation.InverseFkOperationSecondaryNavigation.Where(op
=> op.Status.Equals("Неуспішно")).Count() ==
operation.InverseFkOperationSecondaryNavigation.Count)
            {
                availableActs[7] = !operation.Status.Equals("Неуспішно");
            }

            if (string.IsNullOrWhiteSpace(operation.ActType))
            {
                if (operation.FkOperationLeadNavigation == null &&
operation.CounteragentSecondary == null)
                {
                    availableActs[0] = true;
                    availableActs[1] = true;
                    availableActs[2] = true;
                    availableActs[3] = true;
                }

                if (operation.FkOperationLeadNavigation != null &&
operation.FkOperationSecondaryNavigation != null
                    && (operation.InverseFkOperationLeadNavigation == null
                        || operation.InverseFkOperationLeadNavigation.Count == 0
                        || operation.InverseFkOperationLeadNavigation.Where(op
=> op.Status.Equals("Неуспішно")).Count() ==
operation.InverseFkOperationLeadNavigation.Count)
                    && (operation.InverseFkOperationSecondaryNavigation == null
                        || operation.InverseFkOperationSecondaryNavigation.Count
== 0
                        ||
operation.InverseFkOperationSecondaryNavigation.Where(op =>
op.Status.Equals("Неуспішно")).Count() ==
operation.InverseFkOperationSecondaryNavigation.Count))
                {
```

```
                    if
((ActTypesConst.SellAgentDeal.Equals(operation.FkOperationLeadNavigation.ActType
)
                            &&
ActTypesConst.BuyAgentDeal.Equals(operation.FkOperationSecondaryNavigation.ActTy
pe))
                           ||
(ActTypesConst.ForRentAgentDeal.Equals(operation.FkOperationLeadNavigation.ActTy
pe)
                               &&
ActTypesConst.RentAgentDeal.Equals(operation.FkOperationSecondaryNavigation.ActT
ype)))
                    {
                        availableActs[4] = true;
                    }
                    else if
(ActTypesConst.ObjectCheck.Equals(operation.FkOperationLeadNavigation.ActType)
                                &&
ActTypesConst.ObjectCheck.Equals(operation.FkOperationSecondaryNavigation.ActTyp
e))
                    {
                        availableActs[5] = true;
                    }
                    else if
(ActTypesConst.Pledge.Equals(operation.FkOperationLeadNavigation.ActType)
                                &&
ActTypesConst.Pledge.Equals(operation.FkOperationSecondaryNavigation.ActType))
                    {
                        availableActs[6] = true;
                    }
                }
            }

            return availableActs;
        }

        public bool GetNextAvailability(Operation operation)
        {
            if (string.IsNullOrWhiteSpace(operation.ActType))
            {
                return false;
            }
            else if ((ActTypesConst.SellAgentDeal.Equals(operation.ActType) ||
ActTypesConst.ForRentAgentDeal.Equals(operation.ActType))
                && operation.Status.Equals("Успішно"))
            {
                return TravelOperationTree(operation, operation.ActType);
            }
            else if (ActTypesConst.BuyAgentDeal.Equals(operation.ActType) ||
ActTypesConst.RentAgentDeal.Equals(operation.ActType)
                     && operation.Status.Equals("Успішно"))
            {
                return TravelOperationTreeSecondary(operation, 1,
operation.ActType);
            }
            else
            {
                Operation leadOperation = operation;

                while (leadOperation.FkOperationLead != null)
                {
                    context.Entry(leadOperation)
```

```
                        .Reference(o => o.FkOperationLeadNavigation)
                        .Load();
                    leadOperation = leadOperation.FkOperationLeadNavigation;
                };

