**Форма № 24**

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
**Національний університет «Запорізька політехніка»**

Факультет комп'ютерних наук і технологій
(повне найменування факультету)

Кафедра програмних засобів
(повне найменування кафедри )

# Пояснювальна записка

до дипломного проєкту (роботи)
бакалавра
(ступінь вищої освіти)

на тему ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ПРОКАТУ
(назва теми)
МУЗИЧНИХ ІНСТРУМЕНТІВ

MUSICAL INSTRUMENT RENTAL SOFTWARE

Виконав(ла): студент(ка) 4 курсу, групи КНТ-119

Спеціальності 121 Інженерія програмного
(код і найменування спеціальності)
забезпечення

Освітня програма (спеціалізація)
Інженерія програмного забезпечення

КОЗІЙ Д.В.
(ПРІЗВИЩЕ та ініціали)

Керівник СКРУПСЬКИЙ С.Ю.
(ПРІЗВИЩЕ та ініціали)
Рецензент ГОЛУБ Т.В.
(ПРІЗВИЩЕ та ініціали)

2023

**Форма № 25**

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
**Національний університет «Запорізька політехніка»**

Факультет ФКНТ
Кафедра програмних засобів
Ступінь вищої освіти бакалавр
Спеціальність 121 Інженерія програмного забезпечення
(код і найменування)
Освітня програма (спеціалізація) Інженерія програмного забезпечення
(назва освітньої програми (спеціалізації)

**ЗАТВЕРДЖУЮ**
**Завідувач кафедри** ПЗ, д.т.н, проф.
Сергій СУББОТІН
«_____» _____2023 року

# З А В Д А Н Н Я
## НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

КОЗІЯ Дмитра Володимировича
(ПРІЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Програмне забезпечення прокату музичних інструментів. Musical Instrument Rental Software
керівник проєкту (роботи) к.т.н., доцент СКРУПСЬКИЙ Степан Юрійович ,
(науковий ступінь, вчене звання, ПРІЗВИЩЕ, ім'я, по батькові)
затверджені наказом закладу вищої освіти від « 04 » квітня 2023 року № 87
2. Строк подання студентом проєкту (роботи) 12 червня 2023 року
3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

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

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

6. Консультанти розділів проєкту (роботи)

| Розділ | ПРІЗВИЩЕ, ініціали та посада консультанта | Підпис, дата | |
|---|---|---|---|
| | | завдання видав | прийняв виконане завдання |
| 1-4 Основна частина | СКРУПСЬКИЙ С.Ю., доцент | | |
| Нормоконтроль | БЄЛОВА А.В., асистент | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |

7. Дата видачі завдання «__11__» _____квітня_____2023 року.

## КАЛЕНДАРНИЙ ПЛАН

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

**Студент(ка)** _____ __Дмитро КОЗІЙ_____
( підпис ) (Ім'я ПРІЗВИЩЕ)

**Керівник проєкту (роботи)** _____ __Степан СКРУПСЬКИЙ
( підпис ) (Ім'я ПРІЗВИЩЕ)

# РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 94 с., 26 рис., 2 табл., 3 дод., 10 джерел.

ПРОКАТ, МУЗИЧНІ ІНСТРУМЕНТИ, АКСЕСУАРИ, ЗАМОВЛЕННЯ, ЦІНОВА ПРОПОЗИЦІЯ, ВЕБДОДАТОК.

Об’єктом роботи є процес прокату музичних інструментів. Предметом роботи є програмне забезпечення прокату музичних інструментів. Мета роботи – розробити програмне забезпечення прокату музичних інструментів для підтримки взаємодії між власниками інструментів та орендарями.

Матеріали, методи та технічні засоби: мови Python, HTML, JavaScript, система керування базами даних MySQL.

Результати. Програма призначена для забезпечення власників музичних інструментів можливістю надати їх у прокат в той час, коли вони не використовуються, визначаючи параметри цього прокату, включаючи вартість, характеристики музичного інструменту і можливості подальшого придбання музичних інструментів та для забезпечення людей, які шукають можливість взяти музичний інструмент на прокат, такою можливістю.

Висновки. Використання програми може бути корисним як під час навчання грі на музичних інструментах, коли придбання нового інструменту може бути не дуже доцільним, так і під час професійної кар’єри, коли потрібна можливість гнучкого змінювання інструментів для виконання різних партій і такому підходу віддається перевага над персональним володінням одного інструменту або ці підходи використовуються разом. Реалізація програми як вебдодатку сприяє більш широкій підтримці користувачів.

Галузь використання – прокат музичних інструментів, які знаходяться у користуванні.

# ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 94 pages, 26 figures, 2 tables, 3 appendixes, 10 sources.

RENTAL, MUSICAL INSTRUMENTS, ACCESSORIES, ORDERS, PRICE OFFER, WEB APPLICATION.

The object of the work is the process of renting musical instruments. The subject of the work is musical instrument rental software. The goal of the work is to develop musical instrument rental software to support interaction between instrument owners and renters.

Materials, methods and technical means: Python, HTML, JavaScript languages, database management system MySQL.

The results. The program is intended to provide owners of musical instruments with the opportunity to rent them when they are not in use, determining the parameters of this rental, including the cost, characteristics of the musical instrument and the possibility of further purchase of musical instruments, and to provide people, who are looking for the opportunity to take a musical instrument on rental, such an opportunity.

Conclusions. The software can be useful both when learning to play musical instruments, when the purchase of a new instrument may not be very appropriate, and during a professional career, when the ability to flexibly change instruments to perform different parts is needed and this approach is preferred over personal possession of one instrument or these approaches are used together. Implementation of the program as a web application contributes to wider user support.

The field of use is the rental of musical instruments that are in use.

# ЗМІСТ

С.

# ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

БД – база даних;

ПЗ – програмне забезпечення.

# ВСТУП

Сервіси прокату є достатньо популярними у всьому світі. На прокат беруть як преміальні речі, так і достатньо буденні. Самі по собі музичні інструменти важко назвати буденними речами, адже часто інструменти коштують достатньо дорого, відповідно далеко не кожен може дозволити собі придбати новий музичний інструмент, а щонайменше навчатися хочуть багато людей. Прокат музичних інструментів може бути вирішенням цієї проблеми. Однак як в багатьох галузях, прокат від спеціалізованих служб є проблемою, адже такі компанії зазвичай встановлюють певні інтервали цін, а знизити їх орендар ніяк не може. Наявність знижок на тривалий прокат може бути єдиним виходом вплинути на вартість, проте ці тарифи зазвичай є достатньо високими. Така ситуація характерна для різних сфер, де виконується прокат.

Прокат інструментів від власників може вирішити проблему, яку було описано. Однак для контактування з власниками потрібно отримати необхідну інформацію про інструменти, мати можливість дізнатися з відгуків про те, яким був досвід людей, які вже спробували ці можливості. Тобто без відповідного програмного забезпечення (ПЗ) не обійтися при вирішенні цієї проблеми. Зважаючи на це, саме розробка такого ПЗ і є метою даної роботи.

За цією метою на роботу встановлено такі завдання:

– аналіз програмних засобів прокату музичних інструментів і визначення концепції програми;

– проєктування ПЗ прокату музичних інструментів;

– реалізація ПЗ прокату музичних інструментів.

# 1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

## 1.1 Аналіз програмних засобів прокату музичних інструментів

Існують різноманітні програмні засоби прокату музичних інструментів, які мають власні особливості і концепції підтримки прокату.

Додаток Rentmyinstrument [1] (рис. 1.1) передбачає надання інструментів для навчання, тому інструменту визначаються за рівнем вмінь, яким вони відповідають.

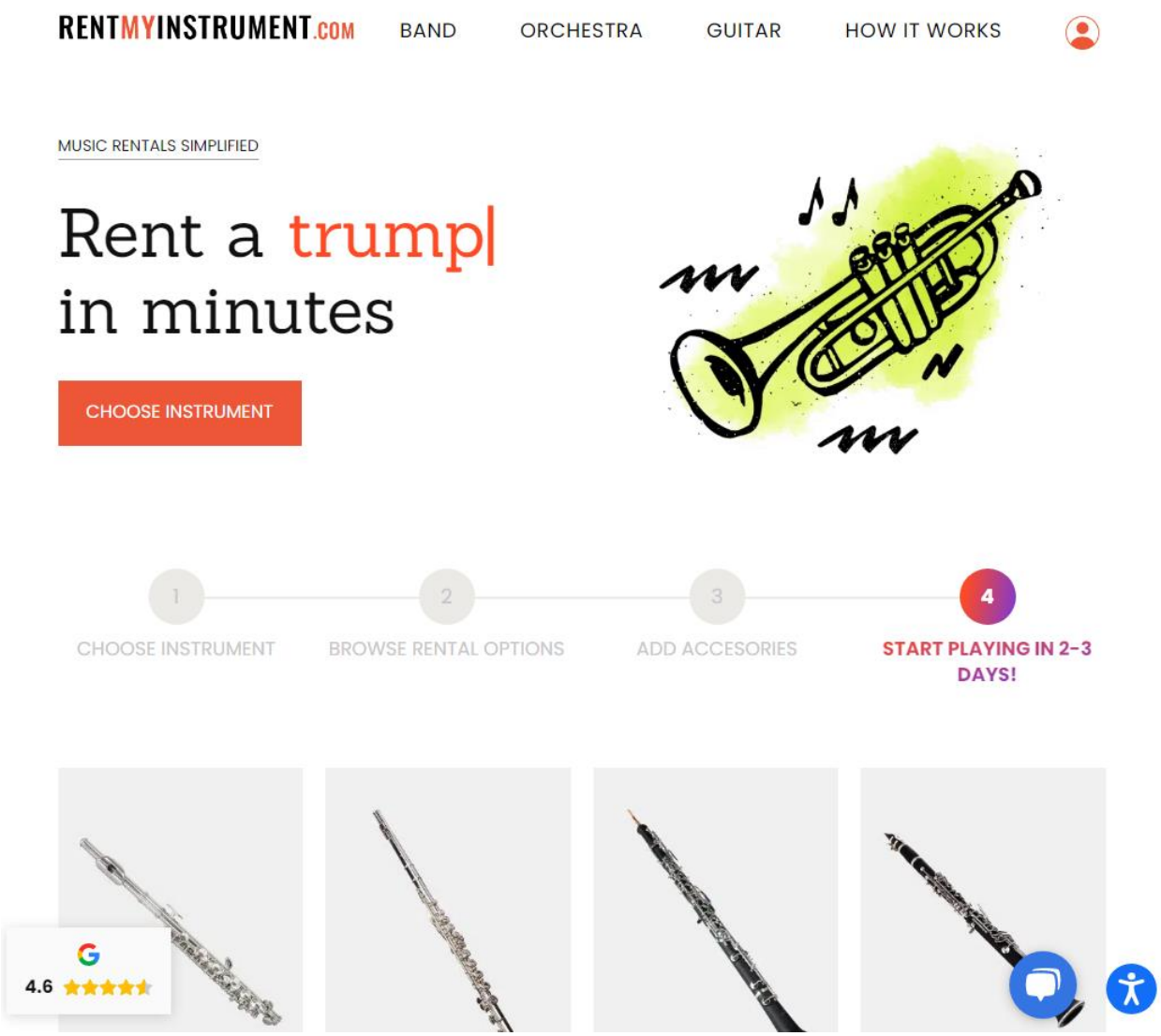


Рисунок 1.1 – Додаток Rentmyinstrument

Прокат відбувається з музичних інструментів, які належать даній компанії, але можлива організація доставки. Підтримується можливість подальшого переходу інструменту у власництво орендаря. Тобто для кожного інструменту визначено місячний платіж та кількість таких платежів, яку потрібно виконати, щоб стати в підсумку власником цього інструменту. Для кожного інструменту під час оформлення замовлення пропонується додати аксесуари, які залежать від того, який саме інструмент було обрано.

Додаток Grover [2] (рис. 1.2) також підтримується відповідною компанією, яка надає музичні інструменти, але це вже не інструменти для навчання.

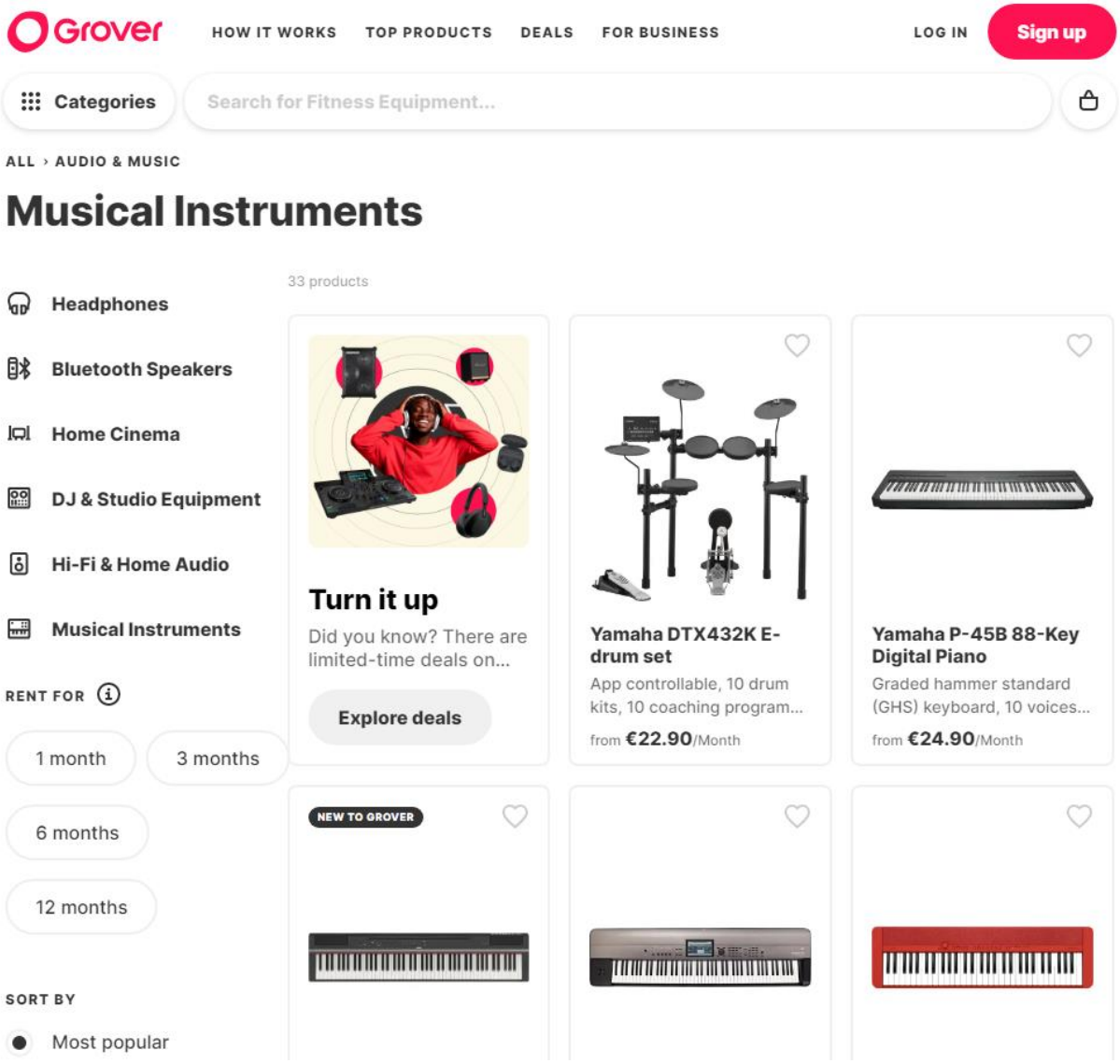


Рисунок 1.2 – Додаток Grover

Grover дозволяє виконувати повноцінний пошук інструментів, а не тільки вибирати інструменти в належності від сімейства, як це є в Rentmyinstrument. Тобто фактично наявна більша кількість інструментів.

У пошуку вказується період прокату, який не може бути меншим за місяць і визначається кількістю місяців, бренд виробника, тип інструменту та інтервал вартості на місяць.

За кожним музичним інструментом вказується можливість придбання його або відсутність такої можливості. Вартість за прокату на більшу кількість місяців зменшується, але визначається тільки кількістю місяців, інші періоди неможливі. Інструменти можна зберігати у власний список бажань.

Додаток Musicalinstrumenthire [3] (рис. 1.3) також підтримується і є компанією з прокату.

Musicalinstrumenthire пропонує як нові інструменти, так і інструменти, в яких були виконані певні зміни (ремонт). Усі інструменти типові, тому можна замовити одразу декілька екземплярів того самого інструменту. За інструментами користувачі залишають відгуки, що дозволяє отримати додаткову інформацію, яка відсутня в попередніх програмах. Також надається можливість придбання інструменту. Мінімальна тривалість прокату – 3 місяці. Особливістю є те, що можна у випадку недоступності інструмента вносити його до свого листа очікування. За інструмент може отримуватися депозит.

Достатньо особливим додатком є Instrurentals. Це також додаток, який підтримує роботу сервісу прокату, але він відзначається тим, що тривалість прокату інструментів може дорівнювати дню, тижню, половині місяця або місяцю. За інструментами також користувачі можуть залишати відгуки.

Однак всі ці додатки не представлені в Україні. Аналіз продемонстрував, що як такого повноцінного додатку з забезпеченням замовлень в Україні немає, хоча і є сервіси прокату музичних інструментів. З них тільки Komora надає можливість ознайомитися з наявними

інструментами і отримати інформацію про вартість прокату на місяць. Вартість стандартна, знижок не передбачається. Замовити інструмент або отримати актуальну інформацію про нього не можна, можна тільки залишити заявку, щоб менеджер зв’язався з користувачем.

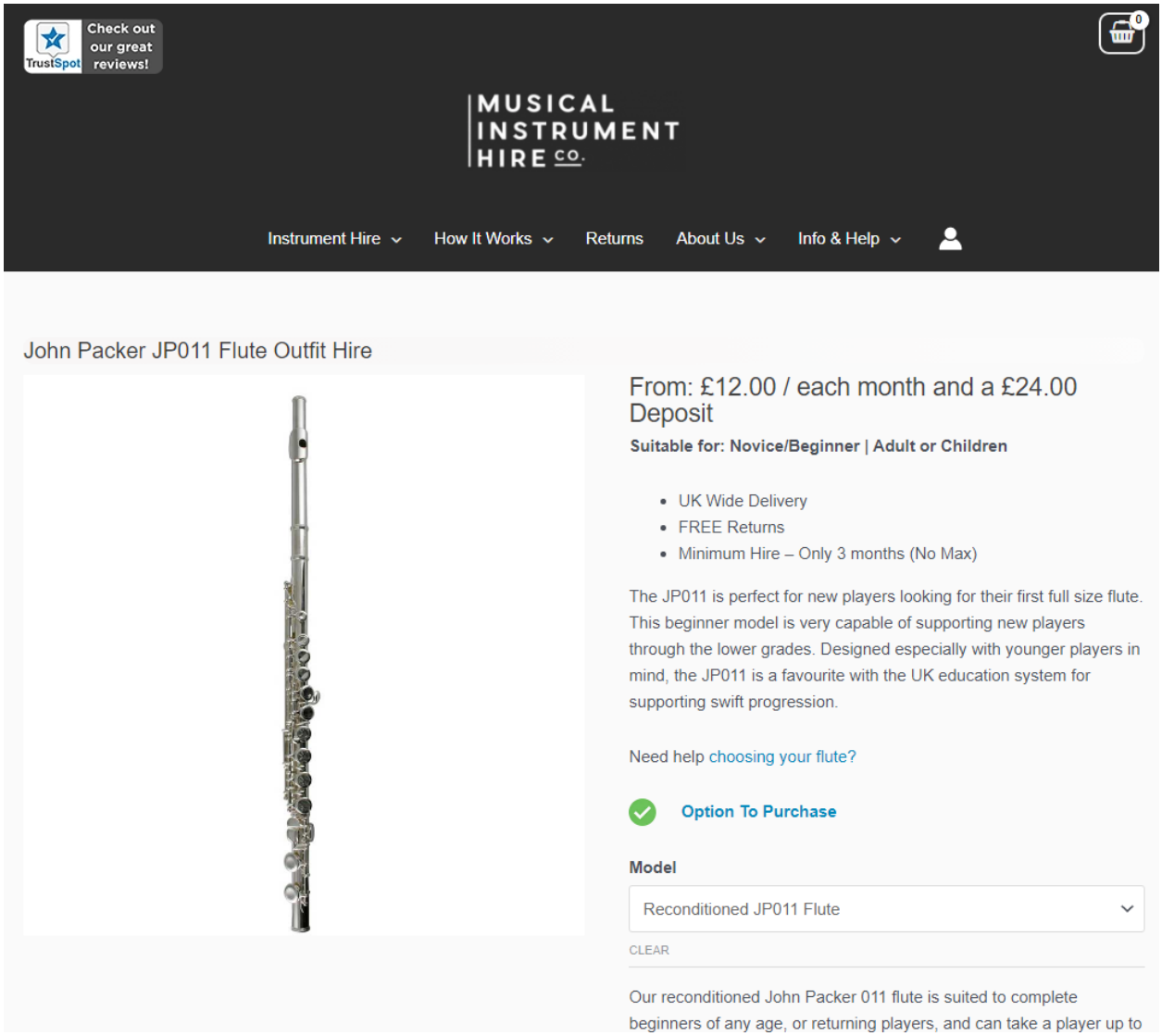


Рисунок 1.3 – Додаток Musicalinstrumenthire

Усі ці додатки підтримують роботу сервісів прокату, а тому відрізняючись деякими особливостями, загалом мають одну й ту саму концепцію.

