

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проекту (роботи)

бакалавра

(ступінь вищої освіти)

на тему ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ПОШУКУ ПРОСТОРУ

(назва теми)

ДЛЯ ПРОВЕДЕННЯ ПОДІЙ. SOFTWARE TO SEARCH FOR

SPACE FOR HOLDING EVENTS

Виконав(ла): студент(ка) 4 курсу, групи КНТ-110

Спеціальності 121 Інженерія програмного

(код і найменування спеціальності)

забезпечення

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

ШАНІН К.А.

(ПРІЗВИЩЕ та ініціали)

Керівник ЛЬОВКІН В.М.

(ПРІЗВИЩЕ та ініціали)

Рецензент ГОЛУБ Т.В.

(ПРІЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

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

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф. _____
Сергій СУББОТІН
« _____ » _____ 2024 року

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

ШАНІНА Кирила Андрійовича

(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Програмне забезпечення пошуку простору для проведення подій. Software to Search for Space for Holding Events

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

затверджені наказом закладу вищої освіти від « 24 » квітня 2024 року № 168

2. Строк подання студентом проєкту (роботи) 10 червня 2024 року

3. Вихідні дані до проєкту (роботи) технічне завдання

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

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

Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	ПРИЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	ЛЬОВКІН В.М., доцент		
Нормоконтроль	ДЕЙНЕГА Л.Ю., ст. викладач		

7. Дата видачі завдання « 15 » квітня 2024 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Срок 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)

(підпис) Кирило ШАНІН
(Ім'я ПРИЗВИЩЕ)

Керівник проєкту (роботи)

(підпис) Валерій ЛЬОВКІН
(Ім'я ПРИЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 92 с., 27 рис., 3 табл., 3 дод., 9 джерел.

ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, ПРОСТІР, ПРОВЕДЕННЯ ПОДІЙ,
МОДЕЛЬ-ПРЕДСТАВЛЕННЯ-ШАБЛОН, ВЕБДОДАТОК.

Об'єкт роботи – процес пошуку простору для проведення подій. Предмет роботи – програмне забезпечення пошуку простору для проведення подій. Мета роботи – розробити програмне забезпечення пошуку простору, який міг би забезпечити необхідну підтримку проведення події.

Матеріали, методи та технічні засоби: мова Python, система підтримки баз даних MySQL, фреймворк Django.

Результати. У результаті виконаної роботи створено програму, що забезпечує потреби представників просторів для проведення подій та відвідувачів-орендарів. Представники просторів для проведення подій мають можливість використовувати її для визначення власних просторів для проведення подій, керування даними про них та доступність для оренди, створеними відвідувачами-орендарями замовленнями оренди просторів для проведення подій. Відвідувачі-орендарі мають можливість шукати такий простір, який дозволить задовольнити їхні вимоги щодо проведення події, замовляти оренду такого простору.

Висновки. Розроблена програма дозволяє не тільки шукати приміщення для проведення подій, а працювати з ними саме як з просторами за рахунок вибору потрібного додаткового інвентаря для проведення подій, потрібних варіантів харчування.

Галузь використання. Організація і проведення подій стосовно пошуку і оренди просторів для проведення таких подій.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 92 pages, 27 figures, 3 tables, 3 appendixes, 9 sources.

SOFTWARE, SPACE, EVENT HOSTING, MODEL-VIEW- TEMPLATE, WEB APPLICATION.

The object of the work is the process of finding a space for holding events. The subject of work is software for finding space for events. The goal of the work is to develop software for finding a space that could provide the necessary support for the event.

Materials, methods and technical means: Python language, MySQL database support system, Django framework.

The results. As a result of the work performed, a program was created that meets the needs of event space representatives and tenant visitors. Event space representatives can use it to identify their own event spaces, manage data on them, and make them available for rent by event space rental orders created by visiting tenants. Visitors-renters have the opportunity to search for a space that will meet their requirements for holding an event, to order the rental of such a space.

Conclusions. The developed program allows not only to search for premises for holding events, but to work with them exactly as with spaces by choosing the necessary additional inventory for holding events, the necessary food options.

Field of use. Organization and holding of events in relation to finding and renting spaces for such events.

ЗМІСТ

	С.
Перелік скорочень та умовних познач.....	7
Вступ.....	8
1 Аналіз предметної області.....	9
1.1 Огляд програм пошуку простору для проведення подій	9
1.2 Порівняння аналогів для визначення характеристик власної розробки	14
1.3 Висновки за розділом 1	15
2 Проєктування програми.....	16
2.1 Вибір засобів проєктування і розробки	16
2.2 Моделювання взаємодії користувачів з програмою.....	16
2.3 Визначення структури бази даних	21
2.4 Висновки за розділом 2	35
3 Розробка програми	36
3.1 Визначення послідовності роботи з програмою	36
3.2 Особливості програмної реалізації.....	36
3.3 Висновки за розділом 3	49
4 Експлуатація та тестування програми	50
4.1 Призначення програми	50
4.2 Умови виконання програми	50
4.3 Експлуатація програми.....	50
4.4 Тестування програми	55
4.5 Висновки за розділом 4	57
Висновки	58
Перелік джерел посилання	59
Додаток А Технічне завдання	60
Додаток Б Текст програми	65
Додаток В Слайди презентації.....	85

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

БД – база даних;

МПШ – модель-представлення-шаблон.

ВСТУП

Простори для проведення подій є особливими місцями, які не зовсім відповідають звичайному житлу. Тому при виконанні оренди таких просторів необхідно це враховувати, надаючи можливість задати, визначити, застосувати ті особливості, які характерні для просторів для проведення подій. Наприклад, простори можуть орендуватися для проведення подій на специфічну тривалість. Якщо для житла це або місяць (певна кількість місяців), або день (певна кількість днів), то для просторів для проведення подій це може бути повна доба, половина доби, погодинна оренда. До того ж може додаватися спеціальне обладнання, яке необхідне, щоб провести весілля, конференцію. Тобто це призводить до того, що орендується не просто приміщення, а саме простір для проведення подій. Основна мета – забезпечити повноцінне проведення події необхідного типу в орендованому для цього приміщенні. Необхідність врахування всього цього і диктує потребу в розробці окремого програмного забезпечення. З такими рішеннями в Україні є достатньо багато проблем на даний момент.

Відповідно дану ідею було виражено при формулюванні мети роботи – розробити програмне забезпечення пошуку простору, який міг би забезпечити необхідну підтримку проведення події.

Мета роботи має включати завдання, досягнення яких сукупно має її вирішити:

- проаналізувати проблему розробки програми пошуку простору для проведення подій;
- спроектувати програмне забезпечення пошуку простору для проведення подій;
- реалізувати програмне забезпечення пошуку простору для проведення подій.

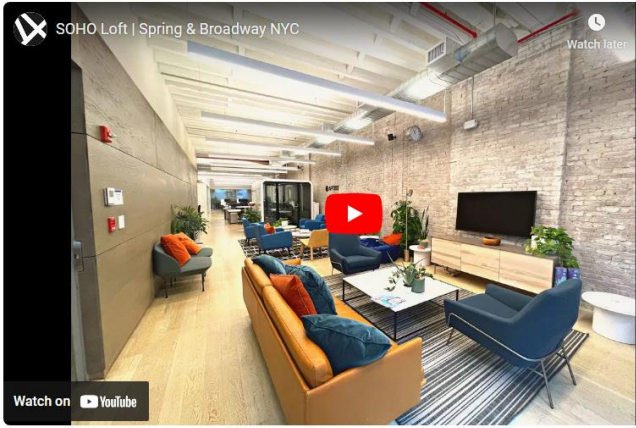
1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд програм пошуку простору для проведення подій

Storefront [1] (рис. 1.1) є першим програмним засобом простору для проведення подій, який представлений в результатах огляду.

Storefront [Find a space](#) [List your space](#) [Estimate Listing Revenue](#) [Magazine](#) [Help](#) [Sign up](#) [Log in](#)

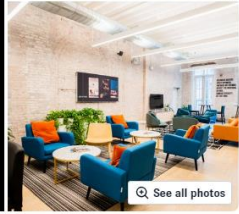
Event Space in the heart of SoHo
[Event spaces in New York](#) [Save to favorites](#)



SOHO Loft | Spring & Broadway NYC

Watch later Share

Watch on YouTube

See all photos

Event space in New York
[New York, United States](#)

Event Space Partial 3,000 sq ft

Ideal uses

Retail	Showroom	Event	Art	Food
Photo Shooting	Conference	Meeting	Office	

Put in your dates to get an estimated price

\$810.00 per day

Start date > End date

\$810.00 per day

Start date > End date

Request more information
(No credit card required)

The owner typically responds within 4 hours

About this space

This space is located in the heart of Soho surrounded by amazing Soho brick roads. Best location in the City for your event! Link for the video below.

Featuring an open space with a capacity of up to 60 people. Fully renovated building with Broadway street entrance with the elevator in the building. The space features professional adjustable lighting along with many different stands for product display. That's a LOFT space with hardwood floors and exposed brick walls giving it a rough yet...

[Show more >](#)

Amenities

Internet	Handicap Accessible	Electricity
Air Conditioning	Heating	Toilets
Kitchen	Security System	Stock Room
Counters	Fitting Rooms	Lighting
Furniture	Office Equipment	Sound & Video Equipment

Рисунок 1.1 – Сторінка простору для проведення подій у додатку Storefront

Storefront дозволяє шукати потрібний простір, виставляти свої простори для оренди. Кожен простір для проведення подій на своїй сторінці визначає ті варіанти використання його, які слід вважати ідеальними, опис, фотографії, зручності, місце розташування, доступність на потрібні дати, враження від відвідувачів-орендарів. Надається можливість відзначати простори для проведення подій як улюблені. Мінусом цього додатку є те, що простір може бути орендовано тільки на повну добу. Менший строк не допускається. Також програма працює тільки щодо того, щоб орендувати сам простір цілком, не надаючи додаткових можливостей, тобто як приміщення.

We Are Spaces [2] (рис. 1.2) дозволяє шукати простори для проведення подій в Сингапурі.