                Operation secondaryOperation = operation;

                while (secondaryOperation.FkOperationSecondary != null)
                {
                    context.Entry(secondaryOperation)
                        .Reference(o => o.FkOperationSecondaryNavigation)
                        .Load();
                    secondaryOperation =
secondaryOperation.FkOperationSecondaryNavigation;
                };

                return TravelOperationTree(leadOperation, operation.ActType) &&
TravelOperationTreeSecondary(secondaryOperation, 1, operation.ActType);
            }
        }

        private bool TravelOperationTree(Operation operation, string initialAct)
        {
            context.Entry(operation)
                .Collection(o => o.InverseFkOperationLeadNavigation)
                .Load();

            foreach (Operation op in operation.InverseFkOperationLeadNavigation)
            {
                if (ActTypesConst.Pledge.Equals(op.ActType) &&
op.Status.Equals("Успішно")
                    && !ActTypesConst.Pledge.Equals(initialAct))
                {
                    return false;
                }
            }

            foreach (Operation op in operation.InverseFkOperationLeadNavigation)
            {
                return TravelOperationTree(op, initialAct);
            }

            return true;
        }

        private bool TravelOperationTreeSecondary(Operation operation, int
level, string initialAct)
        {
            List<Operation> operations = new List<Operation>();

            if (level > 1)
            {
                context.Entry(operation)
                    .Collection(o => o.InverseFkOperationSecondaryNavigation)
                    .Load();

                operations =
operation.InverseFkOperationSecondaryNavigation.ToList();
            }
            else
            {
                context.Entry(operation)
```

```
                    .Collection(o => o.InverseFkOperationLeadNavigation)
                    .Load();

                operations =
operation.InverseFkOperationLeadNavigation.ToList();
            }

            foreach (Operation op in operations)
            {
                if (ActTypesConst.Pledge.Equals(op.ActType) &&
op.Status.Equals("Успішно")
                    && !ActTypesConst.Pledge.Equals(initialAct))
                {
                    return false;
                }
            }

            foreach (Operation op in operations)
            {
                level += 1;
                return TravelOperationTreeSecondary(op, level, initialAct);
            }