Єдиним додатком, який надає можливість прокату інструменту від власників, є FriendWithA [4] (рис. 1.4). Головним його недоліком для українських музикантів є те, що він працює тільки в США.

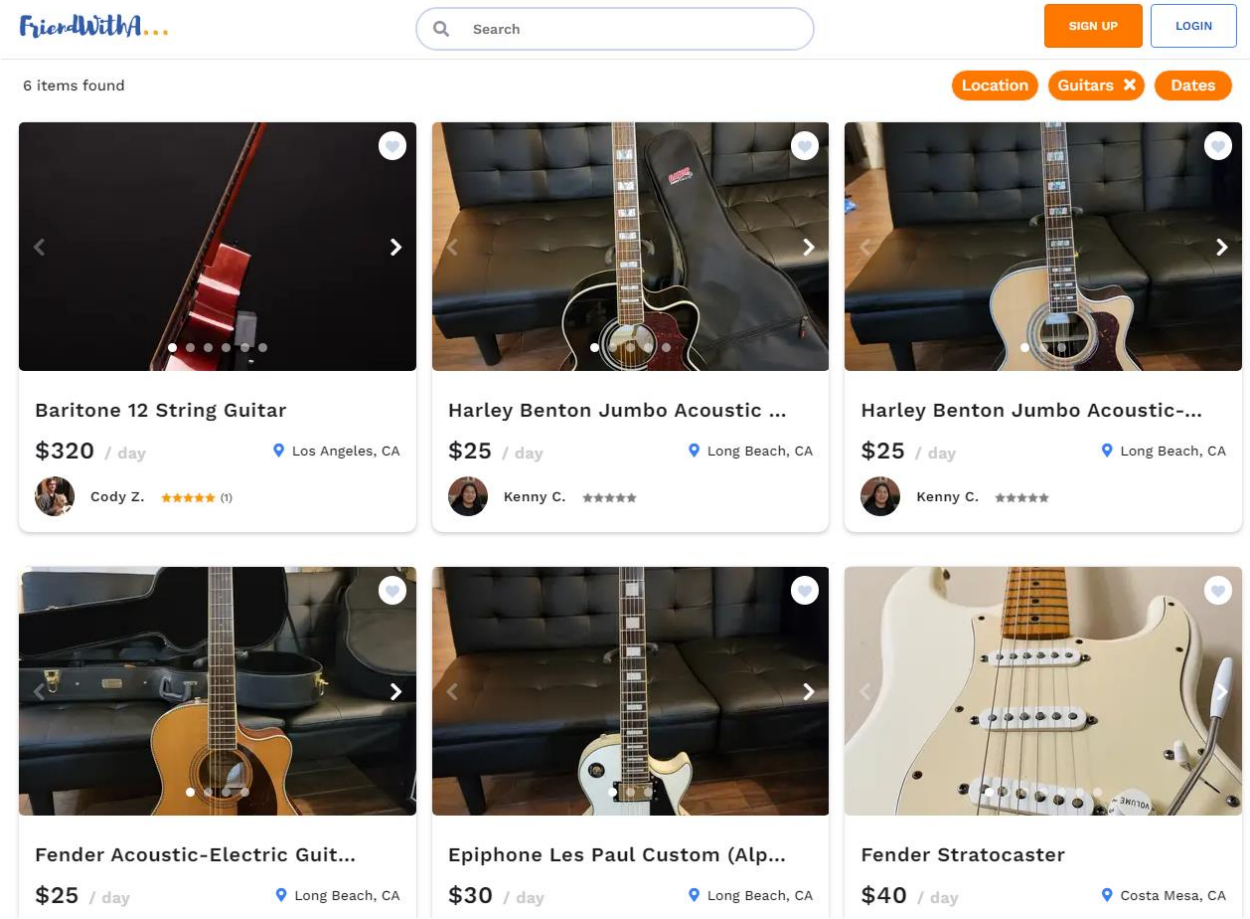


Рисунок 1.4 – Додаток FriendWithA

FriendWithA групує інструменти за їх власниками. За кожним власником надається перелік його інструментів, виставлена оцінка та відгуки. Відгуки формуються за власниками, а не за інструментами. За кожним інструментом виводяться дані про те, скільки разів він був взятий на прокат. Прокат можливий або за денним тарифом, або за тижневим, або за місячним. Період прокату можна вказати будь-який, визначаючи дати початку і закінчення. Надається можливість створювати списки бажань, а також виставляти власні музичні інструменти на прокат.

## 1.2 Визначення концепції розроблюваної програми

Проведений аналіз було зведено до порівняння (табл. 1.1), у якому використано Rentmyinstrument, Grover, FriendWithA та програму CozyMIRS, яка розробляється.

Таблиця 1.1 – Порівняння програмних засобів прокату музичних інструментів

| Критерій | Rentmy-instrument | Grover | FriendWithA | CozyMIRS |
|---|---|---|---|---|
| Концепція | Служба прокату | Служба прокату | Прокат від власників | Прокат від власників |
| Територія дії | США (Канзас, служба доставки) | Німеччина, Нідерланди, Іспанія, США | США | Україна |
| Пошук інструментів | Відсутній | Присутній | Присутній | Присутній |
| Відгуки про інструменти | Відсутні | Присутні | Відсутні | Присутні |
| Списки бажання | Відсутні | Присутні | Присутні | Присутні |
| Списки очікування | Відсутні | Відсутні | Відсутні | Присутні |
| Період прокату | Місяць | Місяць | День, тиждень, місяць | Від одного дня |
| Придбання інструменту | Можливе | Можливе | Можливе | Можливе |

Розроблювана програма базується як і FriendWithA на прокаті від власників, але на відміну від останньої та всіх інших вона направлена на реалізацію цих можливостей в Україні. Програма має надавати можливість залишати відгуки про інструменти подібно до Grover, адже в FriendWithA відгуки можуть бути тільки про власників. У CozyMIRS мають бути відгуки і про інструменти, і про власників. Програма також має надавати можливості створення списків бажань та очікування, які не у всіх подібних програмах існують. Періодом прокату може бути будь-який, починаючи від 1 дня, що подібно до FriendWithA, але при цьому власник може визначити знижки більш гнучко: наприклад, вартість прокату за 5 днів може бути сумарно меншою ніж вартість одного дня, помножена на 5.

### 1.3 Висновки за розділом 1

Виконаний аналіз програмних засобів, які використовуються для підтримки прокату музичних інструментів, дозволив виявити цілий ряд програм, включаючи зокрема Rentmyinstrument, Grover, FriendWithA, які при цьому можна розбити на 2 групи: переважно це служби прокату з власними музичними інструментами, а представником другої групи є FriendWithA, який орієнтований на підтримку прокату музичних інструментів від власників. Саме цю концепцію і взято за основу в програмі в даній роботі, проте з наданням деяких додаткових функцій та з врахуванням, що прокат у FriendWithA не доступний музикантами з України, а тому слід сказати, що повноцінний ринок прокату музичних інструментів в Україні на даний момент відсутній.

# 2 РОЗРОБКА АРХІТЕКТУРИ ПРОГРАМИ

## 2.1 Вибір мови розробки

Мовою реалізації програми має бути Python [5]-[6], яку було порівняно з найпопулярнішою мовою розробки – PHP. Такий вибір є наслідком того що мова Python має перевагу в довготривалих проєктах, адже подібні системи мають довгий життєвий цикл, а python дозволяє вносити в код достатньо просто в тому числі структурні зміни: наприклад, можна додавати в будь-який потрібний момент до об’єктів необхідні поля, тобто немає базової структури, яку потім постійно потрібно використовувати, вона може в будь-який момент, коли це стає потрібним, розширюватися. Це сприяє кращому налагодженню коду та більш простій підтримці. Враховуючи це, а також швидкість написання програми, на противагу яким не найвища продуктивність, мова Python стала остаточним вибором (табл. 2.1).

Таблиця 2.1 – Вибір мови для реалізації програми

| Критерій | Python | PHP |
|---|---|---|
| Перевага в довготривалих проєктах | Так | Ні |
| Продуктивність | Висока | Вища |
| Налагодження | Швидше | Довше |
| Підтримка коду | Легша | Складніша |
| Швидкість написання коду | Вища | Висока |

## 2.2 Моделювання вимог до програми

Функціональні вимоги мають реалізовуватися в програмі для забезпечення роботи орендаря та власника музичних інструментів (рис. 2.1).

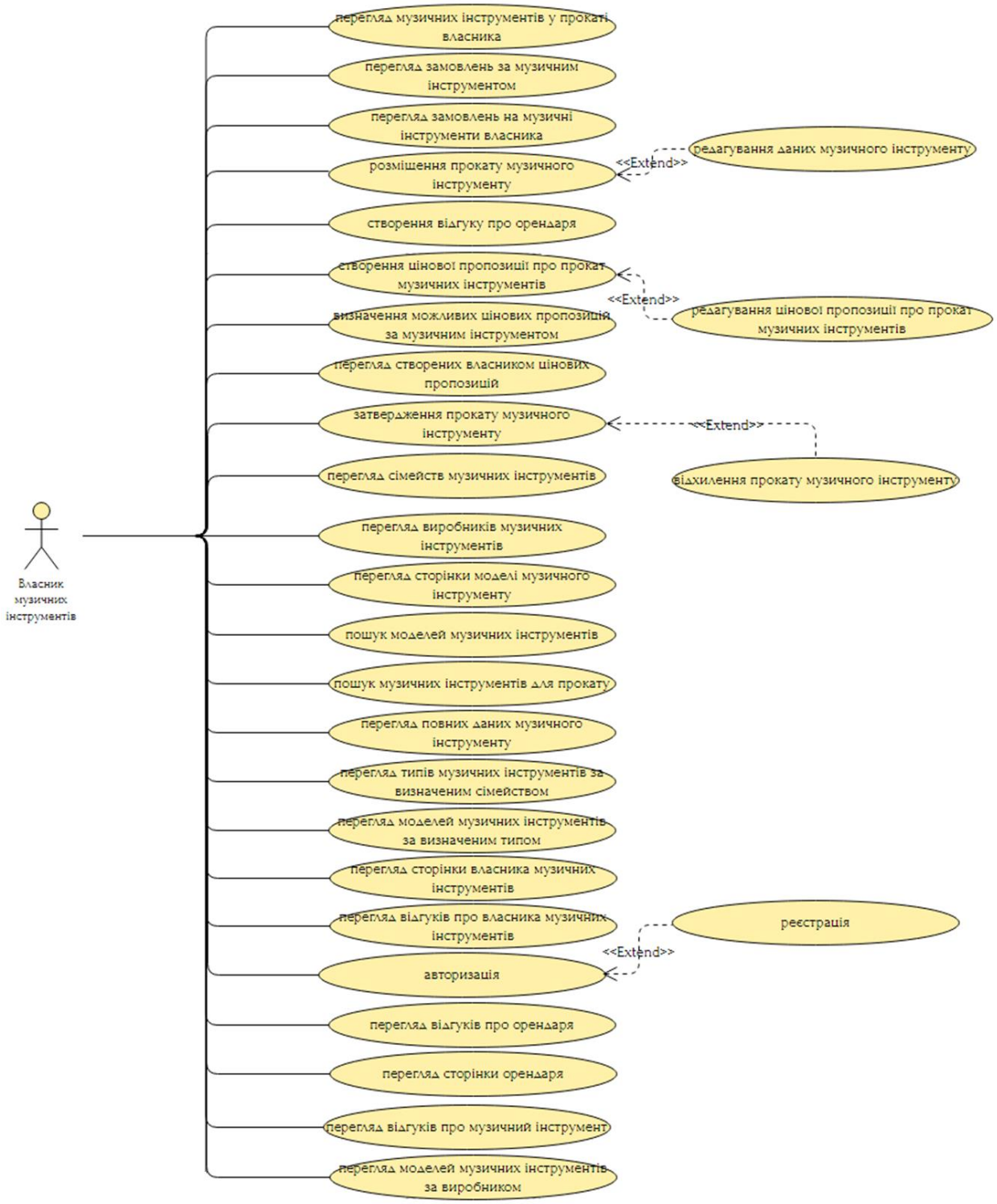


Рисунок 2.1 – Діаграма варіантів використання програми власником музичних інструментів

Орендар (рис. 2.2) може працювати з власними замовленнями, замовляти прокат потрібних йому музичних інструментів та відкладати певні з них, що до них повернутися.

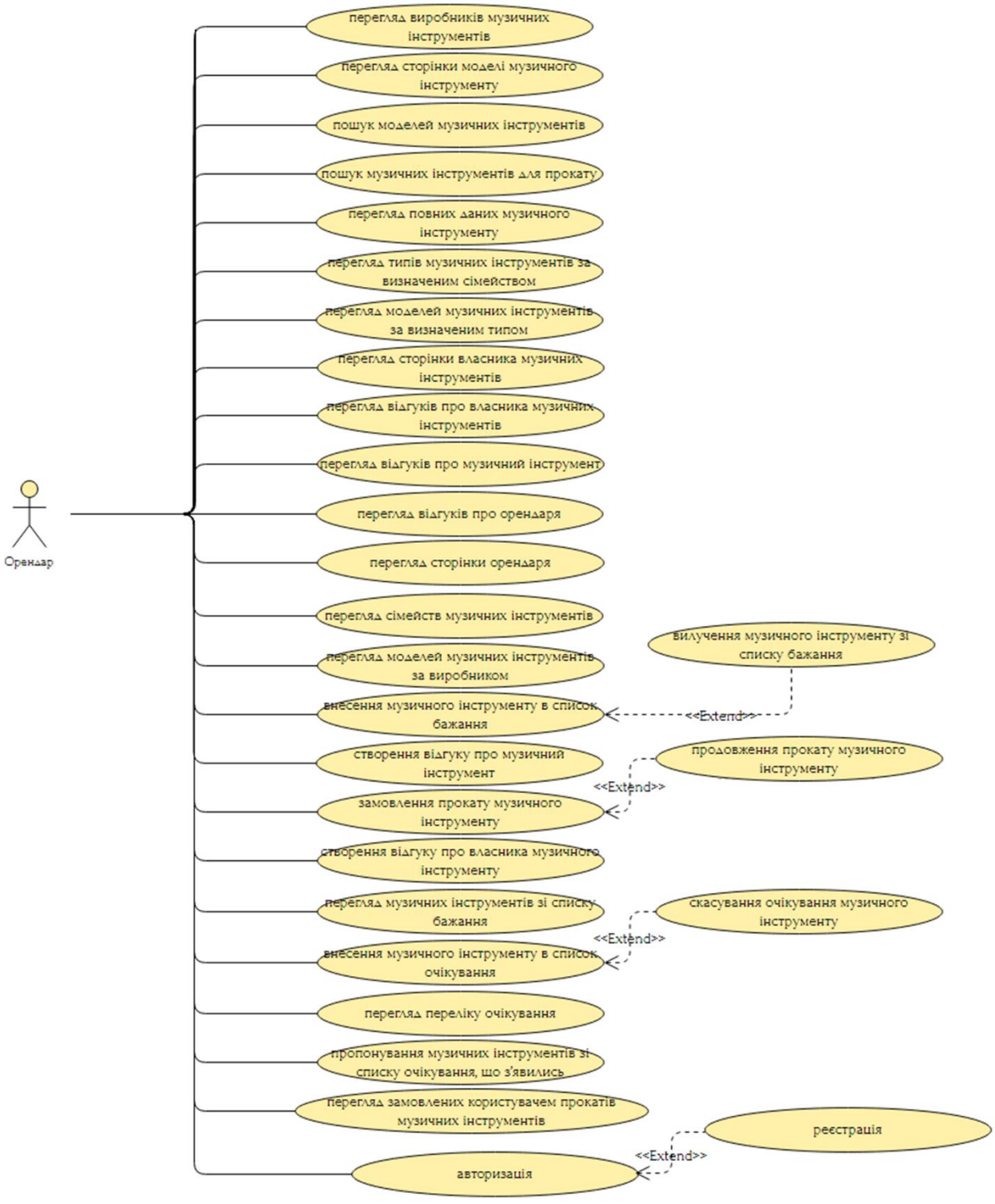


Рисунок 2.2 – Діаграма варіантів використання програми орендарем

У результаті функціональні вимоги включають:

– перегляд сторінки моделі музичного інструменту;

– пошук моделей музичних інструментів;

– пошук музичних інструментів для прокату;

– перегляд повних даних музичного інструменту;

– розміщення прокату музичного інструменту;

– перегляд музичних інструментів у прокаті власника;

– замовлення прокату музичного інструменту;

– внесення музичного інструменту в список бажання;

– перегляд музичних інструментів зі списку бажання;

– вилучення музичного інструменту зі списку бажання;

– внесення музичного інструменту в список очікування;

– скасування очікування музичного інструменту;

– пропонування музичних інструментів зі списку очікування, що з’явились;

– перегляд замовлених користувачем прокатів музичних інструментів;

– перегляд замовлень за музичним інструментом;

– перегляд замовлень на музичні інструменти власника;

– продовження прокату музичного інструменту;

– створення відгуку про музичний інструмент;

– створення відгуку про власника музичного інструменту;

– створення відгуку про орендаря;

– перегляд сторінки власника музичних інструментів;

– перегляд відгуків про власника музичних інструментів;

– перегляд відгуків про музичний інструмент;

– перегляд відгуків про орендаря;

– перегляд сторінки орендаря;

– створення цінової пропозиції про прокат музичних інструментів;

– редагування цінової пропозиції про прокат музичних інструментів;

– визначення можливих цінових пропозицій за музичним інструментом;

– перегляд створених власником цінових пропозицій;

– редагування даних музичного інструменту;

– затвердження або відхилення прокату музичного інструменту;

– перегляд сімейств музичних інструментів;

– перегляд типів музичних інструментів за визначеним сімейством;

– перегляд моделей музичних інструментів за визначеним типом;

– перегляд виробників музичних інструментів;

– перегляд моделей музичних інструментів за виробником;

– перегляд переліку очікування;

– авторизація і реєстрація як орендарів, так і власників музичних інструментів.

### 2.3 Проєктування бази даних

База даних (БД) організована на основі системи управління БД MySQL [7]-[9] з визначенням відповідної структури.

Сімейства (види) музичних інструментів знаходяться в таблиці Instrumentfamily з полями:

– id є ідентифікатором сімейства музичних інструментів;

– family є назвою сімейства музичних інструментів;

– note

– photo

Наприклад, сімейством музичних інструментів є саксофон, а вже альт-саксофон відноситься до типу музичних інструментів.

Типи музичних інструментів знаходяться в таблиці Instrumenttype з полями:

– id є ідентифікатором типу музичних інструментів;

– instrumentfamily є зовнішнім ключем, який визначає сімейство музичних інструментів, до якого належить цей тип інструментів, зв’язаний з таблицею Instrumentfamily;

– typename є назвою цього типу музичних інструментів;

– note є коротким описом цього типу музичних інструментів;

– photo є фотографією, яка ілюструє цей тип музичних інструментів;

– guide є керівництвом з обслуговування цього типу музичного інструменту, тобто описує дії, які необхідно виконувати і правила, які потрібно підтримувати, щоб інструмент продовжував працювати без необхідності ремонту.