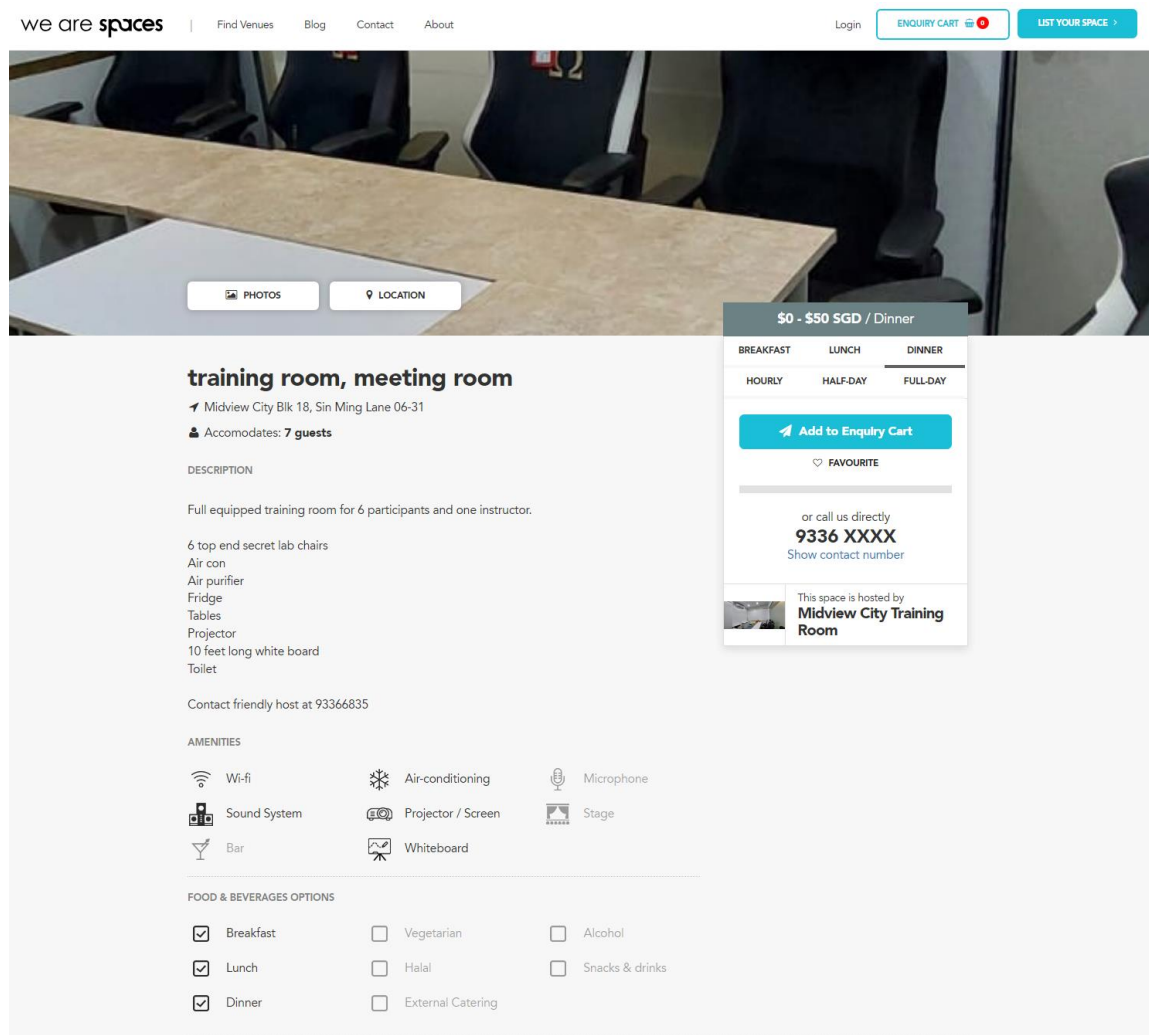


Рисунок 1.2 – Сторінка простору для проведення подій у додатку We Are Spaces

We Are Spaces також дозволяє шукати простори для проведення подій, хоча цей пошук має свої особливості, враховуючи, що пошук здійснюється тільки за одним містом (наприклад, задається частина міста). До параметрів пошуку також належить потрібні зручності, варіант харчування. Важливим є те, що вже під час замовлення оренди можна обрати потрібний варіант харчування, а оренда може виконуватися погодинно, на півдня або на день. Додаток також дозволяє відзначати простори для проведення подій як улюблені, виставляти свої простори для оренди. Надати враження про простір у програмі неможливо.

Splacer (рис. 1.3) [3] також дозволяє шукати, орендувати простори для проведення подій, в тому числі надаючи можливість виконувати оренду погодинно (хоча допускається вибір часу з кроком у 30 хвилин, але мінімальна тривалість складає 1 годину).

За кожним простором для різних варіантів застосування визначається певний план розташування гостей та відповідна кількість, що є деякою відмінністю саме цього додатку.

За кожним простором для проведення подій визначаються його фотографії, варіанти проведення подій, інформація про власника, опис простору, плани розташування гостей, часова доступність, зручності, доступність, сервіси (включені до оренди), місце розташування, що все разом відображається на сторінці простору.

Під час пошуку просторів для проведення подій можна визначити місто проведення, тип простору, кількість гостей, розмір простору (площу), вартість, зручності, необхідні сервіси, правила простору, особливості, район розташування. Проте відповідні сервіси дозволяють тільки перевірити за простором наявність такого сервісу, тобто замовити додаткові можливості, які не входять в стандарту вартість, неможливо. Це ж стосується і замовлення варіантів харчування – це недоступне.

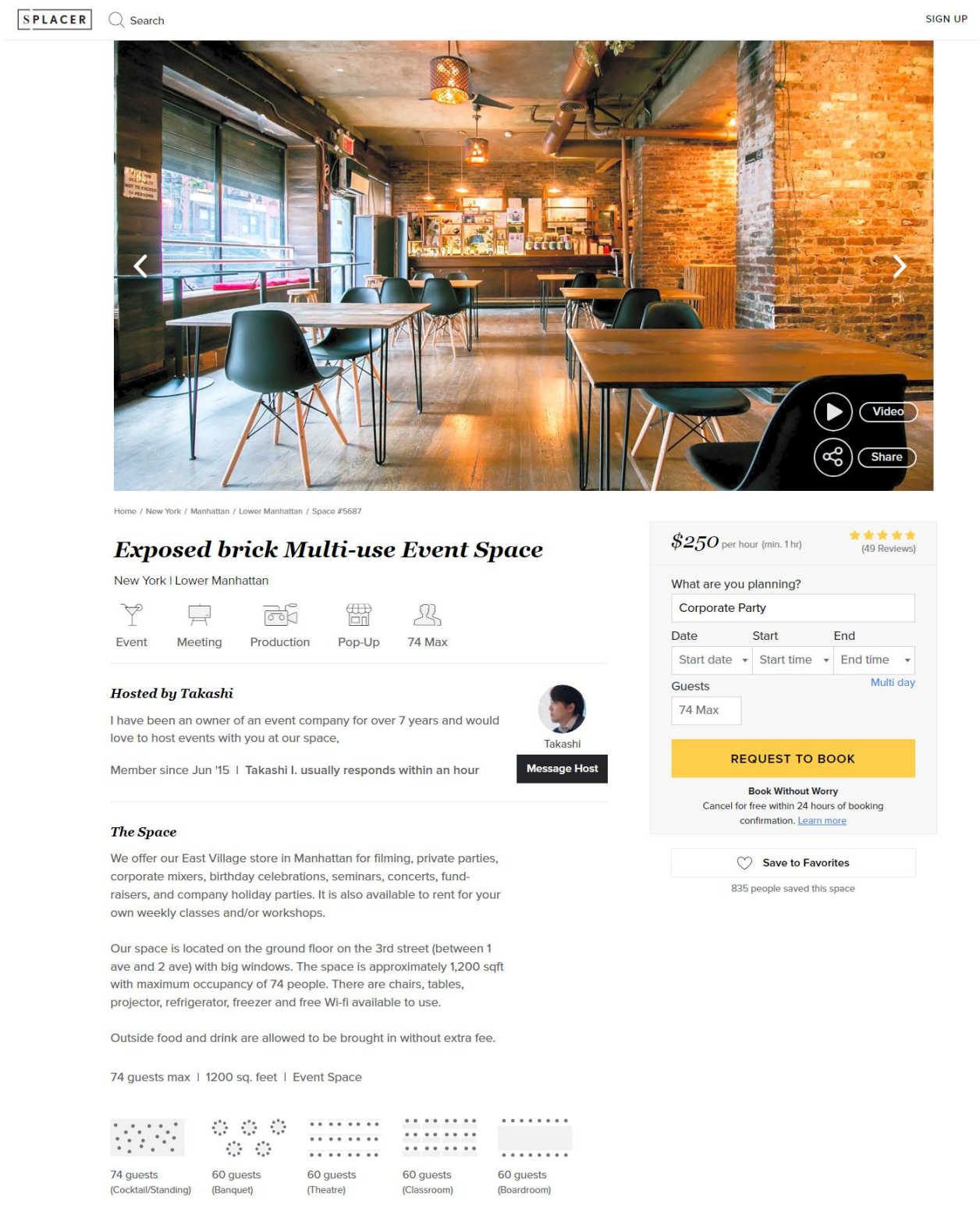


Рисунок 1.3 – Сторінка простору для проведення подій у додатку Splacer

Hire Space (рис. 1.4) [4] не підтримує як таку оренду, а забезпечує формування запитів до власника, таким чином підтримуючи подальше спілкування на основі наданих даних про користувача.

HIRE SPACE

Venues

Solutions

Ask An Expert

My Hire Space

PRIVATE DINING ROOM

Pergola on the Wharf

£ From £25 per person

20 Standing

Canary Wharf

Space description

Our Private Dining Room offers complete privacy away from our main restaurant for you and your guests. You will have the choice of 2 menus and our carefully curated drinks menu. The space is perfect to host colleagues, family or friends.

You will have access to the in house sound system, a flat screen tv and a dedicated team to look after you during your event.