            return true;
        }
    }
}
```

ДОДАТОК Б
ПРОЄКТ ІНТЕРФЕЙСУ

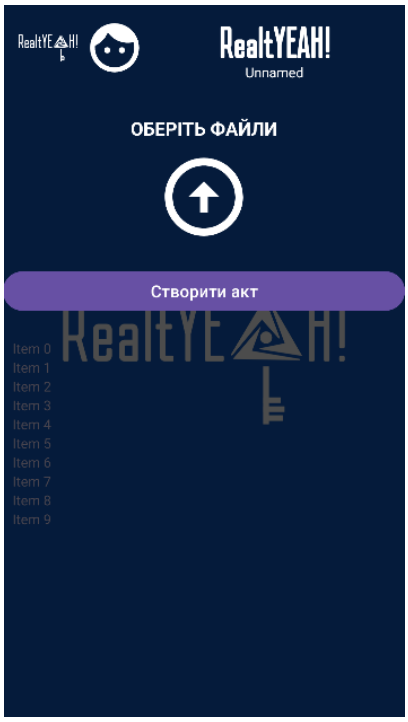


Рисунок Б.1 – Проєкт сторінки прив’язування документів

а б

Рисунок Б.2 – Проєкт сторінки клієнта

Рисунок Б.3 – Проєкт сторінки пошуку клієнтів

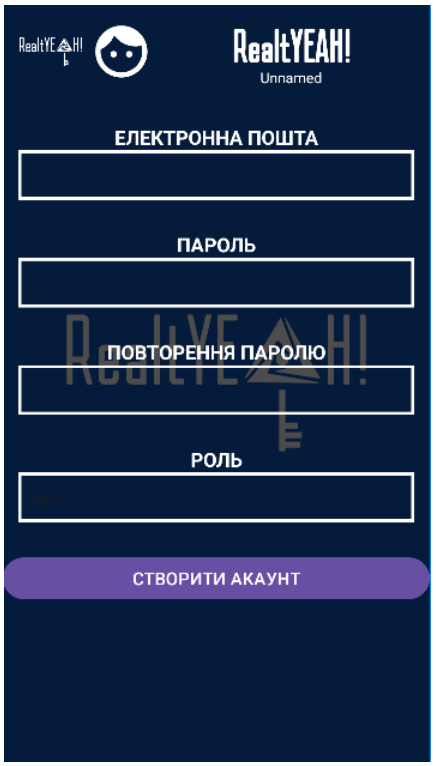
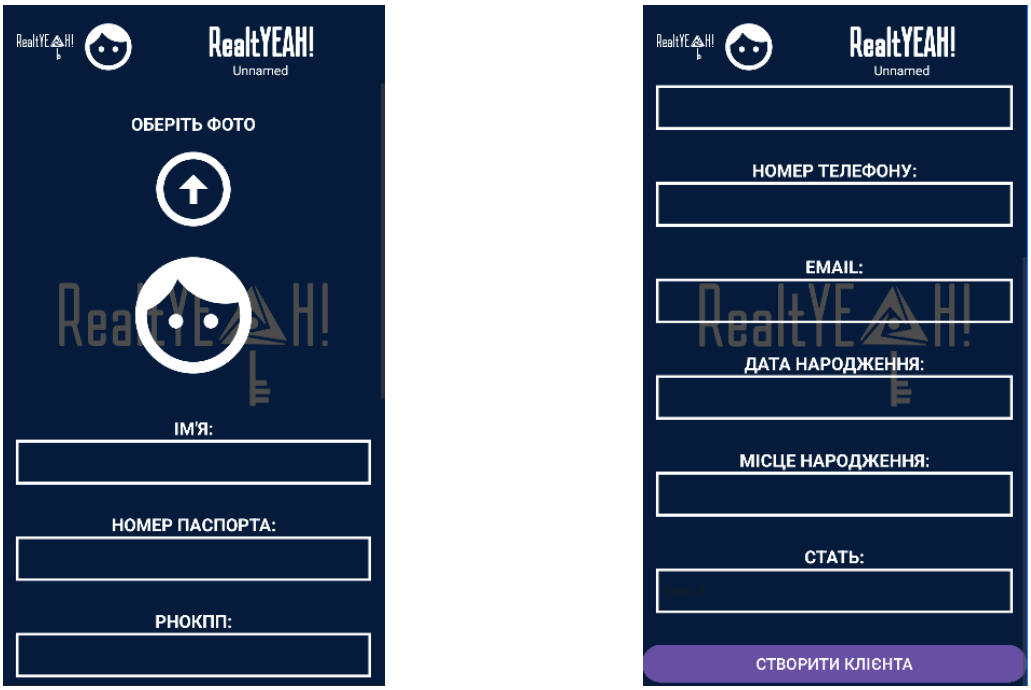
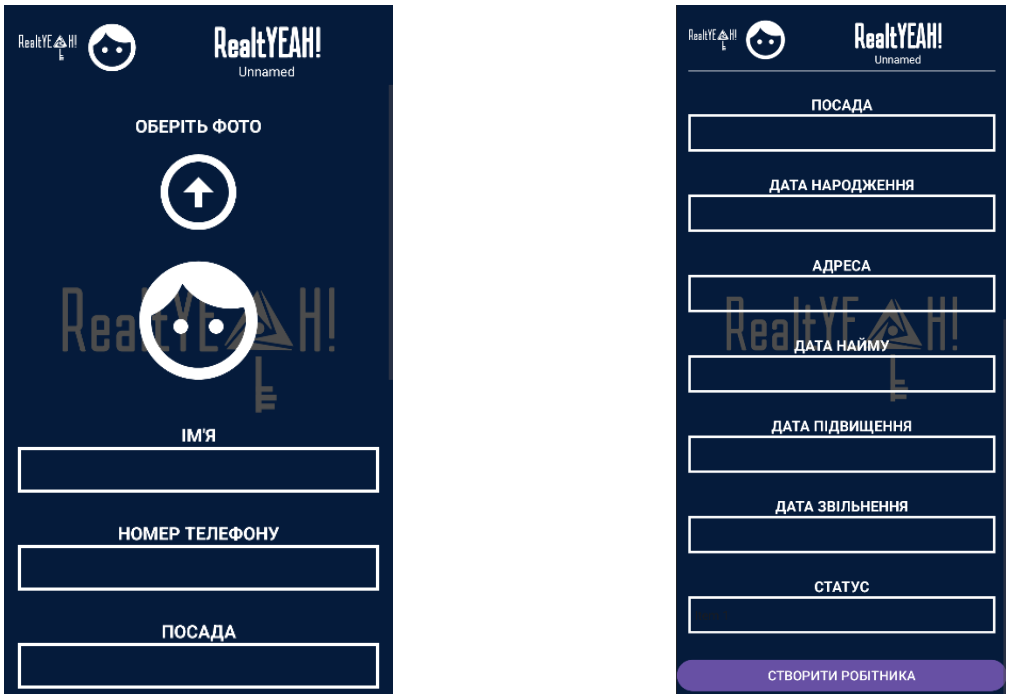


Рисунок Б.4 – Проєкт сторінки створення акаунту рієлтора



а б