За власними музичними інструментами користувач може визначити для них, що певні з них є аксесуарами для інших. Для цього використовується таблиця Accessory з полями:

– baseinstrument є зовнішнім ключем, який є музичним інструментом, для якого визначається аксесуар, зв’язаний з таблицею Musicalinstrument;

– accessoryinstrument є зовнішнім ключем, який є музичним інструментом, що виконує роль аксесуара для даного інструмента, зв’язний з таблицею Musicalinstrument.

Музичні бренди, які представляють виробників музичних інструментів, знаходяться в таблиці Instrumentbrand з полями:

– id є ідентифікатором бренду виробника музичного інструменту;

– name є назвою бренду виробника музичного інструменту.

Моделі музичних інструментів знаходяться в таблиці Instrumentmodel з полями:

– id є ідентифікатором моделі музичного інструменту;

– instrumenttype є зовнішнім ключем, який визначає тип музичного інструменту, до якого відноситься ця модель, зв’язаний з таблицею Instrumenttype;

– instrumentbrand є зовнішнім ключем, який визначає бренд виробника музичного інструменту, до якого відноситься ця модель, зв’язаний з таблицею Instrumentbrand;

– model є назвою моделі, до якої належить цей музичний інструмент;

– note є коротким описом цієї моделі музичного інструменту.

Музичні інструменти, якими вже володіють люди, які віддають їх на прокат у програмі, знаходяться в таблиці Musicalinstrument з полями:

– id є ідентифікатором музичного інструменту;

– instrumentmodel є зовнішнім ключем, який визначає модель музичного інструменту, до якого відноситься цей інструмент, зв’язаний з таблицею Instrumentmodel;

– note є описом цього музичного інструменту, який надав його власник;

– city є зовнішнім ключем, який визначає місто, де знаходиться музичний інструмент, зв’язаний з таблицею City;

– owner є зовнішнім ключем, який визначає власника цього музичного інструменту, зв’язаний з таблицею Owner;

– condition є числом, яке визначає стан музичного інструменту, що відповідає рівню його зношеності;

– buyoption визначає можливість придбання музичного інструменту або можливість тільки взяти його на прокат на певний період;

– price є вартістю, за яку цей музичний інструмент можна придбати (якщо таке допустимо);

– leasingduration є тривалістю оренди, після якої можна придбати цей музичний інструмент;

– leasingpercent є відсотком знижки від вартості, який отримує орендар, який протягом вказаної в полі leasingduration тривалості оренди брав його на прокат;

– leasingoption визначає можливість тривалої оренди музичного інструменту, після якої його можна буде викупити;

– deadline визначає кінцеву дату, після якої музичний інструмент має бути обов’язково повернутий власнику: необхідне для випадків, коли власник не бажає продавати інструмент, позбуватися його, а вирішив тільки протягом деякого періоду часу надати його на прокат;

– actual визначає актуальність цієї інформації: якщо музичний інструмент проданий, то інформація одразу перестає бути актуальною, також це відбувається, якщо користувач вирішив перестати надавати музичний інструмент на прокат;

– fromlevel є мінімальним рівнем вмінь, за якого, на думку власника музичного інструменту, варто його використовувати;

– tolevel є максимальним рівнем вмінь, за якого, на думку власника музичного інструменту, варто його використовувати;

– reviewed є середньою оцінкою цього музичного інструменту, яку йому виставили орендарі;

– deposit є сумою, яку потрібно внести в якості депозиту під час прокату цього музичного інструменту.

Міста, в яких відбувається пошук, знаходяться в таблиці City з полями:

– id є ідентифікатором міста;

– country є назвою країни, де знаходиться місто;

– name є назвою міста.

Кожен музичний інструмент може бути представлений різними фотографіями, зробленими його власником. Вони знаходяться в таблиці Instrumentphoto з полями:

– id є ідентифікатором фотографії музичного інструменту;

– musicalinstrument є зовнішнім ключем, який визначає музичний інструмент, який проілюстровано на фотографії, зв’язаний з таблицею Musicalinstrument;

– photo є самою фотографією музичного інструменту.

Усі моделі музичних інструментів мають певні властивості. Це може бути, наприклад, матеріал грифу для гітари. У той же час інший музичний інструмент не має грифу взагалі. Різні моделі можуть мати різні значення властивостей, хоча наприклад, всі гітари мають гриф.

Самі властивості, які описують характеристики музичних інструментів, знаходяться в таблиці Property з полями:

– id є ідентифікатором властивості;

– name є назвою властивості (рис. 2.3).

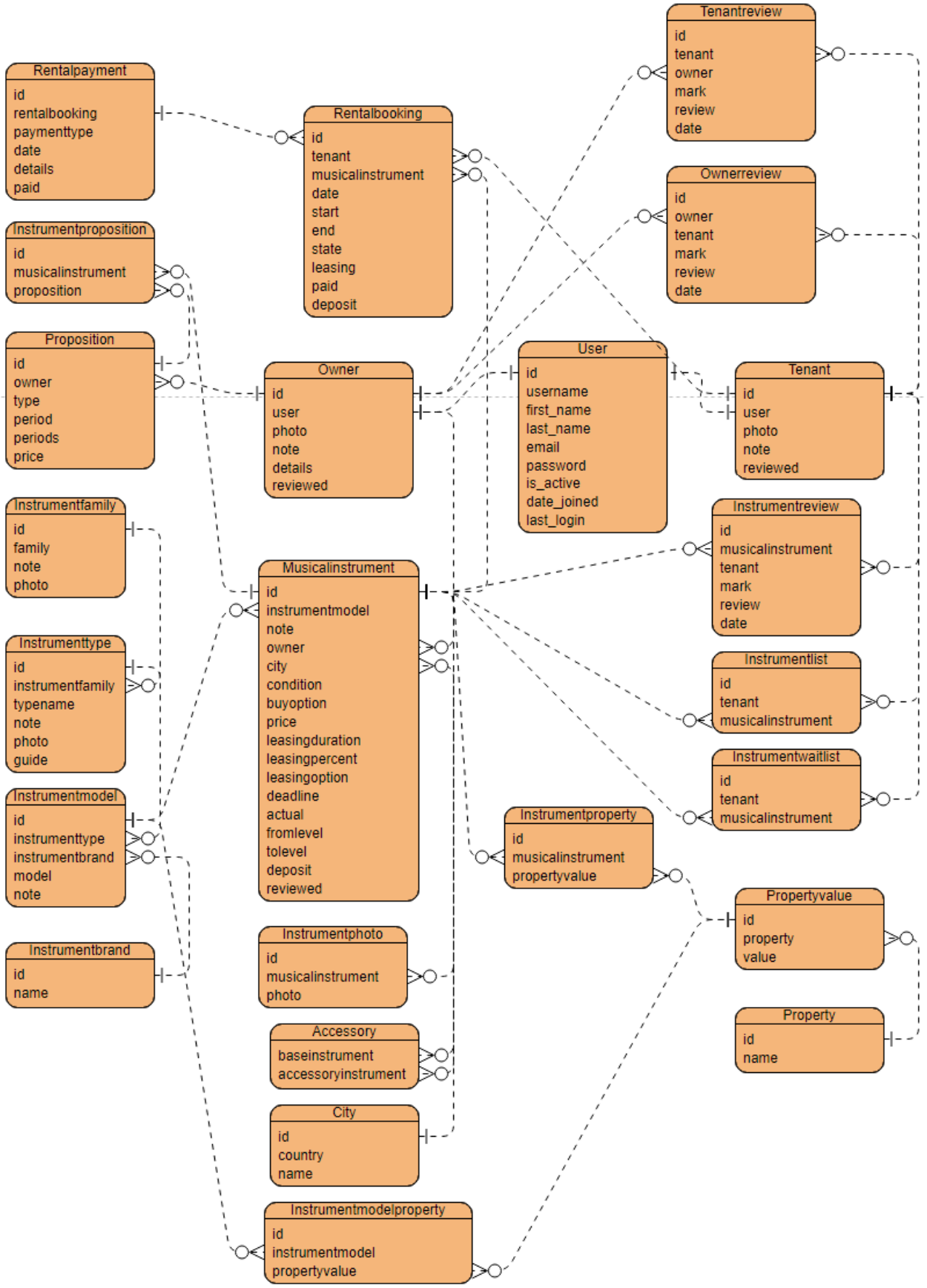


Рисунок 2.3 – Схема БД

Кожна властивість може мати деякі значення. Це значення з певного набору. Дані про це знаходяться в таблиці Propertyvalue з полями:

– id є ідентифікатором можливого значення властивості;

– property є зовнішнім ключем, який визначає властивість, якій відповідає це можливе значення, зв’язаний з таблицею Property;

– value є конкретним можливим значенням.

Самі значення встановлюються для конкретної моделі музичного інструменту. Для неї визначається, якими значеннями своїх властивостей вона характеризується або може характеризуватися, якщо таких значень може бути декілька. Дані про це знаходяться в таблиці Instrumentmodelproperty з полями:

– id є ідентифікатором значення властивості, що відповідає заданій моделі музичного інструменту;

– instrumentmodel є зовнішнім ключем, який визначає модель музичного інструменту, до якої відноситься це значення властивості, зв’язаний з таблицею Instrumentmodel;

– propertyvalue є зовнішнім ключем, який визначає значення властивості, яке відповідає цій моделі музичного інструменту, зв’язаний з таблицею Propertyvalue.

Деякі властивості можуть характеризувати не модель, а конкретний музичний інструмент. Прикладом такої властивості є колір. Наприклад, моделі електрогітар можуть мати різний колір, а конкретний колір стосується вже конкретного музичного інструменту. Дані про властивості музичного інструменту знаходяться в таблиці Instrumentproperty з полями:

– id є ідентифікатором значення властивості, що відповідає заданому музичному інструменту;

– musicalinstrument є зовнішнім ключем, який визначає музичний інструмент, до якого відноситься це значення властивості, зв’язаний з таблицею Musicalinstrument;

– propertyvalue є зовнішнім ключем, який визначає значення властивості, яке відповідає цьому музичному інструменту, зв’язаний з таблицею Propertyvalue.

Акаунти користувачів, що використовуються для ідентифікації в системі, знаходяться в таблиці User з полями:

– id є ідентифікатором акаунта користувача;

– username є логіном акаунта;

– first_name є іменем користувача, який використовує цей акаунт;

– last_name є прізвищем користувача, який використовує цей акаунт;

– email є електронною поштою користувача;

– password є паролем акаунта;

– is_active визначає, чи може користувач використовувати цей акаунт у програмі;

– date_joined є датою додавання користувача до програми;

– last_login є датою останньої авторизації користувача в програмі.

Безпосередньо дані власників, тобто користувачів, які можуть надавати на прокат музичні інструменти, знаходяться в таблиці Owner з полями:

– id є ідентифікатором власника;

– user є зовнішнім ключем, який визначає акаунт, який використовує власник, зв’язаний з таблицею User;

– photo є фотографією власника;

– note є описом себе від власника;

– details є банківськими деталями власника;

– reviewed є середньою оцінкою цього власника музичних інструментів, яку йому виставили орендарі.

Орендарі ж виділені в окрему таблицю, яка називається Tenant і має поля:

– id є ідентифікатором орендаря;

– user є зовнішнім ключем, який визначає акаунт, який використовує власник, зв’язаний з таблицею User;

– photo є фотографією орендаря;

– note є описом себе від орендаря;

– reviewed є середньою оцінкою цього орендаря, яку йому виставили власники музичних інструментів, які він орендував раніше.

Цінові пропозиції щодо прокату музичних інструментів можуть повторюватися для різних музичних інструментів власника, але визначаються окремим власником для себе, адже це не система, яка є окремим сервісом, який надає свої власні інструменти, а джерелом інструментів є самі окремі власники. Відповідно цінові пропозиції щодо прокату музичних інструментів знаходяться в таблиці Proposition з полями:

– id є ідентифікатором цінової пропозиції;

– owner є зовнішнім ключем, який визначає власника, який визначив для формування правил стосовно своїх музичних інструментів цю цінову пропозицію, зв’язаний з таблицею Owner;

– type є типом пропозиції: таких пропозицій дві, перша стосується початкового бронювання, а друга – продовження бронювання, коли фактично виконуються дії, які можуть завершитись придбанням інструменту, якщо воно допускається, при цьому продовження вже відбувається не на весь можливий строк, а на певні періоди, наприклад, у місяць;

– period є одиницею виміру періоду (день, тиждень, місяць, рік);

– periods є кількістю одиниць, за якої починає діяти тариф;

– price є вартістю за цей період.

Цінові пропозиції визначаються самим власником, але далі вони мають бути співставлені з конкретними музичними інструментами, адже не всі вони можуть застосовуватися до всіх його музичних інструментів, якщо їх декілька. Зв’язок між ціновими пропозиціями власника і його музичними інструментами знаходяться в таблиці Instrumentproposition з полями:

– id є ідентифікатором цінової пропозиції для музичного інструменту;

– musicalinstrument є зовнішнім ключем, який визначає музичний інструмент, до якого застосовується ця цінова пропозиція, зв’язаний з таблицею Musicalinstrument;

– proposition є зовнішнім ключем, який визначає цінову пропозицію, яка застосовується до музичного інструменту, зв’язаний з таблицею Proposition.

Замовлення прокату музичних інструментів знаходяться в таблиці Rentalbooking з полями:

– id є ідентифікатором прокату музичного інструменту;

– tenant є зовнішнім ключем, який визначає орендаря, який замовив цей прокат музичного інструменту, зв’язаний з таблицею Tenant;

– musicalinstrument є зовнішнім ключем, який визначає музичний інструмент, прокат якого замовлено, зв’язаний з таблицею Musicalinstrument;

– date є часом створення замовлення прокату;

– start є датою початку прокату;

– end є датою закінчення прокату;

– state є станом замовлення;

– leasing є позначкою про те, що орендар планує продовжувати прокат музичного інструменту з можливістю в підсумку купівлі цього інструменту;

– paid є сумарною сплаченою сумою за цей прокат загалом;

– deposit є сумою депозиту, яку було внесено за прокат інструменту.

За замовленням можуть здійснюватися послідовні сплати задля його продовження. Вони знаходяться в таблиці Rentalpayment з полями:

– id є ідентифікатором сплати;

– rentalbooking є зовнішнім ключем, який визначає замовлення, сплата за яке виконується, зв’язаний з таблицею Rentalbooking;

– paymenttype є типом платежа, тобто це первинний платіж за прокат, платіж за продовження прокату, платіж за придбання інструменту, платіж за депозит;

– details є банківськими деталями цієї сплати;

– date є часом виконання сплати;

– paid є сплаченою сумою.

Відгуки про музичні інструменти орендарів знаходяться в таблиці Instrumentreview з полями:

– id є ідентифікатором відгуку про музичний інструмент;

– musicalinstrument є зовнішнім ключем, який визначає музичний інструмент, про який цей відгук, зв’язаний з таблицею Musicalinstrument;

– tenant є зовнішнім ключем, який визначає орендаря, який створив цей відгук про музичний інструмент, зв’язаний з таблицею Tenant;

– mark є виставленою оцінкою у відгуку;

– review є текстом відгуку;

– date є датою публікації відгуку.

Відгуки про власників музичних інструментів, які надавали їх на прокат, знаходяться в таблиці Ownerreview з полями:

– id є ідентифікатором відгуку про власника музичного інструменту;

– owner є зовнішнім ключем, який визначає власника музичного інструменту, про якого створений цей відгук, зв’язаний з таблицею Owner;

– tenant є зовнішнім ключем, який визначає орендаря, який створив цей відгук про власника музичного інструменту, зв’язаний з таблицею Tenant;

– mark є виставленою оцінкою у відгуку;

– review є текстом відгуку;

– date є датою публікації відгуку.

Відгуки про орендарів, які створюються власниками музичних інструментів, знаходяться в таблиці Tenantreview з полями:

– id є ідентифікатором відгуку про орендаря;

– owner є зовнішнім ключем, який визначає власника музичного інструменту, який створив цей відгук, зв’язаний з таблицею Owner;

– tenant є зовнішнім ключем, який визначає орендаря, про якого цей відгук, зв’язаний з таблицею Tenant;

– mark є виставленою оцінкою у відгуку;

– review є текстом відгуку;

– date є датою публікації відгуку.

Збереження музичних інструментів у списки бажань знаходяться в таблиці Instrumentlist з полями:

– id є ідентифікатором збереження музичного інструменту в список бажання орендаря;

– tenant є зовнішнім ключем, який визначає орендаря, який зберіг музичний інструмент у свій список бажань, зв’язаний з таблицею Tenant;

– musicalinstrument є зовнішнім ключем, який визначає музичний інструмент, який збережено в список бажання, зв’язаний з таблицею Musicalinstrument.

Якщо музичний інструмент знаходиться у прокаті, але орендар має бажання спробувати дочекатися його, то він може підписуватися на нього. Збереження музичних інструментів у списки очікування знаходяться в таблиці Instrumentwaitlist з полями:

– id є ідентифікатором збереження музичного інструменту в список очікування орендаря;

– tenant є зовнішнім ключем, який визначає орендаря, який зберіг музичний інструмент у свій список очікування, зв’язаний з таблицею Tenant;

– musicalinstrument є зовнішнім ключем, який визначає музичний інструмент, який збережено в список очікування, зв’язаний з таблицею Musicalinstrument.

## 2.4 Висновки за розділом 2

Другий розділ включає вибір мови програмування Python для реалізації, визначення функціональних вимог у вигляді набору сценаріїв власників музичних інструментів і орендарів, структурування БД під роботу з відповідними функціями.

# 3 РЕАЛІЗАЦІЯ ПРОГРАМИ

## 3.1 Порядок функціонування програми

Порядок функціонування програми представлено на основі однієї з центральних сутностей – замовлення прокату музичних інструментів (рис. 3.1).

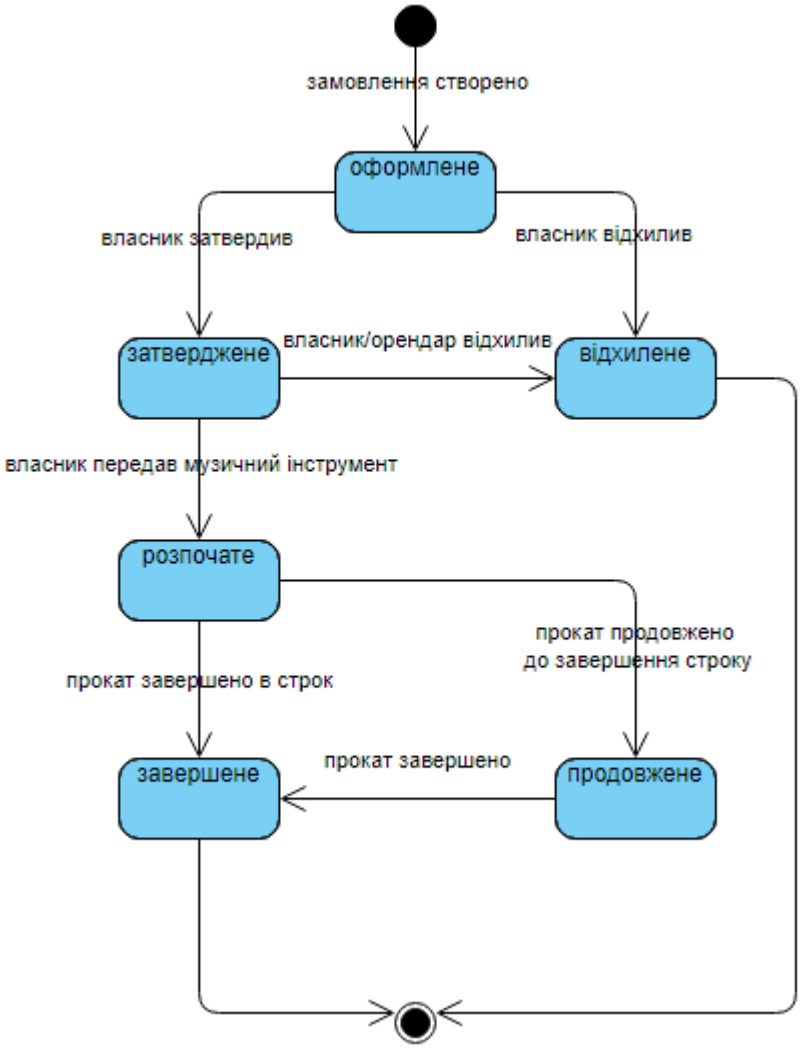


Рисунок 3.1 – Діаграма станів замовлення прокату музичного інструменту

## 3.2 Програмна реалізація вебсистеми

Безпосередньо реалізацію базової роботи з даними виконано через створення класів моделей. Робота безпосередньо за функціональними

вимогами розділена на 2 частини: перша – для базової роботи в межах програми, підтримуючи основні функції орендарів та власників музичних інструментів, а друга – для підтримки замовлень.