Contact Pergola on the Wharf



Maximus Adams
Venue Expert

Add to enquiry

What this space offers

Features

- Flat Screen TV
- 2 Bars
- Cloakroom
- Covered Terrace
- DJ Booth and Decks

Food & Drink

- External Catering Allowed
- Corkage Charge
- Professional Kitchen
- Halal Catering

Space

- Disabled Access

Licenses

- Licensed Until 1am

Capacities

- 10 Boardroom
- 12 Dining
- 20 Standing

Welcomes

- Pop-Ups
- Amplified Events
- Own Music/ DJ Equipment
- 18th Birthday Parties
- 21st Birthday Parties
- Weddings
- Events with Alcohol

Рисунок 1.4 – Сторінка простору для проведення подій у додатку Hire Space

Зважаючи на описану концепцію, додатково варто вказати, що враження про простори користувачів не можуть залишати. Простір для проведення подій характеризується фотографіями, описом, особливостями, які включають харчування, сам простір, правилами визначення вартості. Але визначити, чи простір доступний, чи зайнятий на необхідний час, неможливо.

1.2 Порівняння аналогів для визначення характеристик власної розробки

Усі розглянуті програмні засоби, що можуть використовуватися для пошуку простору для проведення подій, були винесені для порівняння (табл. 1.1).

Таблиця 1.1 – Порівняння існуючих аналогів та розроблюваного програмного забезпечення пошуку простору для проведення подій

Характеристика	Storefront	We Are Spaces	Splacer	Hire Space	Розробка
Період оренди	Кількість діб	Погодинно, півдня, доба, кількість діб	Погодинно	—	Погодинно, півдня, доба, кількість діб
Варіанти харчування	—	+	—	—	+
Додатковий інвентар	—	—	—	—	+
Враження від просторів для проведення подій	+	—	+	—	+

Порівняння включало тільки ті ознаки, які були виділені в якості акцентів для забезпечення пошуку просторів для проведення подій. Було встановлено, що Hire Space тільки надає інформацію про простори для проведення подій, а переглянути інформацію про можливий час оренди взагалі не дозволяє. У той же час Storefront дозволяє виконувати оренду просторів

тільки на певну кількість днів, менший час недоступний. Всі інші таку можливість надають. Проте Splacer не визначає такий шаблон як півдня.

Вибір варіантів харчування доступний тільки в We Are Spaces та в програмі, що розробляється.

Жодний з існуючих аналогів не дозволяє вибирати додатковий інвентар, тобто включає тільки базовий варіант оренди і не дозволяє його розширювати. Це є особливістю розробки.

Не всі програми дозволяють ділитися враженнями від просторів для проведення подій, але це загалом важливо, оскільки дозволяє ділитися досвідом та формувати своє враження про простір для проведення подій. Storefront та Splacer такі можливості надають. Розробка підтримує це також.

1.3 Висновки за розділом 1

Виконано детальний огляд програм пошуку простору для проведення подій, включаючи Storefront, We Are Spaces, Splacer, Hire Space, виділяючи різні підходи до забезпечення роботи з просторами для проведення подій. Серед критеріїв порівняння виділено варіанти періоду оренди, що забезпечуються, підтримувані варіанти харчування під час оренди, доступність додаткового інвентаря під час оренди, можливість визначення вражень від просторів для проведення подій.

2 ПРОЄКТУВАННЯ ПРОГРАМИ

2.1 Вибір засобів проєктування і розробки

Вибір засобів розробки було зведено до єдиного вибору, який базувався на виборі фреймворку між Django [5]-[6] і Spring Boot [7]-[8] (табл. 2.1), але оскільки Django використовує мову Python, а Spring Boot – Java, то одночасно це був вибір і між мовами розробки. Кожен з цих варіантів має свої переваги, однак простота самого коду на Python, його читабельність, здатність до змінювання одразу на місці без необхідності завжди виділяти спершу класи, що характерно для Java, в умовах, коли проєкт створюється як дипломна кваліфікаційна робота, має певні переваги. Також в плюс до самого Django можна віднести наявність великої кількості готових засобів у ньому для роботи з даними, включаючи бази даних, адміністративну панель, підтримки безпеки. Тому обрано було Django.

Таблиця 2.1 – Порівняння фреймворків розробки програми пошуку простору для проведення подій

Критерій	Django	Spring Boot
Фокус розробки	Швидкість, керування даними	Масштабність
Підхід до побудови	Містить усі необхідні засоби одразу	Модульний
Мова програмування	Python	Java

2.2 Моделювання взаємодії користувачів з програмою

Розподіл між користувачами відбувся з визначенням гостя (користувача до авторизації), відвідувача-орендаря (користувача після авторизації, який шукає такий простір для проведення події, який відповідає

його вимогам), представника простору для проведення подій (власник цього простору або делегований для роботи в програмі єдиний представник цього простору).

Сценарії, якими в основному потоку може користуватися гість:

- пошук простору для проведення подій;
- демонстрування сторінки простору для проведення подій, включаючи перевірку параметрів оренди;
- авторизація (рис. 2.1).

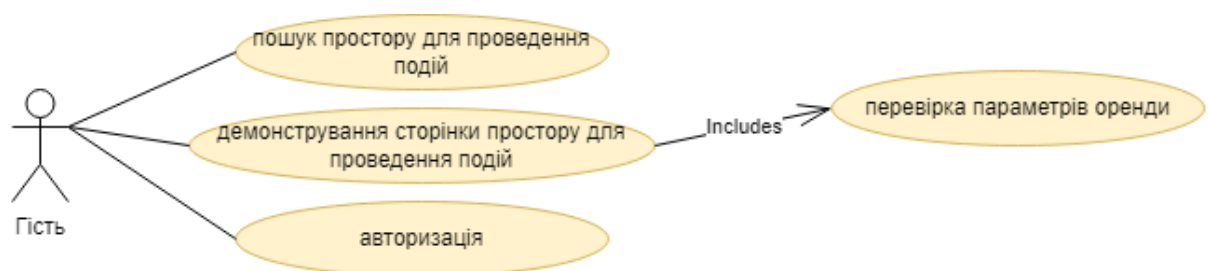


Рисунок 2.1 – Діаграма варіантів використання гостем програми пошуку простору для проведення подій

Альтернативний потік включає тільки один сценарій – реєстрація, що задіюється тоді, коли гість хоче авторизуватися, але не має запису користувача, тоді він може зареєструватися один з двох типів записів користувача (представника просторів для проведення подій або відвідувача-орендаря).

Сценарії, якими в основному потоку може користуватися відвідувач-орендар (рис. 2.2):

- замовлення оренди простору для проведення подій;
- керування своїм створеним замовленням оренди простору для проведення подій;
- формування переліку замовлень оренди простору для проведення подій, створених відвідувачем-орендарем;
- керування особистими даними відвідувачем-орендарем;
- висловлення враження про простір для проведення подій за

результатами оренди;

- формування переліку вражень за простором для проведення подій;
- формування переліку вражень за представником простору для проведення подій;
- демонстрування сторінки представника простору для проведення подій;
- демонстрування сторінки відвідувача-орендаря;
- формування переліку вражень за відвідувачем-орендарем;
- відзначення простору для проведення подій як улюбленого.

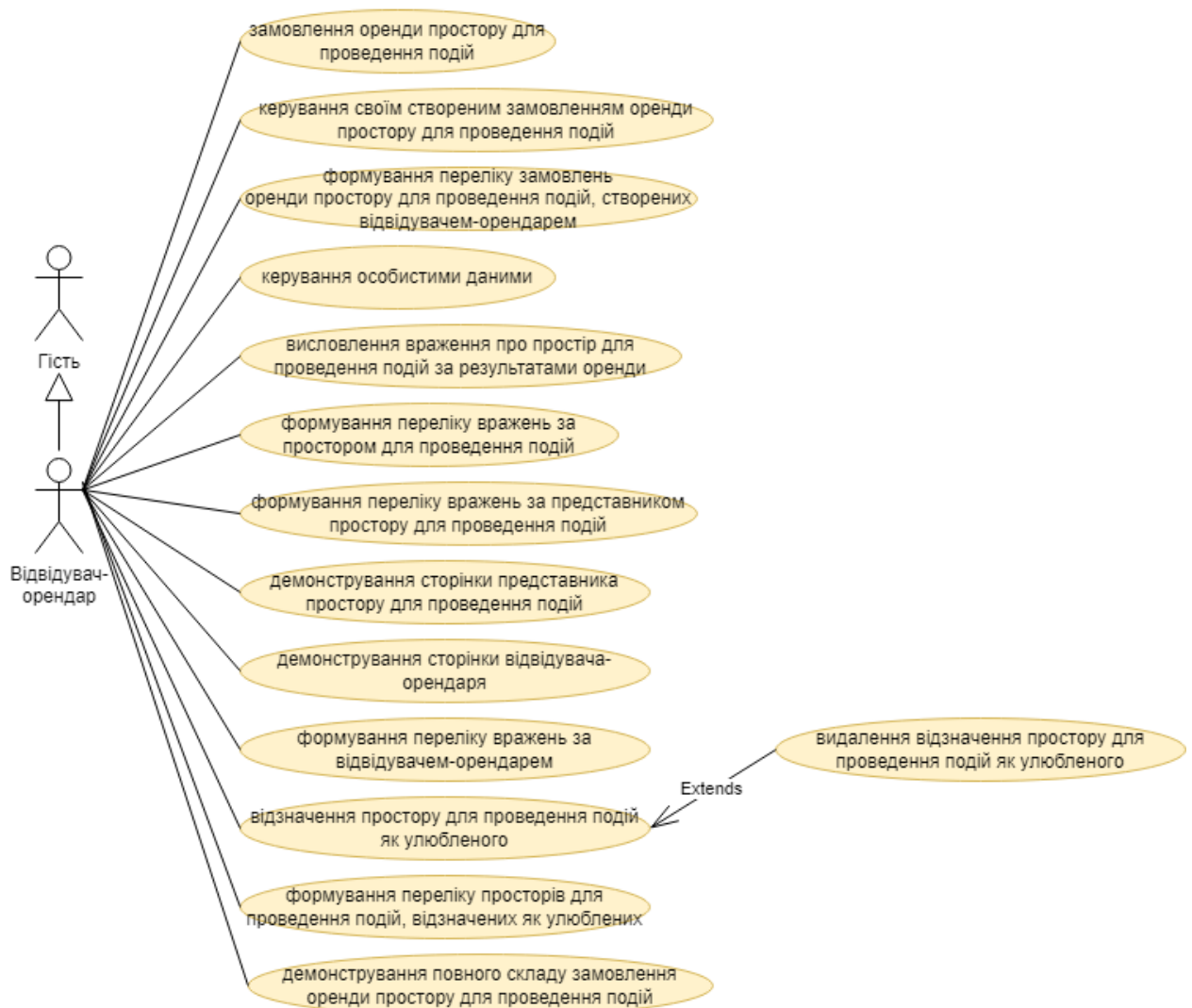


Рисунок 2.2 – Діаграма варіантів використання відвідувачем-орендарем програми пошуку простору для проведення подій

Також сценаріями є:

- формування переліку просторів для проведення подій, відзначених як улюблених;
- демонстрування повного складу замовлення оренди простору для проведення подій.