Рисунок Б.5 – Проєкт сторінки створення клієнта



а б

Рисунок Б.6 – Проєкт сторінки створення рієлтора

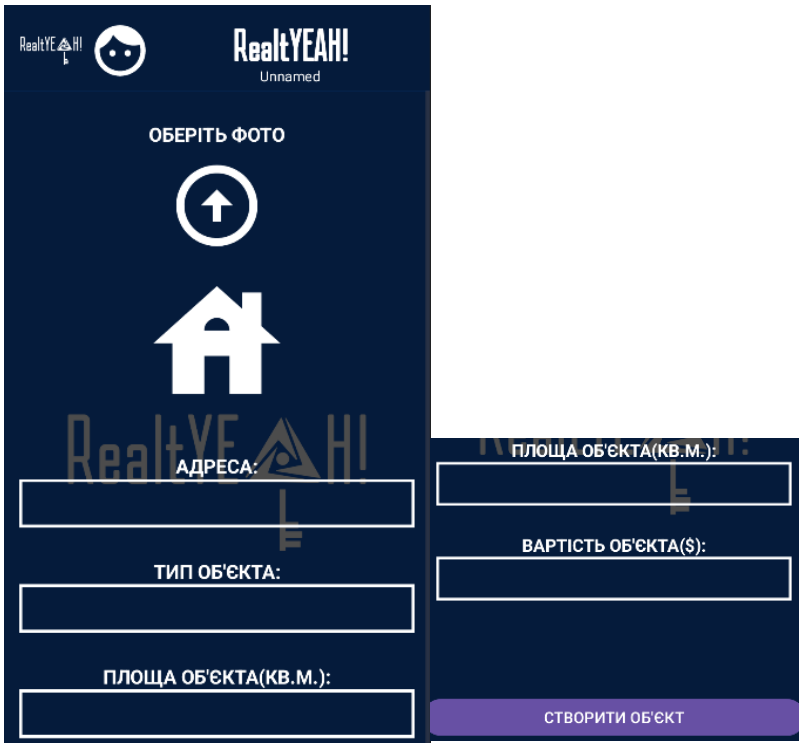


Рисунок Б.7 – Проєкт сторінки створення об’єкта

Рисунок Б.8 – Проєкт сторінки пошуку робітників

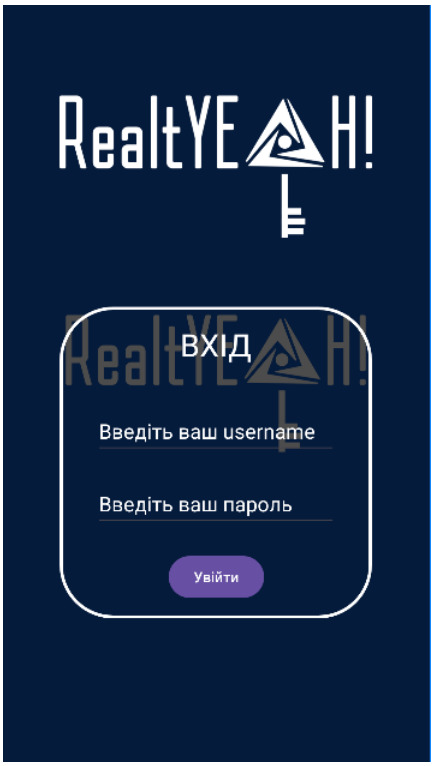


Рисунок Б.9 – Проєкт сторінки авторизації

а

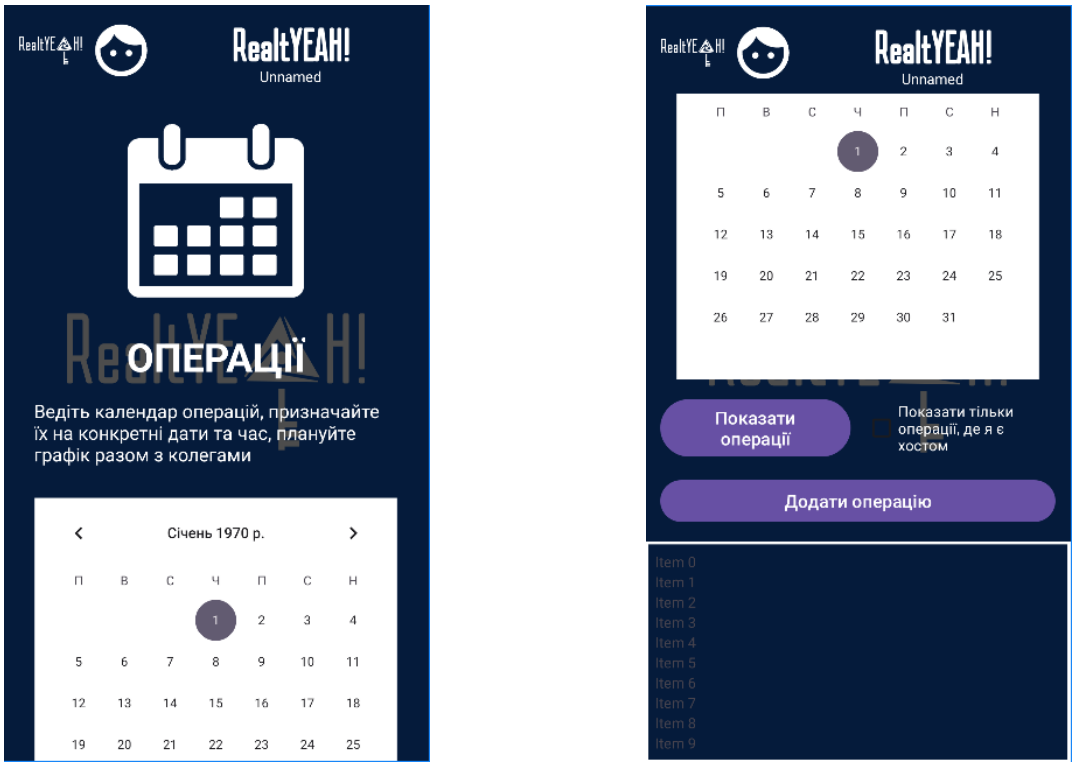
б

Рисунок Б.10 – Проєкт головної сторінки

Рисунок Б.11 – Проєкт сторінки об’єктів нерухомості