Функція owner_instruments_view працює над переглядом музичних інструментів у прокаті власника з використанням створеного шаблону сторінки instruments_owner.html, вона має один основний параметр, який називається owner і може бути не заданий. Якщо він не заданий, то передбачається, що функція підтримує роботу для авторизованого користувача, який є власником музичних інструментів. Тоді за акаунтом вибирається відповідний об’єкт моделі власника, а за ним вибираються всі об’єкти моделі Musicalinstrument, власником яких є цей користувач. Якщо параметр owner заданий, то робиться все те саме, але для заданого власника, який визначається власним логіном.

Перегляд відгуків про власника музичних інструментів реалізований функцією owner_reviews_view, яка також має один основний параметр, який називається owner і може бути не заданий. Тобто підтримує логіку як для роботи будь-якого користувача, в тому числі неавторизованого, так і для власника, який вирішив подивитися відгуки про себе. Після встановлення одним з цих двох варіантів власника відбувається відбір тих відгуків про власників, які написані про цього власника, сортуючи їх за часом публікації від останньої.

Функція tenant_reviews_view працює над переглядом відгуків про орендаря з використанням створеного шаблону сторінки tenant_reviews.html, вона має один основний параметр, який називається tenant і може бути не заданий. Якщо він не заданий, то передбачається, що функція підтримує роботу для авторизованого користувача, який є орендарем. Тоді за акаунтом вибирається відповідний об’єкт моделі орендаря, а за ним вибираються всі об’єкти моделі Tenantreview, тобто відгуки про орендаря, які були створені відносно отриманого таким чином об’єкта орендаря. Якщо параметр tenant

заданий, то робиться все те саме, але для заданого орендаря, який визначається власним логіном.

Перегляд відгуків про ро музичний інструмент реалізований функцією musinstrument_reviews_view, яка має один основний параметр, який називається musicalinstrument. Це номер музичного інструменту, відгуки про який відбираються після отримання відповідного заданому номеру об’єкта.

Функція owner_page_view працює над переглядом сторінки власника музичних інструментів з використанням створеного шаблону сторінки owner.html, вона має один основний параметр, який називається owner і може бути не заданий. Якщо він не заданий, то передбачається, що функція підтримує роботу для авторизованого користувача, який є власником музичних інструментів, тобто це фактично його профільна сторінка. Тоді за акаунтом вибирається відповідний об’єкт моделі власника, а окрім нього ще три складові:

– перелік з 3 останніми відгуками про цього власника музичних інструментів reviews;

– перелік з 3 останніми доданими власником музичними інструментами instruments;

– кількість підтверджених замовлень прокату всіх музичних інструментів цього власника, яка була здійснена, booking_number.

Функція tenant_page_view працює над переглядом сторінки орендаря з використанням створеного шаблону сторінки tenant.html, вона має один основний параметр, який називається tenant і може бути не заданий. Якщо він не заданий, то передбачається, що функція підтримує роботу для авторизованого користувача, який є орендарем, тобто це фактично його профільна сторінка. Тоді за акаунтом вибирається відповідний об’єкт моделі орендаря, а окрім нього ще дві складові:

– перелік з 3 останніми відгуками про цього орендаря reviews;

– кількість підтверджених замовлень прокату музичних інструментів цим орендарем, яка була здійснена, booking_number.

Шаблони сторінок реалізовані на основі вільного шаблону розмітки [10].

Перегляд сторінки моделі музичного інструменту реалізований функцією instrumentmodel_page_view, яка має один основний параметр, що називається instrumentmodel і визначає номер моделі музичного інструмента. За витягнутим об’єктом цього музичного інструмента визначається також кількість підтверджених замовлень його прокату booking_number, що дозволяє судити в підсумку про популярність цього музичного інструменту.

Перегляд типів музичних інструментів за визначеним сімейством реалізований функцією instrumenttypes_view, параметр якої family визначає номер сімейства. Відповідно всі типи музичних інструментів, які відповідають об’єкту, який заданий цим номером, повертаються в результаті, впорядковані за алфавітом.

Перегляд моделей музичних інструментів за визначеним типом реалізований функцією instrumentmodels_view, параметр якої instrumenttype визначає номер типу. Відповідно всі моделі музичних інструментів, які відповідають об’єкту, який заданий цим номером, повертаються в результаті, впорядковані за алфавітом.

Перегляд виробників музичних інструментів реалізований функцією brands_view, яка не має параметрів і повертає тільки перелік з усіх виробників, яким заповнює шаблон відповідної сторінки brands.html.

Перегляд сімейств музичних інструментів реалізований функцією families_view, яка не має параметрів і повертає тільки перелік з усіх сімейств, яким заповнює шаблон відповідної сторінки families.html.

Перегляд моделей музичних інструментів за виробником реалізований функцією brandmodels_view, яка має основний параметр brand, який визначає номер виробника, а за ним витягується відповідний об’єкт моделі Instrumentbrand, а за ним відповідно всі об’єкти моделі Instrumentmodel, які відповідають цьому виробнику. Дані вносяться в шаблон сторінки brandmodels.html.

Функція propositions_view працює над переглядом створених власником цінових пропозицій з використанням створеного шаблону сторінки propositions.html. Параметрів функція не має, а власник визначається за обліковим записом. Відповідно, якщо це не власник музичних інструментів, а орендар або неавторизований користувач, то він не зможе отримати доступ до даних. Якщо об’єкт власника вдалось отримати, то відбираються всі об’єкти моделі Proposition, власником яких є авторизований власник.

Перегляд музичних інструментів зі списку бажання реалізується функцією instruments_list_view, яка не має параметрів, а орендар, для якого потрібно відобразити список бажань, визначається за обліковим записом. Якщо об’єкт орендаря за авторизованим акаунтом вдалось отримати, то відбираються всі об’єкти переліку з поля savedinstruments орендаря.

Пропонування музичних інструментів зі списку очікування, що з’явились, реалізується функцією waiting_instruments_list_view, яка не має параметрів, а орендар, для якого потрібно відобразити пропозиції зі списку очікування, визначається за обліковим записом. Якщо об’єкт орендаря за авторизованим акаунтом вдалось отримати, то відбираються всі об’єкти переліку з поля waitinginstruments орендаря. При цьому кожен такий об’єкт, тобто кожен музичний інструмент, розширюється полем saved. Воно визначає, чи внесений цей музичний інструмент авторизованим орендарем у список бажань. Якщо так, то міститься значення True, якщо ні – False. Окрім того до результуючого переліку вносяться тільки ті музичні інструменти, за якими немає підтверджених бронювань на дати після поточного моменту, а дедлайн або не встановлений, або встановлений пізніше поточної дати. Тобто це всі ті музичні інструменти, які в поточний момент орендар може замовити одразу.

Безпосередньо стандартний перегляд переліку очікування реалізовано функцією waitinglist_instruments_list_view, яка також для авторизованого орендаря виконує дії, але вибирає всі музичні інструменти, які він зберіг у

свій список очікування. При цьому інформація про те, чи можна цей інструмент замовити прямо зараз, вміщується в поле free за кожним інструментом. Істинне значення вказує, що цей інструмент одразу можна взяти в прокат, хибне, що ні.

Функція locate_rentalbooking_view працює над замовленням прокату музичного інструменту з використанням створеного шаблону сторінки rentalbooking.html. В основі використовується форма RentalbookingForm. Авторизований орендар визначається орендарем, який створив це замовлення, а параметром функції musicalinstrument визначається номер музичного інструменту прокат якого має бути забронювано.

Перегляд замовлених користувачем прокатів музичних інструментів реалізується функцією tenant_rentalbookings_view, яка за авторизованим орендарем за порядком, починаючи з найновішого, заповнює шаблон вебсторінки myrentalbookings.html даними замовлень у всіх станах, які були створені цим орендарем.

Перегляд замовлень на музичні інструменти власника реалізується функцією owner_rentalbookings_view, яка за авторизованим власником музичних інструментів за порядком, починаючи з найновішого, заповнює шаблон вебсторінки ownerrentalbookings.html даними замовлень у всіх станах, які були створені за музичними інструментами, власниками яких є авторизований користувач.

Перегляд замовлень за музичним інструментом реалізується функцією instrument_rentalbookings_view, яка за значенням параметра musinstrument, який є номером музичного інструменту, виводить перелік всіх замовлень, які стосуються його, перед цим перевіряючи, що авторизований на даний момент власник музичних інструментів є власником цього музичного інструменту.

Функція locate_rentalpayment_view працює над продовженням прокату музичного інструменту з використанням створеного шаблону сторінки rentalpayment.html зокрема. В основі використовується форма

RentalpaymentForm. Авторизований орендар перевіряється на те, що він створив замовлення, якого стосується цей платіж. Номер замовлення передається через параметр rentalbooking у функцію.

Функція approve_rentalbooking_view працює над затвердженням прокату музичного інструменту. Номер прокату передається через параметр rentalbooking. За цим замовленням перевіряється, що в ньому вказаний музичний інструмент, власником якого є авторизований користувач. Тільки в такому випадку змінюється стан замовлення в БД.

Функція cancel_rentalbooking_view працює над відхиленням прокату музичного інструменту. Номер прокату передається через параметр rentalbooking. За цим замовленням перевіряється, що в ньому вказаний музичний інструмент, власником якого є авторизований користувач. Тільки в такому випадку змінюється стан замовлення в БД.

Пошук моделей музичних інструментів реалізовано функцією search_modelinstruments_view, яка має параметри: type – визначає тип музичного інструменту, brand – виробника, family – сімейство, properties – словник значень властивостей, де за ключем, яким є назва властивості, зберігається значення цієї властивості. Будь-який параметр може бути не заданий. За встановленими параметрами відбираються моделі музичних інструментів, які їм відповідають, та повертаються в результаті в сторінці на основі шаблону search_models.html.

Пошук музичних інструментів для прокату реалізовано функцією search_instruments_view, яка має параметри: type – визначає тип музичного інструменту, brand – виробника, family – сімейство, properties – словник значень властивостей, де за ключем, яким є назва властивості, зберігається значення цієї властивості, при цьому це може бути значення властивості моделі інструмента, а може бути значення властивості самого музичного інструмента, city – номер міста, в якому має знаходитися музичний інструмент, country – країна, де знаходиться музичний інструмент, condition_min – найгірший стан музичного інструмента, який може бути

відібраний, condition_max – найкращий стан музичного інструмента, який може бути відібраний, buyoption – можливість придбання інструмента одразу (для обов’язкової наявності цієї можливості має бути встановлено значення True), leasingoption – можливість оренди з подальшим придбанням (для обов’язкової наявності цієї можливості має бути встановлено значення True), price_min – мінімальна вартість прокату, price_max – максимальна вартість прокату, start – дата початку прокату, end – дата завершення прокату, level – рівень вмінь, якому має відповідати інструмент (використовується тоді, коли інструмент шукається для навчання), no_deposit – відсутність необхідності внесення депозиту (для обов’язкової відсутності потрібно вказати значення True), review_min – мінімальна середня оцінка, review_max – максимальна середня оцінка. За цими параметрами відбувається відбір, залишаючи тільки ті музичні інструменти в результаті, які відповідають цим параметрам. Окремо перевіряється доступність для замовлення музичних інструментів та вартості замовлення. Для цього за кожним музичним інструментом, який пройшов відбір за всіма іншими параметрами, реалізується перевірка того, чи не має стосовно нього підтвердженого замовлення, яке за датами перетинається зі вказаними датами замовлення. Якщо такого замовлення немає, то тоді перевіряється, чи є замовлення, яке розпочалося і має закінчитися до вказаних строків і якщо є, то чи не визначено воно як замовлення з придбанням у кінці. Якщо це саме такий музичний інструмент, для якого таке замовлення є, то в полі inleasing для нього вказується істинне значення, якщо немає, то вказується хибне значення. Після цього відбувається встановлення вартості замовлення на вказані дати. Для цього за кожним тарифом денним, тижневим, місячним та річним перевіряється, скільки в такому випадку може коштувати замовлення. Найменша вартість перевіряється на входження в інтервал, заданий в пошуку. Якщо вартість входить в цей інтервал, то музичний інструмент залишається в результатах, якими наповнюється шаблон сторінки search_models.html.

Перегляд повних даних музичного інструменту реалізовано функцією musicalinstrument_view, яка окрім параметра musinstrument, що визначає номер музичного інструменту, має також параметри start і end, що визначають час прокату музичного інструменту. Тобто на цій сторінці можна не тільки переглянути дані інструменту, але і перевірити вартість і можливість замовлення цього інструменту на потрібні дати. Визначення можливості замовлення прокату і встановлення вартості такого прокату відбувається аналогічно тому, як це було для пошуку. Так само як і для пошуку перевіряється, чи збережений у список бажань цей музичний інструмент, а результатом є ключ saved. Ключ waited визначає, чи доданий інструмент до списку очікування авторизованого орендаря. Ключ waiteable визначає, чи може ще з’явитися потенційно можливість замовлення прокату цього інструменту, якщо він зараз у прокаті знаходиться. Логіка перевірки така сама, як описувалась вище. Також за ключем reviews наповнюється шаблон сторінки musicalinstrument.html 3 останніми відгуками про цей музичний інструмент.

Перегляд сімейств музичних інструментів реалізовано функцією families_view.

Внесення музичного інструменту в список бажання авторизованого орендаря та вилучення відбуваються у відповідності з номером інструменту, який задається параметром musinstrument функції save_musinstrument_view. Якщо музичний інструмент у список внесений, то він вилучається, якщо ні, то додається.

Внесення музичного інструменту в список очікування авторизованого орендаря та скасування цього відбуваються у відповідності з номером інструменту, який задається параметром musinstrument функції wait_musinstrument_view. Якщо музичний інструмент у список внесений, то він вилучається, якщо ні, то додається. Внесення реалізується тільки тоді, якщо цей музичний інструмент може звільнитися до часу його дедлайну

(коли його має бути остаточно повернуто власнику) за даними поточних замовлень.

Функція locate_musicalinstrument_view працює над розміщенням прокату музичного інструменту з використанням створеного шаблону сторінки instrumentlocate.html. Розміщення можливе тільки тоді, якщо авторизований акаунт належить власнику музичних інструментів і він є активним. Тоді надається доступ до роботи з формою, яка створена через клас MusicalinstrumentForm, що визначає всі поля відповідної таблиці БД, значення яких має задати користувач. Аналогічно влаштовано створення цінової пропозиції про прокат музичних інструментів на основі функції locate_proposition_view. Редагування даних музичного інструменту реалізовано за допомогою функції edit_musinstrument_view, параметром якої є musinstrument, що визначає номер музичного інструменту, редагування даних якого має відбутися. Форма використовується та сама, що і під час розміщення. У ній можливе також визначення пропозицій за цим інструментом з пропозицій власника. Редагування цінової пропозиції про прокат музичних інструментів реалізовано через функцію edit_proposition_view.

Функція locate_instreview_view працює над створенням відгуку про музичний інструмент з використанням створеного шаблону сторінки instreview.html. Вона також підтримує в основі форму класу MusicalinstrumentForm у даному випадку. Але при створенні відгуку перевіряється, що орендарю, який авторизувався, ця можливість доступна. Для цього витягається останнє підтверджене замовлення прокату цього орендаря за цим музичним інструментом, номер якого вказується в параметрі musinstrument. Якщо воно було створено пізніше ніж останній опублікований відгук цього орендаря за цим музичним інструментом, то відбувається підтримка через описану форму. Аналогічно влаштовано функції locate_ownerreview_view для створення відгуку про власника музичного інструменту, locate_tenantreview_view для створення відгуку про орендаря.

### 3.3 Висновки за розділом 3

Заплановані функції програми реалізовано за допомогою класів моделей для роботи з даними, функцій для підтримки формування готових вебсторінок на основі файлів шаблонів з заповненням їх даними у відповідністю з кожною конкретною функцією.

# 4 ЕКСПЛУАТАЦІЯ ТА ТЕСТУВАННЯ ПРОГРАМИ

## 4.1 Призначення програми

Програма призначена для:

– забезпечення власників музичних інструментів можливістю надати їх у прокат в той час, коли вони не використовуються, визначаючи параметри цього прокату, включаючи вартість, характеристики музичного інструменту та можливості подальшого придбання музичних інструментів;

– забезпечення людей, які шукають можливість взяти музичний інструмент на прокат, такою можливістю без необхідності взаємодії зі службами прокату музичних інструментів.

## 4.2 Виконання і тестування програми

Перевірку програми виконано в браузерах Google Chrome версії 113, Firefox версії 113 та Microsoft Edge версії 113. Виконання функціональних вимог забезпечено повністю, сторінки коректно відображаються у всіх перелічених браузерах зазначених версій.

Робота з програмою реалізується спочатку через інтерфейс, який можна вважати стандартним, адже він використовується і для неавторизованого користувача, і для орендаря музичних інструментів після того, як він авторизувався або зареєструвався, якщо не має облікового запису. При спробі замовлення неавторизований користувач перенаправляється до авторизації. Якщо це авторизований орендар музичних інструментів, то такі функції працюють без перенаправлення.

Меню стандартного режиму надає доступ до основних функцій. Меню пошуку складається з пошуку за моделями музичних інструментів, щоб дізнатися про наявні моделі різних інструментів загалом на ринку, та пошуку музичних інструментів, які вже орендар може безпосередньо взяти на прокат.

Пошук передбачає встановлення параметрів, які було детально представлено в розділі 3. З пошуку можна перейти до певного оголошення про прокат (рис. 4.1), на якому демонструється виробник і назва моделі, зображення серця для збереження у список бажань (заповнене, коли цей інструмент вже внесено в список бажань, і порожнє в іншому випадку). Біля назви знаходиться фотографія музичного інструменту, при цьому можна переходити між всіма створеними за інструментом фотографіями за допомогою посилань вліво та вправо під фотографію. Кнопка Замовити дозволяє одразу переміститься на частину сторінки, де можна задати строк замовлення і перевірити вартість.

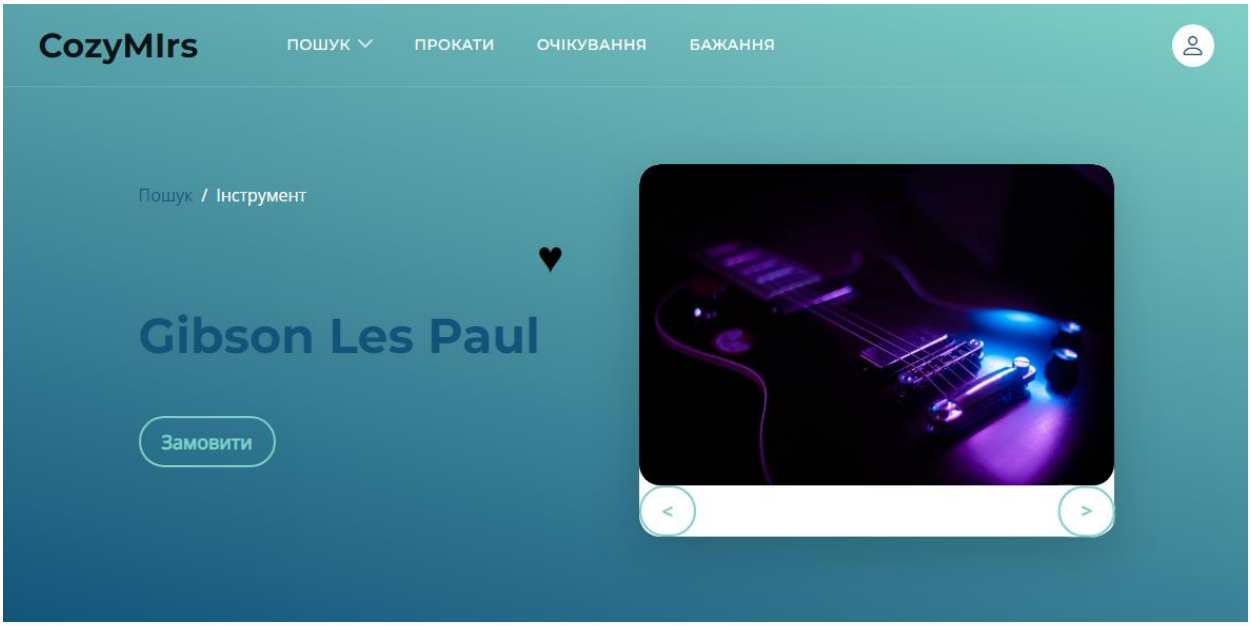


Рисунок 4.1 – Сторінка музичного інструменту

Нижче під цим рядком знаходиться інформація про тип музичного інструменту та опис, який створив власник. Нижче всі відповідні параметри, які визначаються за інструментом (розглянуті далі при розміщенні оголошення). Під цими даними знаходиться перелік властивостей інструмента та моделі інструмента зі значеннями (наприклад, матеріали грифу).