Альтернативний потік включає тільки один сценарій – видалення відзначення простору для проведення подій як улюбленого, що виконується стосовно відзначеного простору для проведення подій, який вже був відзначений улюбленим.

Сценарії, якими в основному потоку може користуватися представник простору для проведення подій:

- створення запису про новий простір для проведення подій;
- оновлення запису про простір для проведення подій;
- формування переліку просторів для проведення подій представника;
- пошук простору для проведення подій;
- демонстрування сторінки простору для проведення подій;
- керування створеним замовленням оренди простору для проведення подій;
- формування переліку замовлень оренди простору для проведення подій за простором для проведення подій;
- формування переліку замовлень оренди простору для проведення подій за представником просторів для проведення подій;
- керування особистими даними;
- висловлення враження про відвідувача-орендаря за результатами оренди;
- формування переліку вражень за простором для проведення подій;
- формування переліку вражень за представником простору для проведення подій;
- демонстрування сторінки представника простору для проведення подій;

- демонстрування сторінки відвідувача-орендаря;
- формування переліку вражень за відвідувачем-орендарем;
- демонстрування повного складу замовлення оренди простору для проведення подій (рис. 2.3).

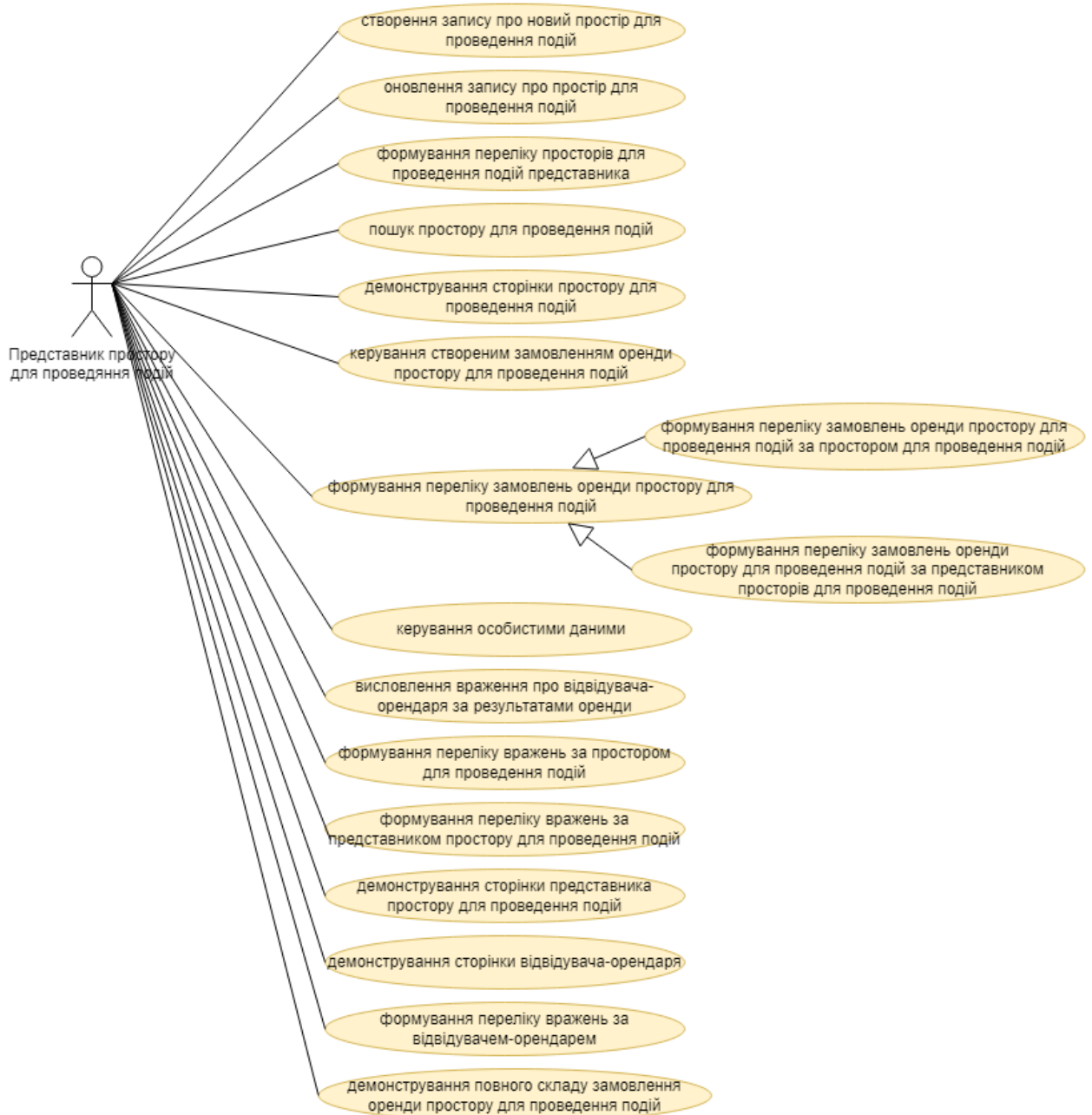


Рисунок 2.3 – Діаграма варіантів використання представником простору для проведення подій програми пошуку простору для проведення подій

2.3 Визначення структури бази даних

Накопичення і правильна структуризація даних, базуючись на сценаріях, які мають бути забезпечені програмно, є критичними для забезпечення функціонування програми, тому логічним є створення для цього бази даних (БД). БД підтримується за допомогою MySQL, на основі чого було створено відповідну схему (рис. 2.4).

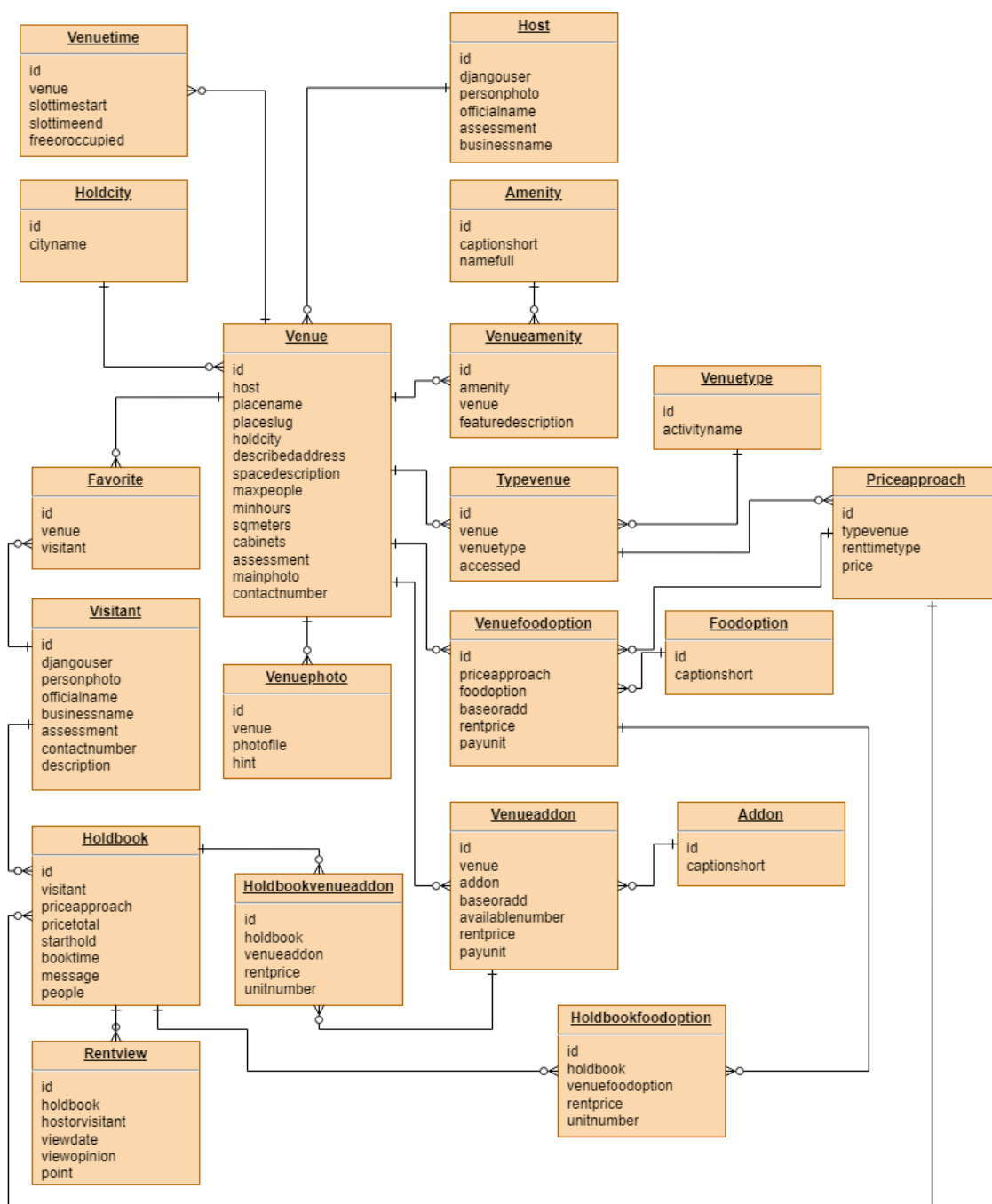


Рисунок 2.4 – Схема БД програми пошуку просторів для проведення подій

Таблиця Venue задає простір для проведення подій, структуруючи його за вказаним порядком:

- id – номер простору для проведення подій, що унікально його ідентифікує;
- host – представник простору для проведення подій;
- placename – назва, за якою цей простір для проведення подій має відобразитися;
- placeslug – унікальний ідентифікатор, сформований як текст для виконання звернень до простору для проведення подій у складі адреси вебсторінки;
- holdcity – місто, де знаходиться простір для проведення подій;
- describedaddress – адреса простору для проведення подій в тому вигляді, в якому вона відображається користувачам;
- spacedescription – представлення простору для проведення подій на основі його опису, особливостей оренди, проведення самих подій за цим простором в текстовому вигляді;
- maxpeople – найбільше число людей, які можуть відвідати цей простір для проведення подій одночасно;
- minhours – найменша кількість послідовних годин, на яку може бути орендовано цей простір для проведення подій;
- sqmeters – сумарна площа в квадратних метрах простору для проведення подій;
- cabinets – число окремих приміщень (кімнат, кабінетів) простору для проведення подій;
- assessment – бал, що всередньому всі відвідувачі-орендарі за результатами обслуговування встановили цьому простору для проведення подій;
- mainphoto – основне зображення, що представляє простір для проведення подій, до якого додатково можуть визначатися інші фотографії, але цим зображенням простір для проведення подій представляється завжди,

коли є можливість продемонструвати тільки одне зображення (наприклад, в результатах пошуку);

- `contactnumber` – номер, за яким телефоном можна зв'язатися з цим простором для проведення подій (може надаватися не один номер, а декілька, можуть додатково або окремо вказуватися дані посилань у соціальних мережах).

Таблиця Venue має зв'язок з таблицею БД Host через поле `host`, яке містить номер зовнішнього ключа. Так це задає представника простору для проведення подій. Простір для проведення подій може мати тільки одного представника в програмі, тобто множинне представлення не дозволяється. У свою чергу особа-представник може працювати з декількома просторами для проведення подій за наявності. Відповідно дане формулювання підпадає під встановлення зв'язку один-до-багатьох.

Таблиця Venue має зв'язок з таблицею Holdcity через поле `holdcity`, яке містить номер зовнішнього ключа. Так це задає місто розташування простору для проведення подій. Звісно, що простір може розташовуватися тільки в одному конкретно заданому місті. Проте це не заважає у самому місті мати одразу декілька просторів для проведення подій, що є логічним. Відповідно дане формулювання підпадає під встановлення зв'язку один-до-багатьох.

Простори для проведення подій можуть мати певні зручності, які можуть включати, наприклад, кондиціонування, спеціальну систему освітлення, вентиляцію. Оскільки такі зручності можуть бути досить різними, то необхідно мати можливість визначати в програмі конкретний перелік, який до того ж може в подальшому ще змінюватися далі. Тому потрібним є створення окремої таблиці, яка б такі зручності, які можуть у подальшому обиратися для конкретних просторів для проведення подій у відповідність, визначала. Цією таблицею є `Amenity`, що структурована за вказаним порядком:

- `id` – номер зручності, яка може застосовуватися для простору проведення подій, що унікально її ідентифікує;

- `captionshort` – підпис зручності для відображення;

– namefull – повна назва зручності, якщо така потрібна (повний варіант).

Конкретні наявні зручності в конкретних просторах подій визначаються таблицею Venueamenity, що структурована за вказаним порядком:

– id – номер запису про конкретну наявну зручність в конкретному просторі подій, що унікально його ідентифікує;

– amenity – зручність, яка присутня в цьому просторі для проведення подій;

– venue – простір для проведення подій, де така зручність доступна;

– featuredescription – за наявності фіксація тих особливостей, які стосуються наявності цієї зручності за цим простором для проведення подій.

Таблиця Venue має фактичний зв'язок з таблицею Amenity, що виражається типом багато-до-багатьох. Тому таблиця Venueamenity необхідна, щоб за умов реляційної системи керування базами даних його реалізувати. Таким чином визначається, що простір для проведення подій може мати не одну, а декілька різних зручностей, обраних з варіантів з таблиці Amenity, а кожна потенційна зручність призначена для багаторазового застосування для різних просторів для проведення подій.

Таблиця Venuephoto задає фотографії просторів для проведення подій, структуруючи їх за вказаним порядком:

– id – номер фотографії простору для проведення подій, що унікально її ідентифікує;

– venue – простір для проведення подій, який зображений на фотографії;

– photofile – фотографія, де показано простір для проведення подій, його приміщення, особливості;

– hint – підказка, якою може бути прокоментовано конкретно цю фотографію простору для проведення подій.

Таблиця Venuephoto має зв'язок з таблицею Venue через поле venue, яке містить номер зовнішнього ключа. Так це задає простір для проведення подій, що є на фотографії. Одночасно одне зображення може представляти тільки один простір для проведення подій. Але один простір для проведення подій може для інформування потенційних відвідувачів мати декілька зображень. Відповідно дане формулювання підпадає під встановлення зв'язку один-до-багатьох.

Міста, доступні для декларування розташування просторів для проведення подій, визначаються таблицею Holdcity, що структурована за вказаним порядком:

- id – номер міста, доступного для декларування розташування просторів для проведення подій, що унікально його ідентифікує;
- cityname – назва міста.

Типи подій, для проведення яких можуть використовуватися простори, визначаються таблицею Venuetype, що структурована за вказаним порядком:

- id – номер типу події, для проведення якої може використовуватися простір;
- activityname – назва типу події, для проведення якої може використовуватися простір.

Співвіднесення конкретних просторів з конкретними типами подій, для яких вони можуть використовуватись, визначається таблицею Typevenue, що структурована за вказаним порядком:

- id – номер запису про співвіднесення конкретного простору з конкретним типом події, що унікально його ідентифікує;
- venue – простір для проведення подій, в якому такий тип події може проводитися;
- venuetype – тип події, який може проводитися в цьому просторі;
- accessed – доступність або недоступність на поточну мить цього простору для проведення цього типу подій.

Таблиця Venue має фактичний зв'язок з таблицею Venuetype, що виражається типом багато-до-багатьох. Тому таблиця Typevenue необхідна, щоб за умов реляційної системи керування базами даних його реалізувати. Таким чином визначається, що простір для проведення подій може дозволяти проводити не один тип подій, а декілька різних, обраних з варіантів з таблиці Venuetype, а кожен тип подій призначений для багаторазового застосування для різних просторів для проведення подій.

Таблиця Host задає представника просторів для проведення подій, структуруючи його за вказаним порядком:

- id – номер представника просторів для проведення подій, що унікально його ідентифікує;
- personphoto – зображення представника просторів для проведення подій;
- officialname – ім'я представника просторів для проведення подій, як воно демонструється в програмі всім користувачам;
- assessment – бал, що всередньому всі відвідувачі-орендарі за результатами обслуговування встановили всім просторам для проведення подій цього представника;
- businessname – точне персональне ім'я представника простору для проведення подій у відповідності з документами, як воно відображається іншій стороні вже після того, як оренда оформлена;
- djangouser – обліковий запис Django, що відповідає представнику простору для проведення подій.

Простір для проведення подій окрім зручностей може мати додатковий інвентар: наприклад, мобільний проєктор, додатковий стіл тощо. Суть додаткового інвентаря в тому, що його потрібно окремо замовляти. Тобто для того щоб він був доступним, потрібно вказати це в замовленні, щоб були проведені потрібні підготовчі роботи. Якщо це не буде зроблено, то він не буде доступний в орендованому просторі для проведення подій. Такий інвентар може бути безкоштовним, а може надаватися за додаткову плату.

Таблиця `Addon` безпосередньо задає варіанти додаткового інвентаря, структуруючи його за вказаним порядком:

- `id` – номер додаткового інвентаря, що унікально його ідентифікує;
- `captionshort` – назва варіанта додаткового інвентаря.

Конкретний наявний додатковий інвентар в конкретному просторі подій визначається таблицею `Venueaddon`, що структурована за вказаним порядком:

- `id` – номер запису наявного додаткового інвентаря в конкретному просторі для проведення подій, що унікально його ідентифікує;
- `venue` – простір для проведення подій, де цей додатковий інвентар присутній;
- `addon` – додатковий інвентар, який можна використати за цим простором для проведення подій;
- `baseoradd` – потребує додаткової сплати у випадку замовлення в складі оренди цього простору для проведення подій або є безкоштовним (але потребує замовлення);
- `availablenumber` – доступна кількість екземплярів цього додаткового інвентаря;
- `rentprice` – вартість оренди екземпляра цього додаткового інвентаря;
- `payunit` – одиниця додаткового інвентаря, за яку відбувається сплата: поштучно або за годину, або за одиницю, визначену базовою ставкою оренди.

Таблиця `Venue` має фактичний зв'язок з таблицею `Addon`, що виражається типом багато-до-багатьох. Тому таблиця `Venueaddon` необхідна, щоб за умов реляційної системи керування базами даних його реалізувати. Таким чином визначається, що простір для проведення подій може мати декілька варіантів додаткового інвентаря, обраних з варіантів з таблиці `Addon`, а кожен додатковий варіант призначений для багаторазового застосування для різних просторів для проведення подій.

Таблиця `Visitant` задає відвідувача-орендаря, структуруючи його за вказаним порядком:

- id – номер відвідувача-орендаря, що унікально його ідентифікує;
- personphoto – зображення користувача, який є відвідувачем-орендарем;
- officialname – ім'я користувача, який є відвідувачем-орендарем;
- businessname – точне персональне ім'я користувача, який є відвідувачем-орендарем, як воно відображається іншій стороні у замовленні оренди;
- assessment – бал, що всередньому всі представники просторів для проведення подій виставили цьому відвідувачу-орендарю за результатами обслуговування;
- description – власне представлення відвідувачем-орендарем себе;
- .djangouser – обліковий запис користувача Django, який використовує відвідувач-орендар;
- contactnumber – контактний номер відвідувача-орендаря.