Рисунок Б.12 – Проєкт сторінки об’єкта нерухомості

Рисунок Б.13 – Проєкт сторінки пошуку об’єктів



а б

Рисунок Б.14 – Проєкт сторінки операцій

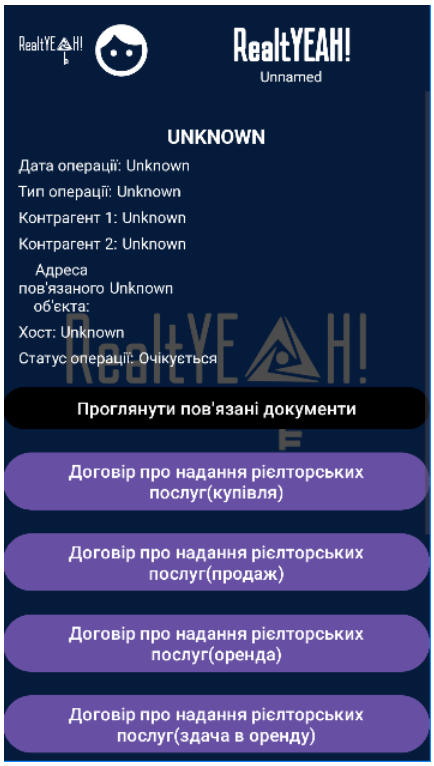


Рисунок Б.15 – Проєкт сторінки операції

Рисунок Б.16 – Проєкт сторінки списку операцій

Рисунок Б.17 – Проєкт сторінки списку документів

Рисунок Б.18 – Проєкт розмітки елемента списку документів для завантаження у додаток