Після цього знаходиться форм перевірки доступності інструмента (рис. 4.2), де потрібно задати початок і кінець прокату, натиснути на кнопку Перевірити.

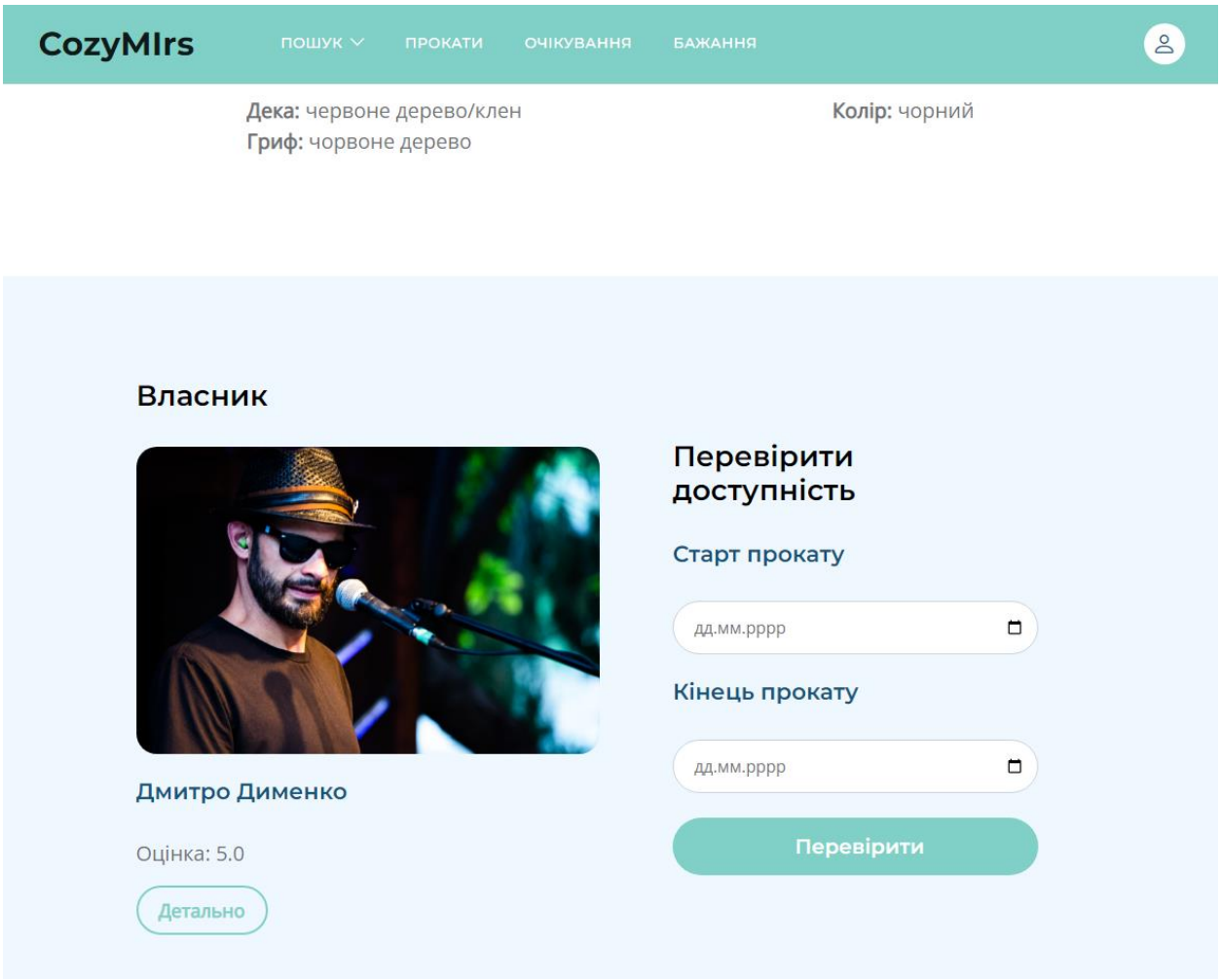


Рисунок 4.2 – Перевірка доступності музичного інструменту

Результатом перевірки може бути або повідомлення про вартість прокату та кнопка Замовити для переходу до замовлення прокату (рис. 4.3),

або повідомлення про те, що інструмент не доступний на ці дати, та кнопка для внесення інструменту в список очікувань.

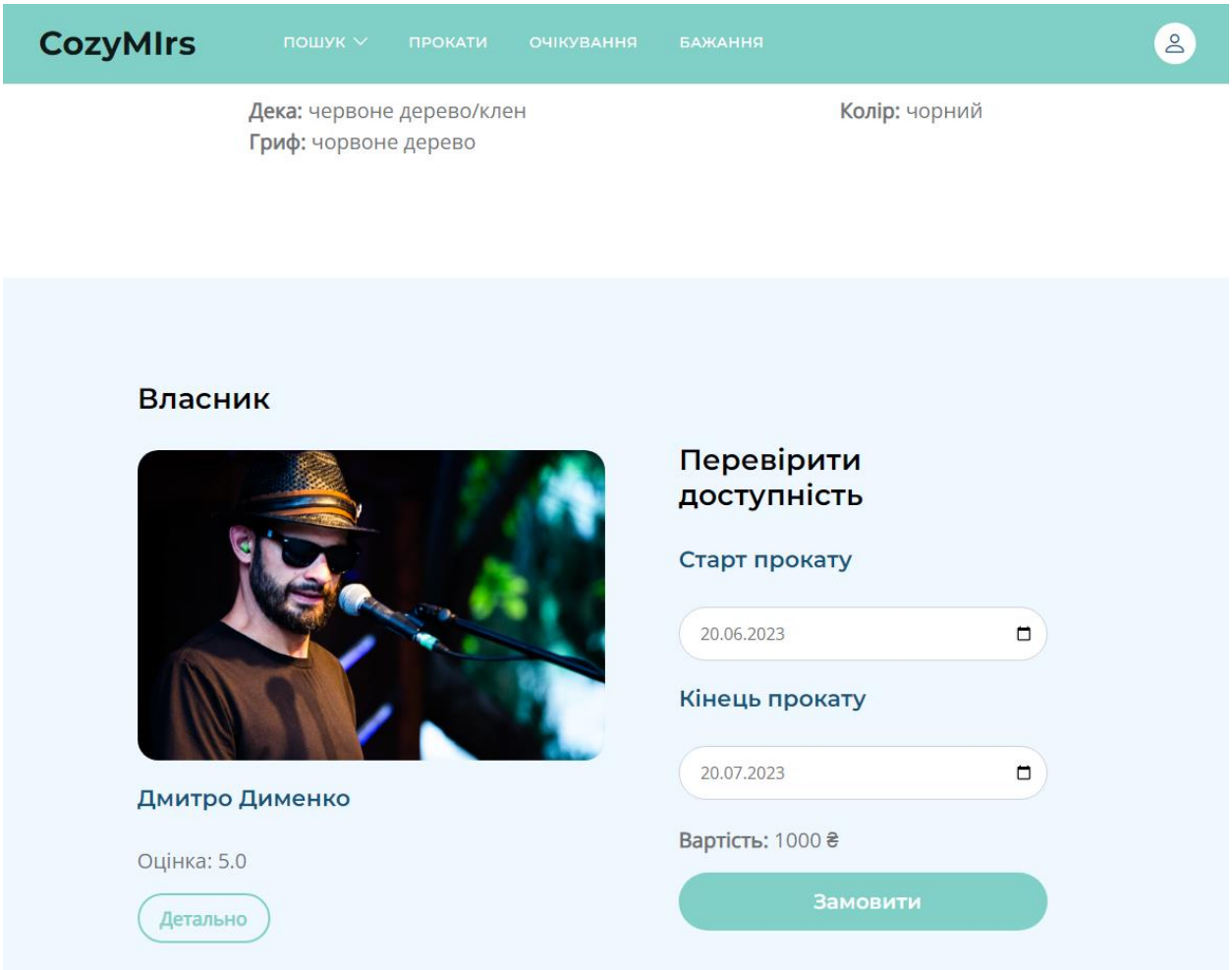


Рисунок 4.3 – Успішна перевірка доступності музичного інструменту

Біля форми перевірки знаходиться інформація про власника музичного інструменту. Натискаючи на ім’я, можна перейти на сторінку з даними про орендаря. Нижче на цій сторінці також знаходяться 3 останні відгуки про музичний інструмент та кнопка для переходу до сторінки зі всіма відгуками.

Меню Прокати необхідно, щоб авторизований орендар зміг переглянути всі замовлення, які він вже створив, щоб дізнатися про їх стан. Там же, якщо замовлення завершено, знаходиться кнопка для створення відгуку.

Меню Очікування відображає всі додані до очікування музичні інструменти (рис. 4.4). Відповідна сторінка містить 2 розділи: один для відображення всіх музичних інструментів, доданих до очікування, та другий – для відображення тільки вже вільних для замовлення в даний момент інструментів. У першому випадку біля кожного інструмента, який з’явився, відображається кнопка Забронювати для переходу до бронювання.

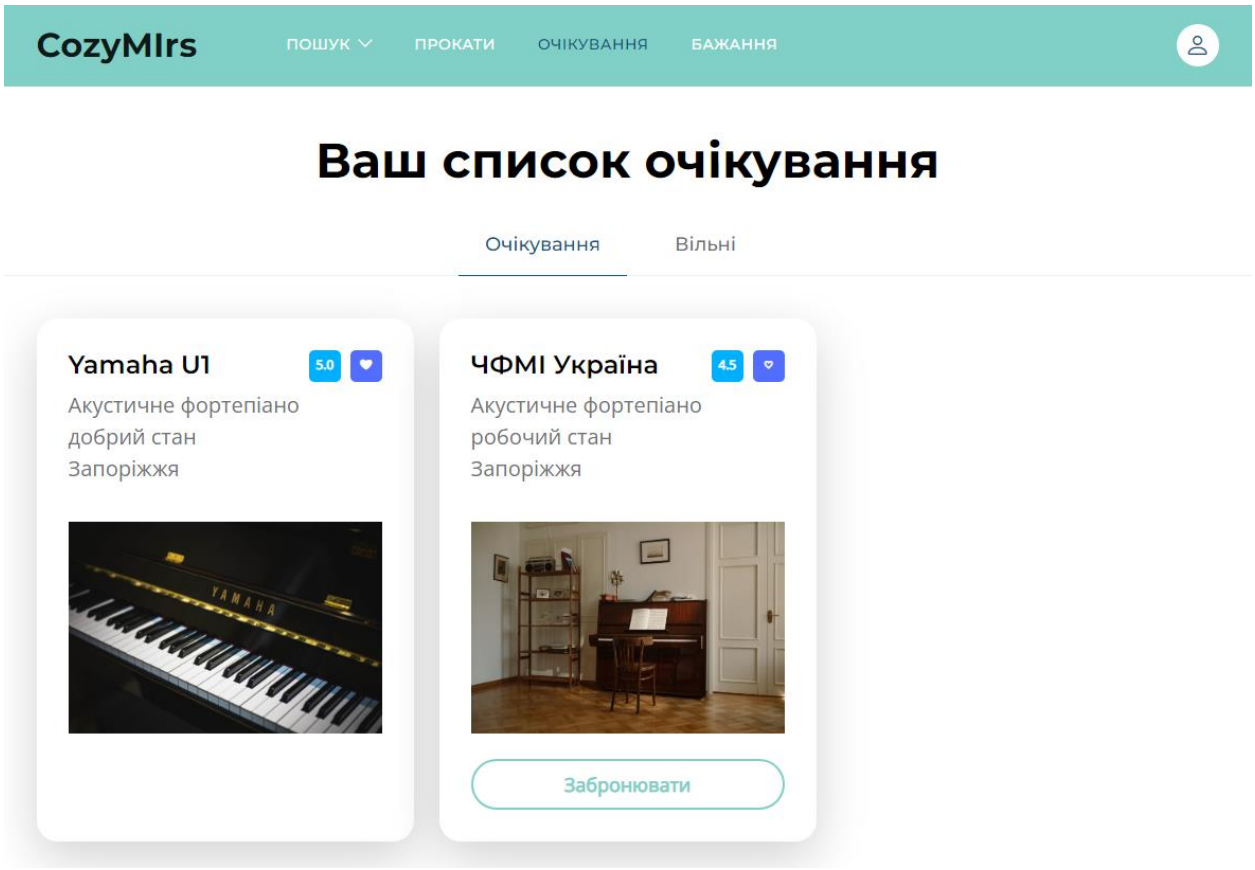


Рисунок 4.4 – Список очікування

Біля кожного інструмента відображається так само, як було описано, зображення серця для внесення інструмента до списку бажань або визначення, що його вже туди внесено. Лівіше від нього демонструється оцінка.

Меню Бажання дозволяє переглянути як раз сформований таким чином список бажань орендаря.

Власник музичних інструментів після введення власних даних на формі авторизації одразу перенаправляється до другого інтерфейсу, який включає дещо змінене меню. Зокрема меню містить Замовлення, що дозволяє переглянути всі замовлення, адресовані за музичними інструментами цього власника, Прокати, що дозволяє переглянути замовлення, за якими інструменти знаходяться у орендарів, Інструменти, де виводяться всі музичні інструменти, якими володіє орендар. Там де доступна можливість редагування даних, а також переходу до розміщення нового музичного інструменту (рис. 4.5).

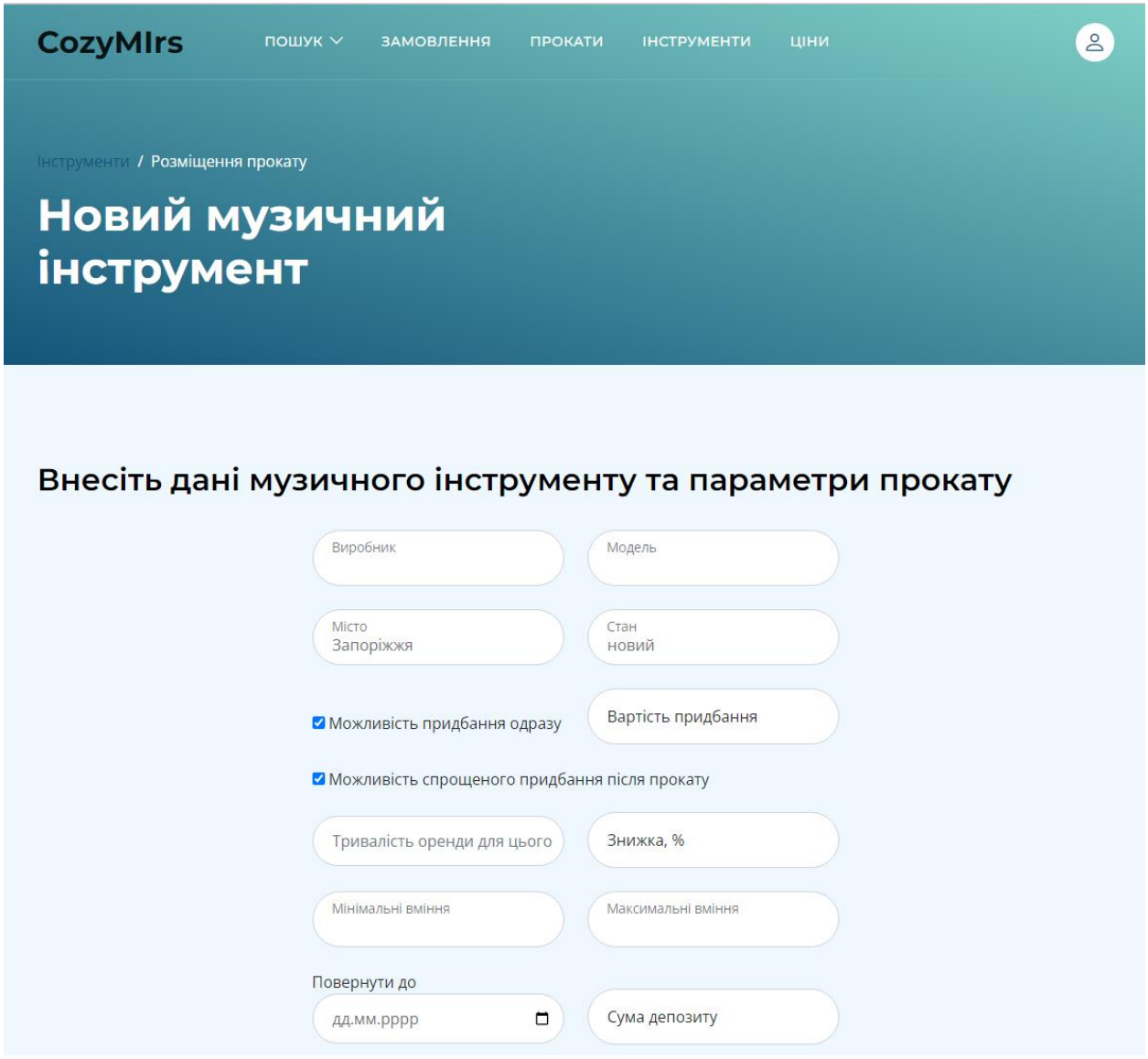


Рисунок 4.5 – Розміщення прокату нового музичного інструменту

При розміщенні прокату потрібно задати різні параметри, частина з них (наприклад, вартість придбання) блокується при виборі певних параметрів (якщо можливість придбання деактивована в цьому випадку). Для додавання фотографії потрібно кнопкою Завантажити визначити файл, який буде завантажений на сервер (рис. 4.6). Для додавання аксесуарів до цього інструмента з усіх інструментів власника потрібно ввести назву, що дозволить з випадаючого переліку вибрати потрібний аксесуар, та натиснути на кнопку Додати. У кінці потрібно зберегти дані, завершивши розміщення.

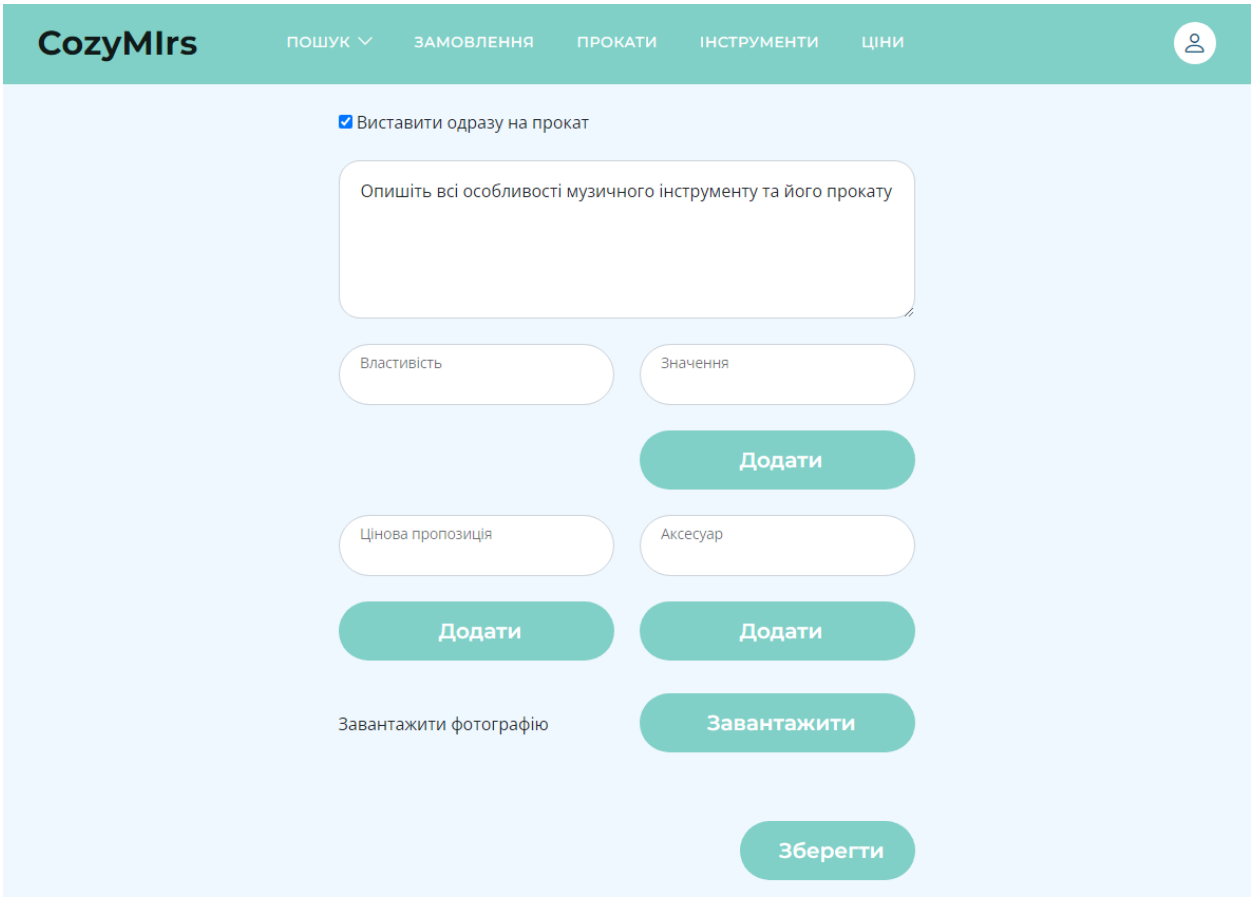


Рисунок 4.6 – Визначення другої частини параметрів під час розміщення прокату нового музичного інструменту

### 4.3 Висновки за розділом 4

Усі функції забезпечено. Створений інтерфейс організований за допомогою меню, що відрізняється для орендарів і власників.