Таблиця Rentview задає враження відвідувача-орендаря про простір для проведення подій або представника просторів для проведення подій про відвідувача-орендаря, структуруючи його за вказаним порядком:

- id – номер враження відвідувача-орендаря про простір для проведення подій або представника просторів для проведення подій про відвідувача-орендаря, що унікально його ідентифікує;
- holdbook – замовлення оренди простору для проведення подій, за яким було сформовано дане враження;
- hostorvisitant – автором заяви є сторона представника простору для проведення подій або відвідувача-орендаря;
- viewdate – час фіксування в програмі враження відвідувача-орендаря про простір для проведення подій або представника просторів для проведення подій про відвідувача-орендаря;
- vieworinion – представлення враження відвідувача-орендаря про простір для проведення подій або представника просторів для проведення подій про відвідувача-орендаря у вигляді тексту;

– point – бал, у який оцінив власне враження відвідувач-орендар про простір для проведення подій або представник просторів для проведення подій про відвідувача-орендаря.

Таблиця Rentview має зв'язок з таблицею Holdbook через поле holdbook, яке містить номер зовнішнього ключа. Так це задає замовлення оренди простору для проведення подій, за яким сформував власне враження відвідувач-орендар або представник простору для проведення подій. Оскільки кожна зі сторін може сформулювати своє враження, то з кожним замовленням оренди простору для проведення подій може бути пов'язано нуль, один або два враження, але враження стосується тільки одного замовлення оренди простору для проведення подій. Відповідно дане формулювання підпадає під встановлення зв'язку один-до-багатьох.

Часовий інтервал роботи простору для проведення подій визначається таблицею Venuetime, структуруючи її за вказаним порядком:

- id – номер запису визначення часового інтервалу роботи простору для проведення подій, що унікально його ідентифікує;
- venue – простір для проведення подій, що має таким чином встановлений часовий інтервал роботи;
- slottimestart – час і дата, коли бере початок цей часовий інтервал роботи;
- slottimeend – час і дата, коли закінчується цей часовий інтервал роботи;
- freeoroccupied – зазначений часовий інтервал є вільним чи зайнятим.

Таблиця Venuetime має зв'язок з таблицею Venue через поле venue, яке містить номер зовнішнього ключа. Так це задає простір для проведення подій, за яким визначено часовий інтервал роботи. Одночасно один часовий інтервал стосується тільки одного простору для проведення подій, а один простір для проведення подій може охоплювати декілька часових інтервалів. Відповідно дане формулювання підпадає під встановлення зв'язку один-до-багатьох.

Базова ставка (вартість) оренди простору для проведення подій визначається таблицею `Priceapproach`, структуруючи її за вказаним порядком:

- `id` – номер базової ставки оренди простору для проведення подій, що унікально її ідентифікує;
- `typevenue` – конкретний тип події, який проводиться в конкретному просторі і на який визначена ця базова ставка оренди простору для проведення подій;
- `renttimetype` – тип оренди простору для проведення подій: весь день, півдня, погодинно;
- `price` – сума, яку необхідно за таких умов сплатити за оренду.

Таблиця `Priceapproach` має зв'язок з таблицею `Typevenue` через поле `typevenue`, яке містить номер зовнішнього ключа. Так це задає конкретний тип події, визначений в межах конкретного простору, для якого задано базову ставку оренди. З іншого боку кожен тип події, визначений в межах конкретного простору, може мати різні базові ставки оренди. Відповідно дане формулювання підпадає під встановлення зв'язку один-до-багатьох.

Таблиця `Holdbook` задає оренди просторів для проведення подій, структуруючи їх за вказаним порядком:

- `id` – номер оренди простору для проведення подій, що унікально його ідентифікує;
- `visitant` – відвідувач-орендар, який створив замовлення на оренду простору для проведення подій;
- `priceapproach` – базова ставка (вартість) оренди простору для проведення подій, за якою було обчислено основну вартість замовлення (через неї також визначається і конкретний тип події у просторі, а вже за цим відповідно і сам простір для проведення подій, який орендовано: тобто оренда відбувається за обраним типом оренди за часом конкретного простору на конкретний тип події);
- `pricetotal` – підсумкова загальна вартість всього сформованого замовлення оренди простору для прокату речей;

– starthold – час і дата, коли розпочинається сама оренда простору для проведення подій;

– booktime – час і дата, коли у програмі було створено замовлення оренди простору для проведення подій;

– message – повідомлення, яке може бути визначено відвідувачем-орендарем щодо особливостей виконання оренди (контактна інформація, особливості проведення);

– people – кількість людей, яка буде присутня в просторі для проведення подій під час оренди.

Таблиця Holdbook має зв'язок з таблицею Visitant через поле visitant, яке містить номер зовнішнього ключа. Так це задає конкретного відвідувача-орендаря, який створив це замовлення оренди простору для проведення подій. З іншого боку кожен відвідувач-орендар може замовляти різні оренди просторів для проведення подій. Відповідно дане формулювання підпадає під встановлення зв'язку один-до-багатьох.

Таблиця Holdbook має зв'язок з таблицею Priceapproach через поле priceapproach, яке містить номер зовнішнього ключа. Так це задає базову ставку оренди простору для проведення подій, за якою було створено замовлення оренди простору. З іншого боку кожна ставка оренди простору для проведення подій може застосовуватися багаторазово в різних замовленнях оренди простору. Відповідно дане формулювання підпадає під встановлення зв'язку один-до-багатьох.

Таблиця Holdbookvenueaddon задає додатковий інвентар, який необхідний під час виконання оренди цього простору для проведення подій, структуруючи його за вказаним порядком:

– id – номер включення додаткового інвентаря до оренди простору для проведення подій, що унікально його ідентифікує;

– holdbook – замовлення оренди простору для проведення подій, до якого відноситься цей додатковий інвентар;

- venueaddon – номер запису наявного додаткового інвентаря в конкретному просторі для проведення подій, який необхідний для цієї оренди;
- rentprice – вартість оренди заданої кількості екземплярів цього додаткового інвентаря;
- unitnumber – кількість одиниць додаткового інвентаря, яка орендується.

Таблиця Holdbook має фактичний зв'язок з таблицею Venueaddon, що виражається типом багато-до-багатьох. Тому таблиця Holdbookvenueaddon необхідна, щоб за умов реляційної системи керування базами даних його реалізувати. Таким чином визначається, що замовлення оренди простору для проведення подій може мати декілька варіантів додаткового інвентаря, доступних за конкретним простором для проведення подій, обраних з варіантів з таблиці Venueaddon, а кожен додатковий інвентар конкретного простору для проведення подій призначений для багаторазового застосування для різних замовлень оренди простору для проведення подій.

Відзначення простору для проведення подій як улюбленого задається таблицею Favorite, структуруючи її за вказаним порядком:

- id – номер відзначення простору для проведення подій як улюбленого, що унікально його ідентифікує;
- venue – простір для проведення подій, який було відзначено як улюблений;
- visitant – відвідувач-орендар, який відзначив простір для проведення подій як улюблений.

Таблиця Favorite має зв'язок з таблицею Venue через поле venue, яке містить номер зовнішнього ключа. Так це задає простір для проведення подій, який було відзначено як улюблений. З іншого боку кожен простір для проведення подій може багаторазово бути відзначеним як улюблений. Відповідно дане формулювання підпадає під встановлення зв'язку один-до-багатьох.

Таблиця Favorite має зв'язок з таблицею Visitant через поле visitant, яке містить номер зовнішнього ключа. Так це задає відвідувача-орендаря, який відзначив пеній простір як улюблений. З іншого боку кожен простір для проведення подій може багаторазово бути відзначеним різними відвідувачами-орендарями як улюблений. Відповідно дане формулювання підпадає під встановлення зв'язку один-до-багатьох.

Окрім додаткового інвентаря під час проведення подій у просторах може бути важливим харчування. Якщо подія відбувається протягом всього дня, то це питання стає нагальним. Таблиця Foodoption визначає можливі варіанти харчування, які можуть бути використані стосовно просторів для проведення подій, структуруючи їх за вказаним порядком:

- id – номер варіанта харчування, що унікально його ідентифікує;
- captionshort – назва варіанта харчування;
- descriptionfull – повний опис варіанта харчування (що саме він включає).

Те, які саме варіанти харчування доступні за просторами для проведення подій, визначається таблицею Venuefoodoption, структуруючи її за вказаним порядком:

- id – номер варіанта харчування, доступного за певним простором для проведення подій, що унікально його ідентифікує;
- priceapproach – базова ставка (вартість) оренди простору для проведення подій, за якою може бути додано цей варіант харчування;
- foodoption – варіант харчування, який доступний за цим простором для проведення подій;
- baseoradd – цей варіант є базовим і використовується за замовчуванням або він вимагає оплати;
- rentprice – вартість оплати харчування за вказану кількість;
- paunit – кількість порцій, на яку поширюється така вартість.

Таблиця Venuefoodoption має зв'язок з таблицею Priceapproach через поле priceapproach, яке містить номер зовнішнього ключа. Так це задає базову

ставку оренди простору для проведення подій, за якою може бути додано цей варіант харчування. З іншого боку кожна ставка може бути застосована до різних варіантів харчування, доступних за просторами для проведення подій. Відповідно дане формулювання підпадає під встановлення зв'язку один-до-багатьох.

Таблиця Venuefoodoption має зв'язок з таблицею Foodoption через поле foodoption, яке містить номер зовнішнього ключа. Так це задає варіант харчування, який доступний за цим простором для проведення подій за цією ставкою. З іншого боку кожен варіант харчування може бути доступним за різними ставками для різних просторів для проведення подій. Відповідно дане формулювання підпадає під встановлення зв'язку один-до-багатьох.