Рисунок Б.19 – Проєкт розмітки елемента списку документів для завантаження на мобільний пристрій

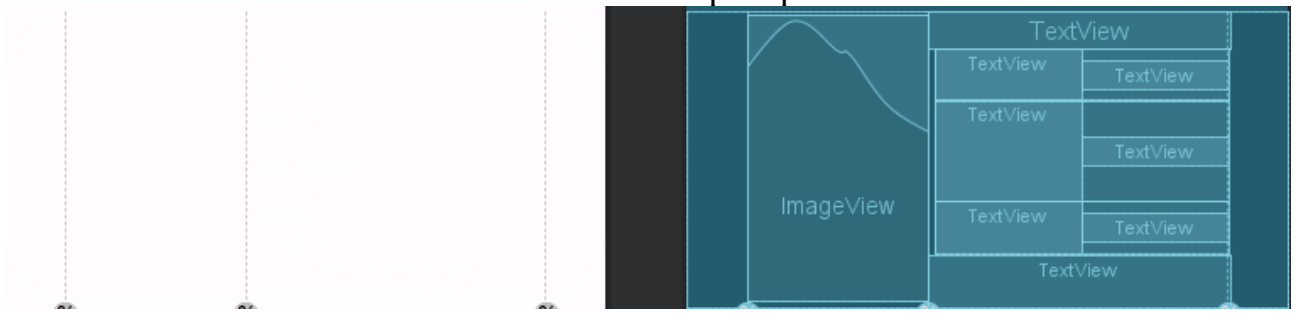


Рисунок Б.20 – Проєкт розмітки елемента списку клієнтів

Рисунок Б.21 – Проєкт розмітки елемента списку робітників

Рисунок Б.22 – Проєкт розмітки елемента списку об’єктів нерухомості

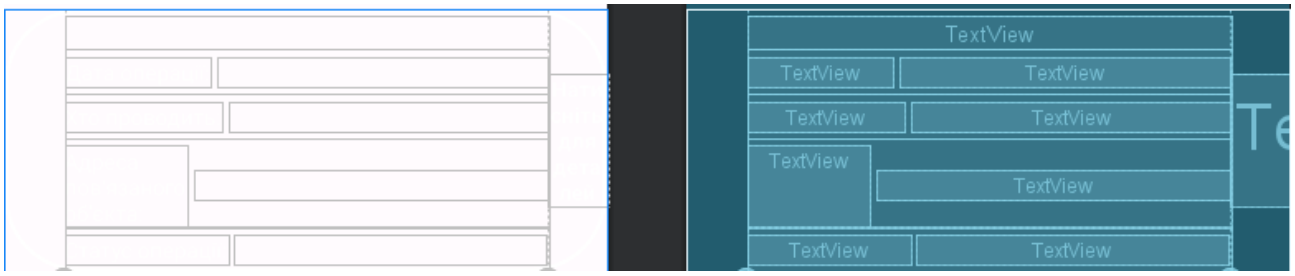


Рисунок Б.23 – Проєкт розмітки елемента списку операцій

Рисунок Б.24 – Проєкт розмітки елемента списку побажань клієнта