# ВИСНОВКИ

У роботі необхідно було забезпечити підтримку взаємодії між власниками музичних інструментів та орендарями, для чого створити ПЗ. Аналіз предметної області показав, що основна маса таких ПЗ підтримує роботу окремої служби прокату музичних інструментів, а тільки FriendWithA реалізує ідею підтримки прокату від власників. Однак наявна програма не вирішує проблем українських музикантів. Сприяючи більш широкому охопленню аудиторію користувачів, розробка виконувалась у вигляді вебдодатку.

Розроблена в підсумку програма призначена для забезпечення власників музичних інструментів можливістю надати їх у прокат в той час, коли вони не використовуються, визначаючи параметри цього прокату, включаючи вартість, характеристики музичного інструменту і можливості подальшого придбання музичних інструментів та для забезпечення людей, які шукають можливість взяти музичний інструмент на прокат, такою можливістю.

Програма реалізована мовою Python. Створені всі програмні функції для підтримки визначених функціональних вимог, розроблені відповідні вебсторінки для їх підтримки.

Використання програми може бути корисним як під час навчання грі на музичних інструментах, коли придбання нового інструменту може бути не дуже доцільним, так і під час професійної кар’єри, коли потрібна можливість гнучкого змінювання інструментів для виконання різних партій і такому підходу віддається перевага над персональним володінням одного інструменту або ці підходи використовуються разом. Реалізація програми як вебдодатку сприяє більш широкій підтримці користувачів.

# ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Rent My Instrument – Online Music Rentals Simplified [Electronic resource]. – Access mode : https://www.rentmyinstrument.com/.

2. Rent Musical Instruments Online [Electronic resource]. – Access mode : https://www.grover.com/de-en/audio-and-music/musical-instruments.

3. Musical Instrument Hire Co. We make it easy, hire today! [Electronic resource]. – Access mode : https://musicalinstrumenthire.com/

4. FriendWithA [Electronic resource] : The Online Marketplace for Gear Rentals. – Access mode : https://friendwitha.com/.

5. Downey, A. Think Python [Text] : How to Think Like a Computer Scientist / A. Downey. – Cambridge : O'Reilly Media, 2016. – 289 p.

6. Ernesti, J. Python 3 [Text] : The Comprehensive Guide to Hands-On Python Programming / Johannes Ernesti, Peter Kaiser. – Bonn : Rheinwerk Computing, 2022. – 1078 p.

7. Lukaszewski, A. MySQL for Python [Text] / A. Lukaszewski. – Birmingham : Packt Publishing, 2010. – 440 p.

8. Grippa, V. Learning MySQL [Text] : Get a Handle on Your Data / V. Grippa, S. Kuzmichev. – Cambridge : O'Reilly Media, 2021. – 629 p.

9. Krogh, J. W. MySQL Connector/Python Revealed [Text] : SQL and NoSQL Data Storage Using MySQL for Python Programmers. – New York City : Apress, 2018. – 538 p.

10. Topic Listing Template [Electronic resource]. – Access mode : https://templatemo.com/tm-590-topic-listing.

# ДОДАТОК А

## Технічне завдання

## Вступ

Сервіси прокату є достатньо популярними у всьому світі. На прокат беруть як преміальні речі, так і достатньо буденні. Самі по собі музичні інструменти важко назвати буденними речами, адже зазвичай інструменти коштують достатньо дорого, відповідно далеко не кожен може дозволити собі придбати новий музичний інструмент, а щонайменше навчатися хочуть багато людей. Прокат музичних інструментів може бути вирішенням цієї проблеми. Однак як в багатьох галузях, прокат від спеціалізованих служб є проблемою, адже такі компанії зазвичай встановлюють певні інтервали цін, а знизити їх орендар ніяк не може. Прокат інструментів від власників може вирішити цю проблему. Однак для контактування з власниками потрібно отримати необхідну інформацію про інструменти, мати можливість дізнатися з відгуків про те, яким був досвід людей, які вже спробували ці можливості. Тобто без відповідного ПЗ не обійтися при вирішенні цієї проблеми.

### А.1 Підстави для розробки

Завдання на дипломну кваліфікаційну роботу визначено на підставі наказу № 87 від 4 квітня 2023 р. за Національним університетом «Запорізька політехніка» за темою «Програмне забезпечення прокату музичних інструментів».

### А.2 Призначення розробки

Програма призначена для:

– забезпечення власників музичних інструментів можливістю надати їх у прокат в той час, коли вони не використовуються, визначаючи параметри цього прокату, включаючи вартість, характеристики музичного інструменту та можливості подальшого придбання музичних інструментів;

– забезпечення людей, які шукають можливість взяти музичний інструмент на прокат, такою можливістю без необхідності взаємодії зі службами прокату музичних інструментів.

### А.3 Основні вимоги до програми

#### А.3.1 Вимоги до функціональних характеристик

Для підтримки описаного призначення програма повинна надавати користувачам наступний набір функцій для роботи:

– перегляд сторінки моделі музичного інструменту;

– пошук моделей музичних інструментів;

– пошук музичних інструментів для прокату;

– перегляд повних даних музичного інструменту;

– розміщення прокату музичного інструменту;

– перегляд музичних інструментів у прокаті власника;

– замовлення прокату музичного інструменту;

– внесення музичного інструменту в список бажання;

– перегляд музичних інструментів зі списку бажання;

– вилучення музичного інструменту зі списку бажання;

– внесення музичного інструменту в список очікування;

– скасування очікування музичного інструменту;

– пропонування музичних інструментів зі списку очікування, що з’явились;

– перегляд оформлених замовлених користувачем прокатів музичних інструментів;

– перегляд замовлень за музичним інструментом;

– перегляд замовлень на музичні інструменти власника;

– продовження прокату музичного інструменту;

– створення відгуку про музичний інструмент;

– створення відгуку про власника музичного інструменту;

– створення відгуку про орендаря;

– перегляд сторінки власника музичних інструментів;

– перегляд відгуків про власника музичних інструментів;

– перегляд відгуків про музичний інструмент;

– перегляд відгуків про орендаря;

– перегляд сторінки орендаря;

– створення цінової пропозиції про прокат музичних інструментів;

– редагування цінової пропозиції про прокат музичних інструментів;

– визначення можливих цінових пропозицій за музичним інструментом;

– перегляд створених власником цінових пропозицій;

– редагування даних музичного інструменту;

– затвердження або відхилення прокату музичного інструменту;

– перегляд сімейств музичних інструментів;

– перегляд типів музичних інструментів за визначеним сімейством;

– перегляд моделей музичних інструментів за визначеним типом;

– перегляд виробників музичних інструментів;

– перегляд моделей музичних інструментів за виробником;

– перегляд сімейств музичних інструментів;

– перегляд переліку очікування;

– авторизація і реєстрація як орендарів, так і власників музичних інструментів.

### А.3.2 Вимоги до складу та параметрів технічних засобів

Апаратні вимоги до серверної сторони:

– процесор у мінімальній конфігурації Intel Core 2 або еквівалент;

– оперативна пам’ять з доступом не менше ніж до 4 Гб;

– жорсткий диск з бажаним доступним обсягом не менше ніж 100 Гб, а у випадку меншого обсягу контролювати обсяг даних у БД при внесенні

фотографій музичних інструментів, при цьому в будь-якому випадку потрібний обсяг може і значно збільшуватися, що залежить від кількості користувачів, кількості та якості фотографій, які вони завантажують.

### А.4 Порядок контролю та приймання

Основою щодо реалізації контролю та приймання дипломної кваліфікаційної роботи є завдання на роботу і безпосередньо дане технічне завдання. За успішного виконання цих завдань у встановлені строки робота після проходження всіх підготовчих процедур виноситься на захист перед екзаменаційною комісією.