Те, які варіанти харчування простору мають бути застосовані під час оренди простору для проведення подій, визначається таблицею Holdbookfoodoption, структуруючи її за вказаним порядком:

- id – номер включення варіанта харчування до оренди простору для проведення подій, що унікально його ідентифікує;
- holdbook – замовлення оренди простору для проведення подій, за яким застосовано цей варіант харчування;
- venuefoodoption – варіант харчування, який замовлено під час оренди простору для проведення подій;
- rentprice – вартість замовлення харчування в заданій кількості одиниць (сумарна);
- unitnumber – кількість одиниць базових порцій, яка необхідна для харчування (кількість одиниць поля paunit таблиці Venuefoodoption).

Таблиця Holdbookfoodoption має зв'язок з таблицею Holdbook через поле holdbook, яке містить номер зовнішнього ключа. Так це задає замовлення оренди простору для проведення подій, за яким застосовано цей варіант харчування. З іншого боку кожне замовлення оренди простору для проведення подій може включати декілька варіантів харчування. Відповідно дане формулювання підпадає під встановлення зв'язку один-до-багатьох.

Таблиця Holdbookfoodoption має зв'язок з таблицею Venuefoodoption через поле venuefoodoption, яке містить номер зовнішнього ключа. Так це задає варіант харчування, який замовлено під час оренди простору для проведення подій. З іншого боку кожен варіант харчування може входити в різні замовлення оренди просторів для проведення подій. Відповідно дане формулювання підпадає під встановлення зв'язку один-до-багатьох.

2.4 Висновки за розділом 2

Виконано завдання спроектувати програмне забезпечення пошуку простору для проведення подій, включаючи кроки з вибору засобів, які підтримували проєктування і розробку (Python, Django, MySQL), сценарії роботи, структуру БД. БД підтримується за допомогою MySQL, на основі чого було створено відповідну схему БД та описано структуру.

3 РОЗРОБКА ПРОГРАМИ

3.1 Визначення послідовності роботи з програмою

До програми забезпечується універсальний варіант доступу, який в залежності від параметрів доступу, визначених обліковим записом користувача, дій користувача направляється відповідним чином за схемою функціонування програми (рис. 3.1).

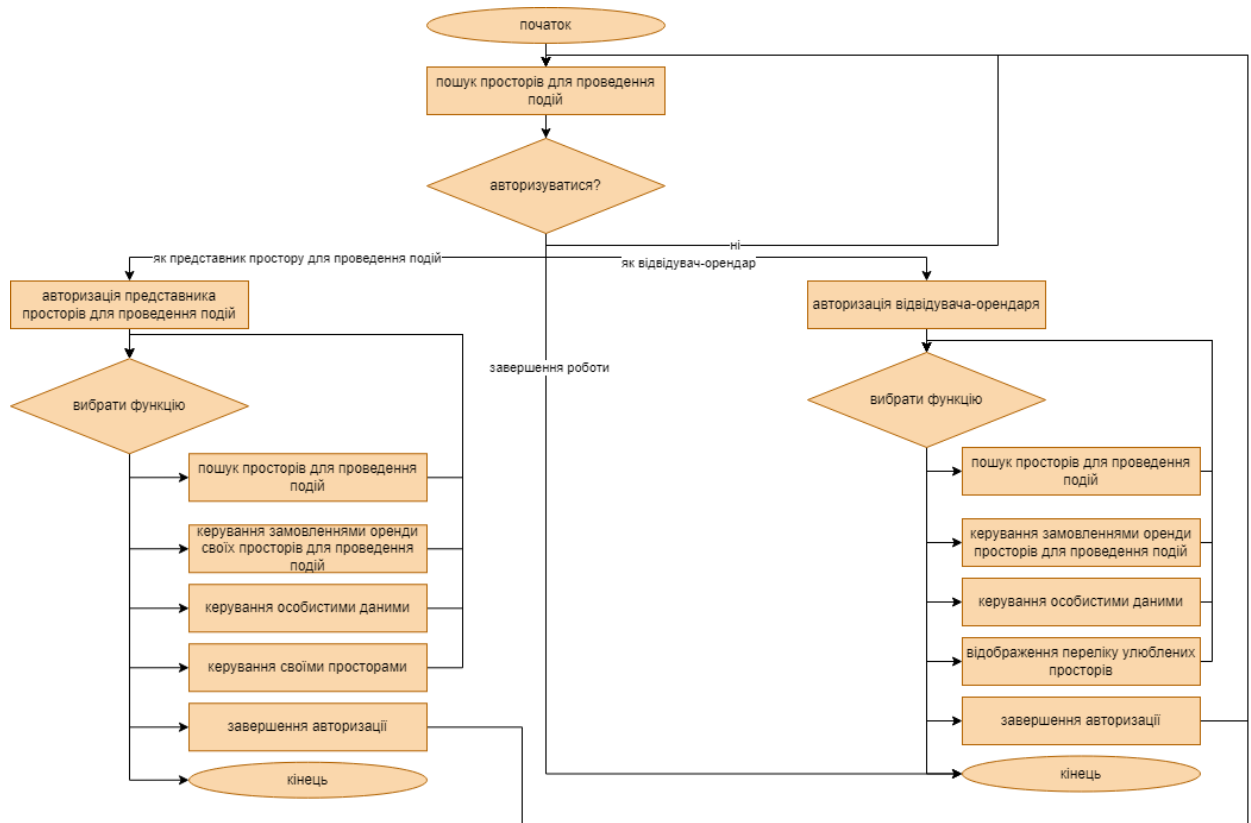


Рисунок 3.1 – Схема функціонування програми пошуку просторів для проведення подій

3.2 Особливості програмної реалізації

Створений додаток організований за принципом багатoshарової архітектури, що забезпечена шаблоном проєктування модель-представлення-шаблон (МПШ), тому нижче описано те, як шари цієї архітектури у відповідності з зазначеним шаблоном реалізовані.

Першими описані моделі як одна з базових одиниць МПШ.

Модель Venue задає простір для проведення подій, структуруючи його за вказаним порядком з полів:

- host є полем представника простору для проведення подій, тип ForeignKey;
- placename є полем назви, за якою цей простір для проведення подій має відображатися, тип CharField;
- placeslug є полем унікального ідентифікатора, сформованого як текст для виконання звернень до простору для проведення подій у складі адреси вебсторінки, тип SlugField;
- holdcity є полем міста, де знаходиться простір для проведення подій, тип ForeignKey;
- describedaddress є полем адреси простору для проведення подій в тому вигляді, в якому вона відображається користувачам, тип CharField;
- spacedescription є полем представлення простору для проведення подій на основі його опису, особливостей оренди, проведення самих подій за цим простором в текстовому вигляді, тип TextField;
- maxpeople є полем найбільшого числа людей, які можуть відвідати цей простір для проведення подій одночасно, тип PositiveSmallIntegerField;
- minhours є полем найменшої кількості послідовних годин, на яку може бути орендовано цей простір для проведення подій, тип PositiveSmallIntegerField;
- sqmeters є полем сумарної площі в квадратних метрах простору для проведення подій, тип FloatField;
- cabinets є полем числа окремих приміщень простору для проведення подій, тип PositiveSmallIntegerField;
- assessment є полем балу, що всередньому всі відвідувачі-орендарі за результатами обслуговування встановили цьому простору для проведення подій, тип FloatField;

- `mainphoto` є полем основного зображення, що представляє простір для проведення подій, до якого додатково можуть визначатися інші фотографії, але цим зображенням простір для проведення подій представляється завжди, коли є можливість продемонструвати тільки одне зображення, тип `ImageField`;

- `contactnumber` є полем номера, за яким телефоном можна зв'язатися з цим простором для проведення подій, тип `CharField`.

Зручності, які можуть застосовуватися для простору проведення подій, реалізовані моделлю `Amenity`, що структурована за вказаним порядком з полів:

- `captionshort` є полем підпису зручності для відображення, тип `CharField`;

- `namefull` є полем повної назви зручності, якщо така потрібна, тип `CharField`.

Конкретні наявні зручності в конкретних просторах подій визначаються моделлю `Venueamenity`, що структурована за вказаним порядком з полів:

- `amenity` є полем зручності, яка присутня в цьому просторі для проведення подій, тип `ForeignKey`;

- `venue` є полем простору для проведення подій, де така зручність доступна, тип `ForeignKey`;

- `featuredescription` є полем фіксації тих особливостей, які стосуються наявності цієї зручності за цим простором для проведення подій, тип `TextField`.

Модель `Venuephoto` реалізує роботу з фотографіями просторів для проведення подій, структуруючи їх за вказаним порядком з полів:

- `venue` є полем простору для проведення подій, який зображений на фотографії, тип `ForeignKey`;

- `photofile` є полем фотографії, де показано простір для проведення подій, його приміщення, особливості, тип `ImageField`;

- `hint` є полем підказки, якою може бути прокоментовано конкретно цю фотографію простору для проведення подій, тип `CharField`.

Міста, доступні для декларування розташування просторів для проведення подій, реалізовані моделлю Holdcity, що має тільки поле cityname з назвою міста, тип CharField.

Типи подій, для проведення яких можуть використовуватися простори, реалізовані моделлю Venuetype, що має тільки поле activityname з назвою типу події, для проведення якої може використовуватися простір, тип CharField.

Співвіднесення конкретних просторів з конкретними типами подій, для яких вони можуть використовуватись, реалізовано моделлю Typevenue, що структурована за вказаним порядком з полів:

- venue є полем простору для проведення подій, в якому такий тип події може проводитися, тип ForeignKey;

- venuetype є полем типу події, який може проводитися в цьому просторі, тип ForeignKey;

- accessed є полем доступності або недоступності на поточну мить цього простору для проведення цього типу подій, тип BooleanField.

Модель Host задає представника просторів для проведення подій, структуруючи його за вказаним порядком з полів:

- personphoto є полем зображення представника просторів для проведення подій, тип ImageField;

- officialname є полем імені представника просторів для проведення подій, як воно демонструється в програмі всім користувачам, тип CharField;

- assessment є полем балу, що всередньому всі відвідувачі-орендарі за результатами обслуговування встановили всім просторам для проведення подій цього представника, тип FloatField;

- businessname є полем точного персонального імені представника простору для проведення подій у відповідності з документами, як воно відображається іншій стороні вже після того, як оренда оформлена, тип CharField.

Модель Addon безпосередньо реалізує варіанти додаткового інвентаря з полем captionshort, що є назвою варіанта додаткового інвентаря.