# ДОДАТОК Б

## Текст програми

```
from .models import Owner, Musicalinstrument, Ownerreview,
Tenantreview, Tenant, Instrumentreview, Rentalbooking, Instrumentmodel,
Instrumentfamily, Instrumenttype, Accessory, Instrumentbrand, City, Property,
Propertyvalue, Proposition,
from django.shortcuts import redirect, render
import datetime
import logging
from django.db.models import Q
import math
from .forms import MusicalinstrumentForm, InstrumentreviewForm,
OwnerreviewForm, TenantreviewForm, PropositionForm

error_log = logging.getLogger(__name__)

def owner_instruments_view(request, owner = None):
    if owner:
        try:
            owner = Owner.objects.get(user__username = owner)
        except Owner.DoesNotExist:
            error_log.error("Власника з логіном "+owner+" не знайдено.")
            owner = None
    if not owner:
        try:
            owner = Owner.objects.get(user = request.user)
        except Owner.DoesNotExist:
            error_log.error("Авторизованого власника з логіном "+
request.user.username+" не знайдено.")
            owner = None
    musinstruments = []
    if owner:
```

```
            musinstruments = Musicalinstrument.objects.filter(owner =
owner).select_related("instrumentmodel").select_related("instrumentmodel__instru
menttype").select_related("instrumentmodel__instrumentbrand").select_related("in
strumentmodel__instrumenttype").select_related("instrumentmodel__instrumentty
pe__instrumentfamily").order_by("-id")
            try:
                tenant = Tenant.objects.get(user =
request.user).prefetch_related("savedinstruments")
            except Tenant.DoesNotExist:
                tenant = None
            if tenant:
                for musinstrument in musinstruments:
                    saved = False
                    if musinstrument in tenant.savedinstruments:
                        saved = True
                    musinstrument.saved = saved
        return render(request, "cozymirs/instruments_owner.html",
{"instruments": musinstruments })


    def owner_reviews_view(request, owner = None):
        if owner:
            try:
                owner = Owner.objects.get(user__username = owner)
            except Owner.DoesNotExist:
                error_log.error("Власника з логіном "+owner+" не знайдено.")
                owner = None
        if not owner:
            try:
                owner = Owner.objects.get(user = request.user)
            except Owner.DoesNotExist:
```

```
                error_log.error("Авторизованого власника з логіном "+
request.user.username+" не знайдено.")
                owner = None
        ownerreviews = []
        if owner:
            ownerreviews = Ownerreview.objects.filter(owner =
owner).select_related("tenant").order_by("-date")
        return render(request, "cozymirs/owner_reviews.html", {"reviews":
ownerreviews })

    def tenant_reviews_view(request, tenant = None):
        if tenant:
            try:
                tenant = Tenant.objects.get(user__username = tenant)
            except Tenant.DoesNotExist:
                error_log.error("Орендаря з логіном "+tenant+" не знайдено.")
                tenant = None
        if not tenant:
            try:
                tenant = Tenant.objects.get(user = request.user)
            except Tenant.DoesNotExist:
                error_log.error("Авторизованого орендаря з логіном "+
request.user.username+" не знайдено.")
                tenant = None
        tenantreviews = []
        if tenant:
            tenantreviews = Tenantreview.objects.filter(tenant =
tenant).select_related("owner").order_by("-date")
        return render(request, "cozymirs/tenant_reviews.html", {"reviews":
tenantreviews })
```

```
        def musinstrument_reviews_view(request, musicalinstrument):
            try:
                musicalinstrument = Musicalinstrument.objects.get(id =
musicalinstrument)
            except Musicalinstrument.DoesNotExist:
                error_log.error("Музичного інструменту з номером
"+musicalinstrument+" не знайдено.")
                musicalinstrument = None
            instrumentreviews = []
            if musicalinstrument:
                instrumentreviews =
Instrumentreview.objects.filter(musicalinstrument =
musicalinstrument).select_related("tenant").order_by("-date")
            return render(request, "cozymirs/instrument_reviews.html", {"reviews":
instrumentreviews })


        def owner_page_view(request, owner = None):
            if owner:
                try:
                    owner = Owner.objects.get(user__username = owner)
                except Owner.DoesNotExist:
                    error_log.error("Власника з логіном "+owner+" не знайдено.")
                    owner = None
            if not owner:
                try:
                    owner = Owner.objects.get(user = request.user)
                except Owner.DoesNotExist:
                    error_log.error("Авторизованого власника з логіном "+
request.user.username+" не знайдено.")
                    owner = None
```

```
            ownerreviews = []
            musicalinstruments = []
            bookings = None
            if owner:
                ownerreviews = Ownerreview.objects.filter(owner =
owner).select_related("tenant").order_by("-date")[:3]
                bookings =
len(Rentalbooking.objects.filter(musicalinstrument__owner = owner, state__gte =
2).values())
                musicalinstruments = Musicalinstrument.objects.filter(owner =
owner).select_related("instrumentmodel").select_related("instrumentmodel__instru
mentbrand").select_related("instrumentmodel__instrumenttype").select_related("in
strumentmodel__instrumenttype__instrumentfamily").select_related("city").order_
by("-id")[:3]
                try:
                    tenant = Tenant.objects.get(user =
request.user).prefetch_related("savedinstruments")
                except Tenant.DoesNotExist:
                    tenant = None
                if tenant:
                    for musicalinstrument in musicalinstruments:
                        saved = False
                        if musicalinstrument in tenant.savedinstruments:
                            saved = True
                        musicalinstrument.saved = saved
            return render(request, "cozymirs/owner.html", {"owner": owner,
"reviews": ownerreviews, "instruments": musicalinstruments, "booking_number":
bookings })


        def tenant_page_view(request, tenant = None):
```

```
            if tenant:
                try:
                    tenant = Tenant.objects.get(user__username = tenant)
                except Tenant.DoesNotExist:
                    error_log.error("Орендаря з логіном "+tenant+" не знайдено.")
                    tenant = None
            if not tenant:
                try:
                    tenant = Tenant.objects.get(user = request.user)
                except Tenant.DoesNotExist:
                    error_log.error("Авторизованого орендаря з логіном "+
request.user.username+" не знайдено.")
                    tenant = None
            tenantreviews = []
            bookings = None
            if owner:
                tenantreviews = Ownerreview.objects.filter(tenant =
tenant).select_related("owner").order_by("-date")[:3]
                bookings = len(Rentalbooking.objects.filter(tenant = tenant, state__gte
= 2).values())
            return render(request, "cozymirs/tenant.html", {"tenant": tenant,
"reviews": tenantreviews, "booking_number": bookings })


        def instrumentmodel_page_view(request, instrumentmodel):
            bookings = None
            if instrumentmodel:
                try:
                    instrumentmodel = Instrumentmodel.objects.get(id =
instrumentmodel).select_related("instrumentbrand").select_related("instrumenttype
").select_related("instrumenttype__instrumentfamily")
```

```
            except Instrumentmodel.DoesNotExist:
                error_log.error("Моделі музичного інструменту з номером "+instrumentmodel+" не знайдено.")
                instrumentmodel = None
        if instrumentmodel:
            bookings = len(Rentalbooking.objects.filter(musicalinstrument__instrumentmodel = instrumentmodel, state__gte = 2).values())
        return render(request, "cozymirs/owner_reviews.html", {"instrumentmodel": instrumentmodel, "booking_number": bookings })

    def instrumenttypes_view(request, family):
        instrumenttypes = []
        try:
            instrumentfamily = Instrumentfamily.objects.get(id = family)
        except Instrumentfamily.DoesNotExist:
            error_log.error("Сімейства музичних інструментів з номером "+ family +" не знайдено.")
            instrumentfamily = None
        if instrumentfamily:
            instrumenttypes = Instrumenttype.objects.filter(instrumentfamily = instrumentfamily).order_by("typename")
        return render(request, "cozymirs/instruments.html", {"instrumenttypes": instrumenttypes })

    def instrumentmodels_view(request, instrumenttype):
        instrumentmodels = []
        try:
            instrumenttype = Instrumenttype.objects.get(id = instrumenttype)
        except Instrumenttype.DoesNotExist:
```

```
            error_log.error("Типу музичних інструментів з номером "+ family
+" не знайдено.")
            instrumenttype = None
        if instrumenttype:
            instrumentmodels = Instrumentmodel.objects.filter(instrumenttype =
instrumenttype).order_by("model")
        return render(request, "cozymirs/models.html", {"instrumentmodels":
instrumentmodels })

    def brands_view(request):
        brands = Instrumentbrand.objects.all()
        return render(request, "cozymirs/brands.html", {"brands": brands })

    def families_view(request):
        families = Instrumentfamily.objects.all()
        return render(request, "cozymirs/families.html", {"families": families })

    def brandmodels_view(request, brand):
        brandmodels = []
        try:
            brand = Instrumentbrand.objects.get(id = brand)
        except Instrumentbrand.DoesNotExist:
            error_log.error("Виробника музичних інструментів з номером "+
brand +" не знайдено.")
            brand = None
        if brand:
            brandmodels = Instrumentmodel.objects.filter(instrumentbrand =
brand).order_by("model")
        return render(request, "cozymirs/brandmodels.html", {"brandmodels":
brandmodels })
```

```
        def propositions_view(request):
            try:
                owner = Owner.objects.get(user = request.user)
            except Owner.DoesNotExist:
                error_log.error("Авторизованого власника з логіном "+
request.user.username+" не знайдено.")
                owner = None
            propositions = []
            if owner:
                propositions = Proposition.objects.filter(owner = owner).order_by("-
id")
            return render(request, "cozymirs/propositions.html", {"propositions":
propositions })

        def instruments_list_view(request):
            try:
                tenant = Tenant.objects.get(user =
request.user).prefetch_related("savedinstruments")
            except Tenant.DoesNotExist:
                error_log.error("Авторизованого орендаря з логіном "+
request.user.username+" не знайдено.")
                tenant = None
            instruments = []
            if tenant:
                instruments = tenant.savedinstruments
            return render(request, "cozymirs/myinstruments.html", {"instruments":
instruments })

        def waiting_instruments_list_view(request):
            try:
```

```
            tenant = Tenant.objects.get(user =
request.user).prefetch_related("waitinginstruments").prefetch_related("savedinstru
ments")
        except Tenant.DoesNotExist:
            error_log.error("Авторизованого орендаря з логіном "+
request.user.username+" не знайдено.")
            tenant = None
        instruments = []
        if tenant:
            for musinstrument in tenant.waitinginstruments:
                saved = False
                if musinstrument in tenant.savedinstruments:
                    saved = True
                musinstrument.saved = saved
                current_time = datetime.datetime.now()
                bookings = Rentalbooking.objects.filter(end__gt = current_time,
state__gte = 2)
                if musinstrument.actual == True and (not musinstrument.deadline
or musinstrument.deadline > current_time) and not bookings:
                    instruments.append(musinstrument)
        return render(request, "cozymirs/myinstruments.html", {"instruments":
instruments })


    def waitinglist_instruments_list_view(request):
        try:
            tenant = Tenant.objects.get(user =
request.user).prefetch_related("waitinginstruments").prefetch_related("savedinstru
ments")
        except Tenant.DoesNotExist:
```

```
                error_log.error("Авторизованого орендаря з логіном "+
request.user.username+" не знайдено.")
                tenant = None
            instruments = []
            if tenant:
                for musinstrument in tenant.waitinginstruments:
                    saved = False
                    if musinstrument in tenant.savedinstruments:
                        saved = True
                    musinstrument.saved = saved
                    current_time = datetime.datetime.now()
                    bookings = Rentalbooking.objects.filter(end__gt = current_time,
state__gte = 2)
                    if musinstrument.actual == True and musinstrument.deadline >
current_time and not bookings:
                        musinstrument.free = True
                    else:
                        musinstrument.free = False
            return render(request, "cozymirs/waitinginstruments.html",
{"instruments": instruments })


        def search_modelinstruments_view(request, type = None, brand = None,
family = None, properties = None):
            modelinstruments = Instrumentmodel.objects.all()
            if type:
                try:
                    instrumenttype = Instrumenttype.objects.get(id = type)
                    modelinstruments = modelinstruments.filter(instrumenttype =
instrumenttype)
                except Instrumenttype.DoesNotExist:
```

```
                error_log.error("Типу музичного інструменту з номером "+ type
+" не знайдено.")
            if brand:
                try:
                    instrumentbrand = Instrumentbrand.objects.get(id = brand)
                    modelinstruments = modelinstruments.filter(instrumentbrand =
instrumentbrand)
                except Instrumentbrand.DoesNotExist:
                    error_log.error("Виробника музичних інструментів з номером "+
brand +" не знайдено.")
            if family:
                try:
                    instrumentfamily = Instrumentfamily.objects.get(id = family)
                    modelinstruments =
modelinstruments.filter(instrumenttype__instrumentfamily = instrumentfamily)
                except Instrumentfamily.DoesNotExist:
                    error_log.error("Сімейства музичних інструментів з номером "+
family +" не знайдено.")
                for property_name in properties.keys():
                    try:
                        property = Property.objects.get(name = property_name)
                    except Property.DoesNotExist:
                        error_log.error("Властивості з назвою "+ property_name +" не
знайдено.")
                        property = None
                    if property:
                        try:
                            property_value = Propertyvalue.objects.get(property =
property, value = properties[property_name])
```

```
                    modelinstruments = modelinstruments.filter(properties =
property_value)
                except Propertyvalue.DoesNotExist:
                    error_log.error("Властивості зі значенням "+
properties[property_name] +" для назви "+ property_name +" не знайдено.")
        properties = Propertyvalue.objects.all()
        families = Instrumentfamily.objects.all()
        types = Instrumenttype.objects.all()
        brands = Instrumentbrand.objects.all()
        return render(request, "cozymirs/search_models.html",
{"modelinstruments": modelinstruments, "families": families, "types": types,
"brands": brands })


    def search_instruments_view(request, type = None, brand = None, family =
None, properties = None, city = None, country = None, condition_min = None,
condition_max = None, buyoption = False, leasingoption = False, price_min = 0,
price_max = None, start = None, end = None, level = None, no_deposit = False,
review_min = 0, review_max = 5):
        musicalinstruments = Musicalinstrument.objects.filter(actual = True)
        if type:
            try:
                instrumenttype = Instrumenttype.objects.get(id = type)
                musicalinstruments =
musicalinstruments.filter(instrumentmodel__instrumenttype = instrumenttype)
            except Instrumenttype.DoesNotExist:
                error_log.error("Типу музичного інструменту з номером "+ type
+" не знайдено.")
        if brand:
            try:
                instrumentbrand = Instrumentbrand.objects.get(id = brand)
```

```
                musicalinstruments =
musicalinstruments.filter(instrumentmodel__instrumentbrand = instrumentbrand)
            except Instrumentbrand.DoesNotExist:
                error_log.error("Виробника музичних інструментів з номером "+
brand +" не знайдено.")
        if family:
            try:
                instrumentfamily = Instrumentfamily.objects.get(id = family)
                musicalinstruments =
musicalinstruments.filter(instrumentmodel__instrumenttype__instrumentfamily =
instrumentfamily)
            except Instrumentfamily.DoesNotExist:
                error_log.error("Сімейства музичних інструментів з номером "+
family +" не знайдено.")
            for property_name in properties.keys():
                try:
                    property = Property.objects.get(name = property_name)
                except Property.DoesNotExist:
                    error_log.error("Властивості з назвою "+ property_name +" не
знайдено.")
                    property = None
                if property:
                    try:
                        property_value = Propertyvalue.objects.get(property =
property, value = properties[property_name])
                        musicalinstruments =
musicalinstruments.filter(Q(instrumentmodel__properties = property_value) |
Q(properties = property_value))
                    except Propertyvalue.DoesNotExist:
```

```
                    error_log.error("Властивості зі значенням "+
properties[property_name] +" для назви "+ property_name +" не знайдено.")
        if city:
            try:
                city = City.objects.get(name = city)
                musicalinstruments = musicalinstruments.filter(city = city)
            except City.DoesNotExist:
                error_log.error("Міста з номером "+ city +" не знайдено.")
        if city:
            try:
                city = City.objects.get(name = city)
                musicalinstruments = musicalinstruments.filter(city = city)
            except City.DoesNotExist:
                error_log.error("Міста з номером "+ city +" не знайдено.")
        if country:
            musicalinstruments = musicalinstruments.filter(city__country =
country)
        if condition_min:
            musicalinstruments = musicalinstruments.filter(condition__gte =
condition_min)
        if condition_max:
            musicalinstruments = musicalinstruments.filter(condition__lte =
condition_max)
        if buyoption:
            musicalinstruments = musicalinstruments.filter(buyoption = True)
        if leasingoption:
            musicalinstruments = musicalinstruments.filter(leasingoption = True)
        if level:
            musicalinstruments = musicalinstruments.filter(fromlevel__lte = level,
tolevel__gte = level)
```

```
            if no_deposit:
                musicalinstruments = musicalinstruments.filter(deposit = 0)
            if review_min > 0:
                musicalinstruments = musicalinstruments.filter(reviewed__gte =
review_min)
            if review_max < 5:
                musicalinstruments = musicalinstruments.filter(reviewed__lte =
review_max)
            if not start:
                start = datetime.datetime.now()
            if not end:
                end = start + datetime.timedelta(months = 1)
            musicalinstruments = musicalinstruments.filter(Q(deadline__is_null =
True) | Q(deadline__gte = end))
            musicalinstruments = musicalinstruments.oder_by("-reviewed")
            musinstruments = []
            rental_time = end - days
            rental_days = rental_time.days
            rental_weeks = math.ceil(rental_days / 7)
            rental_months = rental_time.months
            rental_months_check = start + datetime.timedelta(months =
rental_time.months)
            if end != rental_months_check:
                rental_months = rental_months + 1
            rental_years = rental_time.years
            rental_years_check = start + datetime.timedelta(years =
rental_time.years)
            if end != rental_years_check:
                rental_years = rental_years + 1
            try:
```

```
                tenant = Tenant.objects.get(user =
request.user).prefetch_related("savedinstruments")
            except Tenant.DoesNotExist:
                tenant = None
            for musicalinstrument in musicalinstruments:
                if tenant:
                    saved = False
                    if musicalinstrument in tenant.savedinstruments:
                        saved = True
                    musicalinstrument.saved = saved
                bookings = Rentalbooking.objects.filter(musicalinstrument =
musicalinstrument, end__gte = start, end__lte = end, state__gte = 2)
                if not bookings:
                    bookings = Rentalbooking.objects.filter(musicalinstrument =
musicalinstrument, start__gte = start, start__lte = end, state__gte = 2)
                    if not bookings:
                        bookings = Rentalbooking.objects.filter(musicalinstrument =
musicalinstrument, start__lte = start, end__gte = end, state__gte = 2)
                        if not bookings:
                            current_time = datetime.datetime.now()
                            bookings = Rentalbooking.objects.filter(musicalinstrument =
musicalinstrument, end__lte = start, state__gte = 2, end__gt = current_time,
leasing = True)
                            if bookings:
                                musicalinstrument.inleasing = True
                            else:
                                musicalinstrument.inleasing = False
                            for proposition in musicalinstrument.propositions:
                                price = 0
                                price_calc = None
```

```
                        if proposition.period == 0:
                            if rental_days >= proposition.periods:
                                price = proposition.price * rental_days
                        elif proposition.period == 1:
                            if rental_weeks >= proposition.periods:
                                price = proposition.price * rental_weeks
                        elif proposition.period == 2:
                            if rental_months >= proposition.periods:
                                price = proposition.price * rental_months
                        elif proposition.period == 3:
                            if rental_years >= proposition.periods:
                                price = proposition.price * rental_years
                            if price_calc and price > price_calc:
                                price_calc = price
                            elif not price_calc:
                                price_calc = price
                    if (price_max and price_calc >= price_min and price_calc <=
price_max) or (not price_max and price_calc >= price_min):
                        musinstruments.append(musicalinstrument)
            properties = Propertyvalue.objects.all()
            families = Instrumentfamily.objects.all()
            types = Instrumenttype.objects.all()
            brands = Instrumentbrand.objects.all()
            cities = City.objects.all()
            return render(request, "cozymirs/search_models.html",
{"musicalinstruments": musinstruments, "families": families, "types": types,
"brands": brands, "cities": cities })

        def musicalinstrument_view(request, musinstrument, start = None, end =
None):
```

```
            try:
                musicalinstrument = Musicalinstrument.objects.get(id =
musinstrument).prefetch_related("properties").select_related("city").select_related(
"owner").select_related("instrumentmodel").select_related("instrumentmodel__inst
rumenttype").select_related("instrumentmodel__instrumentbrand").prefetch_relate
d("instrumentmodel__instrumenttype__instrumentfamily").prefetch_related("phot
os").prefetch_related("instrumentmodel__properties").prefetch_related("propositio
ns").prefetch_related("accessories")
            except Musicalinstrument.DoesNotExist:
                error_log.error("Музичного інструменту з номером "+
musinstrument +" не знайдено.")
            try:
                tenant = Tenant.objects.get(user =
request.user).prefetch_related("savedinstruments").prefetch_related("waitinginstru
ments")
            except Tenant.DoesNotExist:
                error_log.error("Авторизованого орендаря з логіном "+
request.user.username+" не знайдено.")
                tenant = None
            if tenant and musicalinstrument:
                if not start:
                    start = datetime.datetime.now()
                if not end:
                    end = start + datetime.timedelta(months = 1)
                bookings = Rentalbooking.objects.filter(musicalinstrument =
musicalinstrument, end__gte = start, end__lte = end, state__gte = 2)
                if bookings:
                    musicalinstrument.free = False
                else:
```

```
                bookings = Rentalbooking.objects.filter(musicalinstrument =
musicalinstrument, start__gte = start, start__lte = end, state__gte = 2)
                if bookings:
                    musicalinstrument.free = False
                else:
                    bookings = Rentalbooking.objects.filter(musicalinstrument =
musicalinstrument, start__lte = start, end__gte = end, state__gte = 2)
                    if bookings:
                        musicalinstrument.free = False
                    else:
                        current_time = datetime.datetime.now()
                        bookings = Rentalbooking.objects.filter(musicalinstrument =
musicalinstrument, end__lte = start, state__gte = 2, end__gt = current_time,
leasing = True)
                        if bookings:
                            musicalinstrument.inleasing = True
                        else:
                            musicalinstrument.inleasing = False
                        price_calc = None
                        for proposition in musinstrument.propositions:
                            price = 0
                            if proposition.period == 0:
                                if rental_days >= proposition.periods:
                                    price = proposition.price * rental_days
                            elif proposition.period == 1:
                                if rental_weeks >= proposition.periods:
                                    price = proposition.price * rental_weeks
                            elif proposition.period == 2:
                                if rental_months >= proposition.periods:
                                    price = proposition.price * rental_months
```

```
                elif  proposition.period == 3:
                    if rental_years >= proposition.periods:
                        price = proposition.price * rental_years
                    if price_calc and price > price_calc:
                        price_calc = price
                    elif not price_calc:
                        price_calc = price
                musicalinstrument.price = price_calc
                musicalinstrument.free = True
        instrumentreviews = Instrumentreview.objects.filter(musicalinstrument =
musicalinstrument).select_related("tenant").order_by("-date")[:3]
        saved = False
        if musicalinstrument in tenant.savedinstruments:
            saved = True
        waited = False
        waiteable = False
        if musicalinstrument in tenant.waitinginstruments:
            waited = True
        else:
            booking = Rentalbooking.objects.filter(musicalinstrument =
musicalinstrument, state__gte = 2).order_by("-end")[0]
            if not musicalinstrument.deadline or musicalinstrument.deadline >
booking.end:
                waiteable = True
        return render(request, "cozymirs/musicalinstrument.html",
{"musicalinstrument": musicalinstrument, "reviews": instrumentreviews, "saved" :
saved, "waited" : waited, , "waiteable" : waiteable })


    def save_musinstrument_view(request, musinstrument):
        try:
```

```
            tenant = Tenant.objects.get(user =
request.user).prefetch_related("savedinstruments")
        except Tenant.DoesNotExist:
            error_log.error("Авторизованого орендаря з логіном "+
request.user.username+" не знайдено.")
            tenant = None
        try:
            musinstrument = Musicalinstrument.objects.get(id = musinstrument)
        except Musicalinstrument.DoesNotExist:
            error_log.error("Музичного інструменту з номером "+
musinstrument +" не знайдено.")
            musinstrument = None
        if tenant and musinstrument in tenant.savedinstruments:
            tenant.savedinstruments.remove(musinstrument)
            return redirect("savedinstruments")
        elif tenant and not (musinstrument in tenant.savedinstruments):
            tenant.savedinstruments.add(musinstrument)
            return redirect("savedinstruments")


    def wait_musinstrument_view(request, musinstrument):
        try:
            tenant = Tenant.objects.get(user =
request.user).prefetch_related("waitinginstruments")
        except Tenant.DoesNotExist:
            error_log.error("Авторизованого орендаря з логіном "+
request.user.username+" не знайдено.")
            tenant = None
        try:
            musinstrument = Musicalinstrument.objects.get(id = musinstrument)
        except Musicalinstrument.DoesNotExist:
```

```
            error_log.error("Музичного інструменту з номером "+
musinstrument +" не знайдено.")
            musinstrument = None
        waiteable = False
        booking = Rentalbooking.objects.filter(musicalinstrument =
musinstrument, state__gte = 2).order_by("-end")[0]
        if not musinstrument.deadline or musinstrument.deadline > booking.end:
            waiteable = True
        if tenant and musinstrument in tenant.waitinginstruments:
            tenant.waitinginstruments.remove(musinstrument)
            return redirect("waitinginstruments")
        elif tenant and not (musinstrument in tenant.waitinginstruments) and
waiteable:
            tenant.waitinginstruments.add(musinstrument)
            return redirect("waitinginstruments")


    def locate_musicalinstrument_view(request):
        try:
            owner = Owner.objects.get(user = request.user)
        except Owner.DoesNotExist:
            error_log.error("Авторизованого власника з логіном "+
request.user.username+" не знайдено.")
            owner = None
        if owner and request.user.is_active:
            if request.method == "POST":
                instrument_form = MusicalinstrumentForm(request.POST)
                if instrument_form.is_valid():
                    instrument = instrument_form.save(commit=False)
                    instrument.owner = owner
                    instrument.save()
```

```
                    instrument.save_m2m()
                    return redirect("ownerinstruments")
                else:
                    instrument_form = MusicalinstrumentForm()
            else:
                instrument_form = None
            propositions = Proposition.objects.filter(owner = owner).order_by("-id")
            return render(request, "cozymirs/instrumentlocate.html", {"form":
instrument_form,  "propositions": propositions})


        def locate_instreview_view(request, musinstrument):
            try:
                tenant = Tenant.objects.get(user = request.user)
            except Tenant.DoesNotExist:
                error_log.error("Авторизованого орендаря з логіном "+
request.user.username+" не знайдено.")
                tenant = None
            try:
                musicalinstrument = Musicalinstrument.objects.get(id =
musinstrument)
            except Musicalinstrument.DoesNotExist:
                error_log.error("Музичного інструменту з номером "+
musinstrument +" не знайдено.")
                musicalinstrument = None
            reviewable = False
            bookings = []
            if tenant and musicalinstrument:
                reviews = Instrumentreview.objects.filter(musicalinstrument =
musicalinstrument, tenant = tenant).order_by("-date")[0]
                reviewed = reviews.date
```

```
            bookings = Rentalbooking.objects.filter(musicalinstrument =
musicalinstrument, tenant = tenant, state__gte = 2, date__gt = reviewed)
        if tenant and tenant.user.is_active and musicalinstrument and bookings:
            if request.method == "POST":
                instreview_form = InstrumentreviewForm(request.POST)
                if instreview_form.is_valid():
                    instreview = instreview_form.save(commit=False)
                    instrument.tenant = tenant
                    instrument.musicalinstrument = musicalinstrument
                    instreview.save()
                    instreviews = Instrumentreview.objects.filter(musicalinstrument
= musicalinstrument)
                    result = 0
                    q = 0
                    for instreview in instreviews:
                        result = result + instreview.mark
                        q = q + 1
                    musicalinstrument.reviewed = result / q
                    musicalinstrument.save()
                    return redirect("instreviews", musicalinstrument.id)
            else:
                instrument_form = MusicalinstrumentForm()
        else:
            instrument_form = None
        return render(request, "cozymirs/instreview.html", {"form":
instrument_form })


    def locate_ownerreview_view(request, owner):
        try:
            tenant = Tenant.objects.get(user = request.user)
```

```
            except Tenant.DoesNotExist:
                error_log.error("Авторизованого орендаря з логіном "+
request.user.username+" не знайдено.")
                tenant = None
            try:
                owner = Owner.objects.get(user__username = owner)
            except Owner.DoesNotExist:
                error_log.error("Власника з логіном "+ owner +" не знайдено.")
                owner = None
            reviewable = False
            bookings = []
            if tenant and musicalinstrument:
                reviews = Ownerreview.objects.filter(owner =
musicalinstrument.onwer, tenant = tenant).order_by("-date")[0]
                reviewed = reviews.date
                bookings = Rentalbooking.objects.filter(musicalinstrument =
musicalinstrument, tenant = tenant, date__gt = reviewed)
            if tenant and tenant.user.is_active and musicalinstrument and bookings:
                if request.method == "POST":
                    ownerreview_form = OwnerreviewForm(request.POST)
                    if ownerreview_form.is_valid():
                        ownerreview = ownerreview_form.save(commit=False)
                        ownerreview.tenant = tenant
                        ownerreview.owner = owner
                        ownerreview.save()
                        ownerreviews = Ownerreview.objects.filter(owner = owner)
                        result = 0
                        q = 0
                        for ownerreview in ownerreviews:
                            result = result + ownerreview.mark
```

```
                    q = q + 1
                owner.reviewed = result / q
                owner.save()
                return redirect("ownerreviews", owner.user.username)
            else:
                ownerreview_form = OwnerreviewForm()
        else:
            ownerreview_form = None
        return render(request, "cozymirs/ownerreview.html", {"form":
ownerreview_form })

    def locate_tenantreview_view(request, tenant):
        try:
            owner = Owner.objects.get(user = request.user)
        except Owner.DoesNotExist:
            error_log.error("Авторизованого власника з логіном "+ owner +" не
знайдено.")
            owner = None
        try:
            tenant = Tenant.objects.get(user__username = tenant)
        except Tenant.DoesNotExist:
            error_log.error("Орендаря з логіном "+ request.user.username+" не
знайдено.")
            tenant = None
        reviewable = False
        bookings = []
        if tenant and owner:
            reviews = Tenantreview.objects.filter(owner = owner, tenant =
tenant).order_by("-date")[0]
            reviewed = reviews.date
```

```
            bookings = Rentalbooking.objects.filter(musicalinstrument__owner =
owner, tenant = tenant, date__gt = reviewed)
        if tenant and request.user.is_active and owner and bookings:
            if request.method == "POST":
                tenantreview_form = TenantreviewForm(request.POST)
                if tenantreview_form.is_valid():
                    tenantreview = tenantreview_form.save(commit=False)
                    tenantreview.tenant = tenant
                    tenantreview.owner = owner
                    tenantreview.save()
                    tenantreviews = Tenantreview.objects.filter(tenant = tenant)
                    result = 0
                    q = 0
                    for tenantreview in tenantreviews:
                        result = result + tenantreview.mark
                        q = q + 1
                    tenant.reviewed = result / q
                    tenant.save()
                    return redirect("tenantreviews", tenant.user.username)
            else:
                tenantreview_form = TenantreviewForm()
        else:
            tenantreview_form = None
        return render(request, "cozymirs/tenantreview.html", {"form":
tenantreview_form })

    def locate_proposition_view(request):
        try:
            owner = Owner.objects.get(user = request.user)
        except Owner.DoesNotExist:
```

```
            error_log.error("Авторизованого власника з логіном "+ owner +" не
знайдено.")
            owner = None
        if request.user.is_active and owner:
            if request.method == "POST":
                proposition_form = PropositionForm(request.POST)
                if proposition_form.is_valid():
                    proposition = proposition_form.save(commit=False)
                    proposition.owner = owner
                    proposition.save()
                    return redirect("propositions")
            else:
                proposition_form = PropositionForm()
        else:
            proposition_form = None
        return render(request, "cozymirs/proposition.html", {"form":
proposition_form })


    def edit_proposition_view(request, proposition):
        try:
            owner = Owner.objects.get(user = request.user)
        except Owner.DoesNotExist:
            error_log.error("Авторизованого власника з логіном "+ owner +" не
знайдено.")
            owner = None
        try:
            proposition = Proposition.objects.get(id =
proposition).select_related("owner")
        except Proposition.DoesNotExist:
            proposition = None
```

```
            if request.user.is_active and owner and proposition.owner == owner:
                if request.method == "POST":
                    proposition_form = PropositionForm(request.POST, instance =
proposition)
                    if proposition_form.is_valid():
                        proposition_form.save()
                        proposition_form.save_m2m()
                        return redirect("propositions")
                else:
                    proposition_form = PropositionForm(None, instance = proposition)
            else:
                proposition_form = None
            return render(request, "cozymirs/proposition.html", {"form":
proposition_form })


        def edit_musinstrument_view(request, musinstrument):
            try:
                owner = Owner.objects.get(user = request.user)
            except Owner.DoesNotExist:
                error_log.error("Авторизованого власника з логіном "+ owner +" не
знайдено.")
                owner = None
            try:
                musinstrument = Musicalinstrument.objects.get(id =
musinstrument).select_related("owner")
            except Musicalinstrument.DoesNotExist:
                musinstrument = None
            if request.user.is_active and owner and musinstrument.owner == owner:
                if request.method == "POST":
```

```
                instrument_form = MusicalinstrumentForm(request.POST, instance
= musinstrument)
                if instrument_form.is_valid():
                    instrument_form.save()
                    return redirect("ownerinstruments")
            else:
                instrument_form = MusicalinstrumentForm(None, instance =
proposition)
        else:
            instrument_form = None
        return render(request, "cozymirs/instrumentlocate.html", {"form":
instrument_form })
```

# ДОДАТОК В

## Слайди презентації

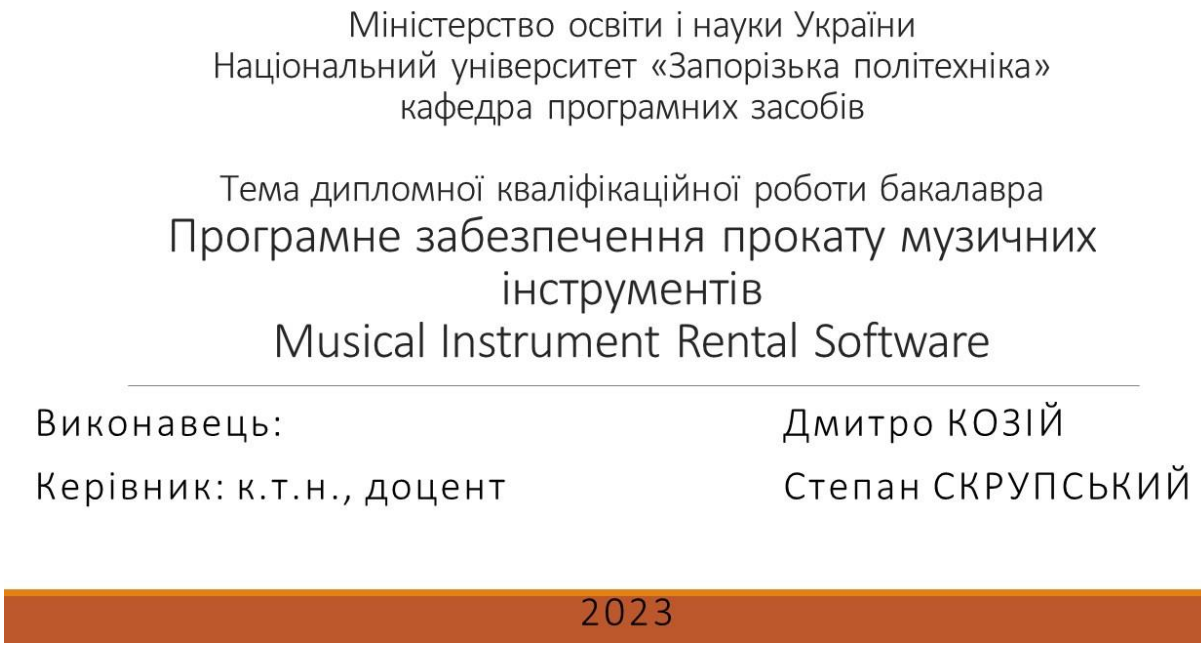


Рисунок В.1 – Титульний слайд

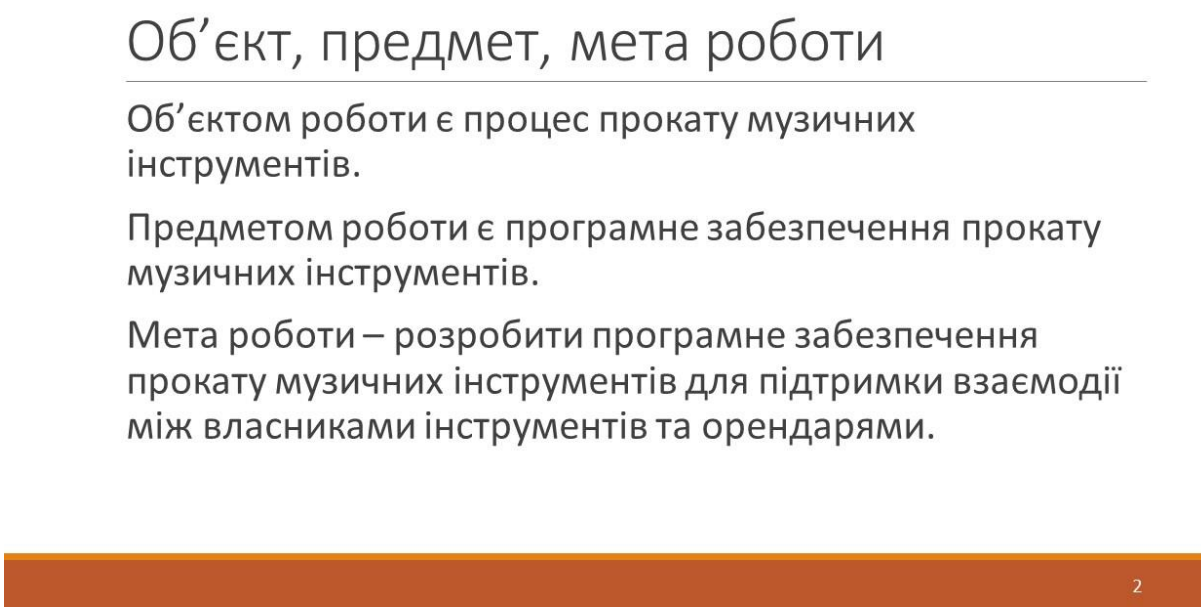


Рисунок В.2 – Об’єкт, предмет, мета роботи

## Завдання роботи

– аналіз програмних засобів прокату музичних інструментів і визначення концепції програми;

– проєктування програмного забезпечення прокату музичних інструментів;

– реалізація програмного забезпечення прокату музичних інструментів.

3

Рисунок В.3 – Завдання роботи

## Порівняння програмних засобів прокату музичних інструментів

| Критерій | Rentmy-instrument | Grover | FriendWithA | CozyMIRS |
|---|---|---|---|---|
| Концепція | Служба прокату | Служба прокату | Прокат від власників | Прокат від власників |
| Територія дії | США (Канзас, служба доставки) | Німеччина, Нідерланди, Іспанія, США | США | Україна |
| Пошук інструментів | Відсутній | Присутній | Присутній | Присутній |
| Відгуки про інструменти | Відсутні | Присутні | Відсутні | Присутні |
| Списки бажання | Відсутні | Присутні | Присутні | Присутні |
| Списки очікування | Відсутні | Відсутні | Відсутні | Присутні |
| Період прокату | Місяць | Місяць | День, тиждень, місяць | Від одного дня |
| Придбання інструменту | Можливе | Можливе | Можливе | Можливе |

4

Рисунок В.4 – Порівняння програмних засобів прокату музичних інструментів

## Вибір мови для реалізації програми

| Критерій | Python | PHP |
|---|---|---|
| Перевага в довготривалих проєктах | Так | Ні |
| Продуктивність | Висока | Вища |
| Налагодження | Швидше | Довше |
| Підтримка коду | Легша | Складніша |
| Швидкість написання коду | Вища | Висока |

5

Рисунок В.5 – Вибір мови для реалізації програми

## Діаграми варіантів використання

6

Рисунок В.6 – Діаграми варіантів використання


Рисунок В.7 – Схема бази даних

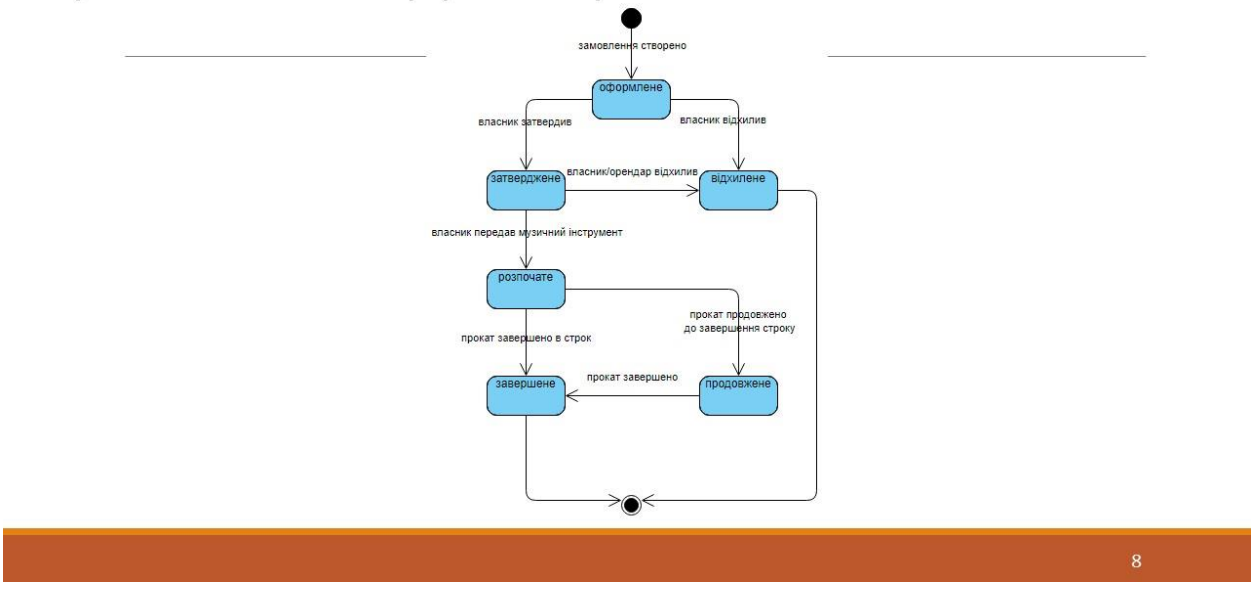


Рисунок В.8 – Діаграма станів замовлення прокату музичного інструменту

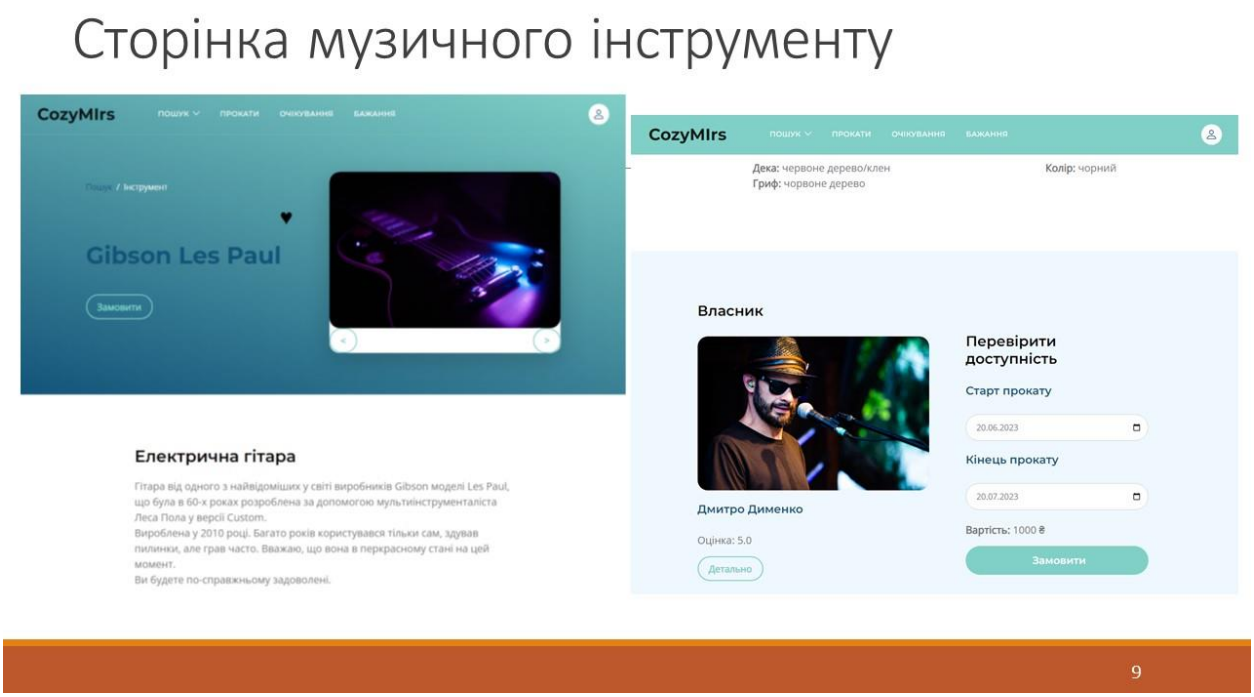


Рисунок В.9 – Сторінка музичного інструменту

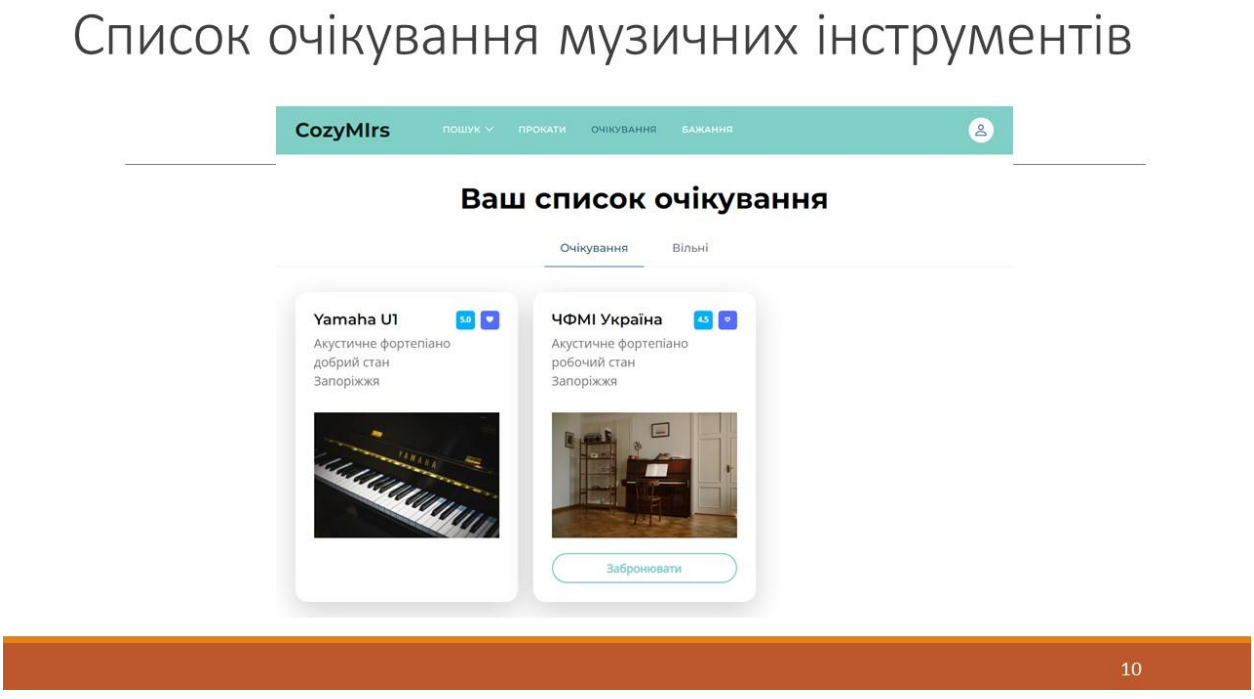


Рисунок В.10 – Список очікування музичних інструментів

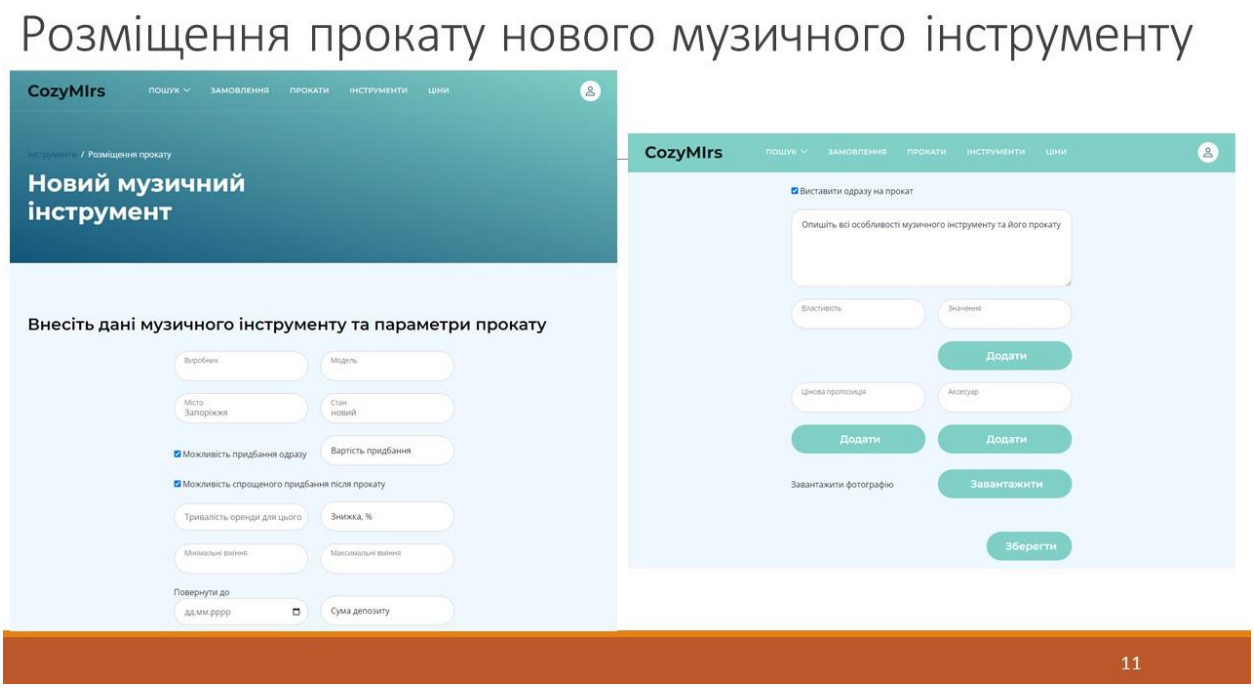


Рисунок В.11 – Розміщення прокату нового музичного інструменту


Рисунок В.12 – Висновки