Конкретний наявний додатковий інвентар в конкретному просторі подій реалізований моделлю Venueaddon, що структурована за вказаним порядком з полів:

- venue є полем простору для проведення подій, де цей додатковий інвентар присутній, тип ForeignKey;
- addon є полем додаткового інвентаря, який можна використати за цим простором для проведення подій, тип ForeignKey;
- baseoradd є полем визначення потреби в додатковій сплаті у випадку замовлення в складі оренди цього простору для проведення подій або безкоштовності, тип BooleanField;
- availablenumber є полем доступної кількості екземплярів цього додаткового інвентаря, тип PositiveSmallIntegerField;
- rentprice є полем вартості оренди екземпляра цього додаткового інвентаря, тип FloatField;
- payunit є полем одиниці додаткового інвентаря, за яку відбувається сплата: поштучно або за годину, або за одиницю, визначену базовою ставкою оренди, тип PositiveSmallIntegerField.

Модель Visitant реалізує відвідувача-орендаря, структуруючи його за вказаним порядком з полів:

- personphoto є полем зображення користувача, який є відвідувачем-орендарем, тип ImageField;
- officialname є полем імені користувача, який є відвідувачем-орендарем, тип CharField;
- businessname є полем точного персонального імені користувача, який є відвідувачем-орендарем, як воно відображається іншій стороні у замовленні оренди, тип CharField;
- assessment є полем балу, що всередньому всі представники просторів для проведення подій виставили цьому відвідувачу-орендарю за результатами обслуговування, тип FloatField;

- description є полем власного представлення відвідувачем-орендарем себе, тип CharField;

- contactnumber є полем контактного номера відвідувача-орендаря, тип CharField.

- favorites є полем відзначених просторів для проведення подій як улюблених цим користувачем, що є відвідувачем-орендарем.

Модель Rentview реалізує враження відвідувача-орендаря про простір для проведення подій або представника просторів для проведення подій про відвідувача-орендаря, структуруючи його за вказаним порядком з полів:

- holdbook є полем замовлення оренди простору для проведення подій, за яким було сформовано дане враження, тип ForeignKey;

- hostorvisitant є полем автора заяви є сторона представника простору для проведення подій або відвідувача-орендаря, тип ForeignKey;

- viewdate є полем часу фіксування в програмі враження відвідувача-орендаря про простір для проведення подій або представника просторів для проведення подій про відвідувача-орендаря, тип DateTimeField;

- vieworpinion є полем представлення враження відвідувача-орендаря про простір для проведення подій або представника просторів для проведення подій про відвідувача-орендаря у вигляді тексту, тип CharField;

- point є полем балу, у який оцінив власне враження відвідувач-орендар про простір для проведення подій або представник просторів для проведення подій про відвідувача-орендаря, тип PositiveSmallIntegerField.

Часовий інтервал роботи простору для проведення подій реалізовано моделлю Venuetime, структуруючи її за вказаним порядком з полів:

- venue є полем простору для проведення подій, що має таким чином встановлений часовий інтервал роботи, тип ForeignKey;

- slottimestart є полем часу і дати, коли бере початок цей часовий інтервал роботи, тип DateTimeField;

- slottimeend є полем часу і дати, коли закінчується цей часовий інтервал роботи, тип DateTimeField;

– freeoccupied є полем визначення зазначеного часового інтервалу як вільного чи зайнятого, тип BooleanField.

Базова ставка (вартість) оренди простору для проведення подій реалізована моделлю Priceapproach, структуруючи її за вказаним порядком з полів:

– typevenue є полем конкретного типу події, який проводиться в конкретному просторі і на який визначена ця базова ставка оренди простору для проведення подій, тип ForeignKey;

– renttimetype є полем типу оренди простору для проведення подій: весь день, півдня, погодинно, тип PositiveSmallIntegerField;

– price є полем суми, яку необхідно за таких умов сплатити за оренду, тип FloatField.

Модель Holdbook задає оренди просторів для проведення подій, структуруючи їх за вказаним порядком з полів:

– visitant є полем відвідувача-орендаря, який створив замовлення на оренду простору для проведення подій, тип ForeignKey;

– priceapproach є полем базової ставки (вартості) оренди простору для проведення подій, за якою було обчислено основну вартість замовлення, тип ForeignKey;

– pricetotal є полем підсумкової загальної вартості всього сформованого замовлення оренди простору для прокату речей, тип FloatField;

– starthold є полем часу і дати, коли розпочинається сама оренда простору для проведення подій, тип DateTimeField;

– booktime є полем часу і дати, коли у програмі було створено замовлення оренди простору для проведення подій, тип DateTimeField;

– message є полем повідомлення, яке може бути визначено відвідувачем-орендарем щодо особливостей виконання оренди, тип CharField;

– people є полем кількості людей, яка буде присутня в просторі для проведення подій під час оренди, тип PositiveSmallIntegerField.

Модель `Holdbookvenueaddon` реалізує додатковий інвентар, який необхідний під час виконання оренди цього простору для проведення подій, структуруючи його за вказаним порядком з полів:

- `holdbook` є полем замовлення оренди простору для проведення подій, до якого відноситься цей додатковий інвентар, тип `ForeignKey`;
- `venueaddon` є полем запису наявного додаткового інвентаря в конкретному просторі для проведення подій, який необхідний для цієї оренди, тип `ForeignKey`;
- `rentprice` є полем вартості оренди заданої кількості екземплярів цього додаткового інвентаря, тип `FloatField`;
- `unitnumber` є полем кількості одиниць додаткового інвентаря, яка орендується, тип `PositiveSmallIntegerField`.

Модель `Foodoption` реалізує можливі варіанти харчування, які можуть бути використані стосовно просторів для проведення подій, структуруючи їх за вказаним порядком з полів:

- `captionshort` є полем назви варіанта харчування, тип `CharField`;
- `descriptionfull` є полем повного опису варіанта харчування, тип `CharField`.

Те, які саме варіанти харчування доступні за просторами для проведення подій, реалізовано моделлю `Venuefoodoption`, структуруючи її за вказаним порядком з полів:

- `priceapproach` є полем базової ставки оренди простору для проведення подій, за якою може бути додано цей варіант харчування, тип `ForeignKey`;
- `foodoption` є полем варіанта харчування, який доступний за цим простором для проведення подій, тип `ForeignKey`;
- `baseoradd` є визначенням того, цей варіант є базовим і використовується за замовчуванням або він вимагає оплати, тип `BooleanField`;
- `rentprice` є полем вартості оплати харчування за вказану кількість, тип `FloatField`;

– `payunit` є полем кількості порцій, на яку поширюється така вартість, тип `PositiveSmallIntegerField`.

Те, які варіанти харчування простору мають бути застосовані під час оренди простору для проведення подій, реалізується моделлю `Holdbookfoodoption`, структуруючи її за вказаним порядком з полів:

- `holdbook` є полем замовлення оренди простору для проведення подій, за яким застосовано цей варіант харчування, тип `ForeignKey`;
- `venuefoodoption` є полем варіанта харчування, який замовлено під час оренди простору для проведення подій, тип `ForeignKey`;
- `rentprice` є полем вартості замовлення харчування в заданій кількості одиниць (сумарна), тип `FloatField`;
- `unitnumber` є полем кількості одиниць базових порцій, яка необхідна для харчування, тип `PositiveSmallIntegerField`.

Далі описані представлення МПШ. Вони забезпечують підсумкове формування відповідних шаблонів з визначенням вебсторінок на виході. Розмітка шаблонів забезпечена безкоштовним шаблоном [9].

Демонстрування сторінки простору для проведення подій було реалізовано представленням `VenueView`. Параметр `venueslug` видає текстовий ідентифікатор простору для проведення подій за його назвою. За цим ідентифікатором реалізується виділення даних цього простору, включаючи зручності, фотографії, можливі типи подій, можливий додатковий інвентар, варіанти харчування, 5 останніх вражень відвідувачів-орендарів про простір, часові інтервали, коли простір доступний та зайнятий для формування календаря (тільки для майбутнього часу), доступні цінові пропозиції. Якщо користувач авторизований, то також визначається, чи він відніс цей простір для проведення подій до улюбленого.

Формування переліку замовлень оренди простору для проведення подій, створених відвідувачем-орендарем, реалізовано представленням `VisitantHoldbooksView` з параметром `display`, який визначає те, які саме

замовлення, в якому саме порядку використовуються для формування переліку:

- 0 – ті, які ще не розпочались, впорядковуючи за часом початку від поточного;

- 1 – ті, які ще не розпочались, впорядковуючи за часом початку від останнього;

- 2 – ті, які вже відбулись, впорядковуючи за часом початку від останнього;

- 3 – ті, які вже відбулись, впорядковуючи за часом початку від найдавнішого;

- 4 – всі, впорядковуючи за часом замовлення від найдавнішого;

- 5 – всі, впорядковуючи за часом замовлення від останнього.

Самі ж замовлення оренди просторів для проведення подій відбираються такі, які були створені як відвідувачем-орендарем тим, який авторизувався.

Формування переліку замовлень оренди простору для проведення подій за представником просторів реалізовано представленням `HostHoldbooksView`. Параметр `display` працює аналогічно попередньому варіанту. Самі замовлення оренди просторів для проведення подій відбираються такі, які були створені стосовно просторів для проведення подій. Які представляє користувач, який авторизувався.

Формування переліку замовлень оренди за простором для проведення подій реалізовано представленням `VenueHoldbooksView`. Ця функція має параметр `display`, описаний вище, та параметр `venueslug`, що задає текстовий ідентифікатор простору для проведення подій. Відбираються тільки ті замовлення, що стосуються заданого простору для проведення подій. Надається перелік тільки тоді, коли користувач є представником цього простору.

Демонстрування складу замовлення оренди простору для проведення подій реалізовано функцією `HoldbookView`, яка параметром `holdbookid`

визначає номер самого замовлення оренди простору для проведення подій. Тільки якщо користувач є відвідувачем-орендарем простору для проведення подій, зазначеного в замовленні, якщо він є представником цього простору, то йому демонструється повний зміст замовлення. До змісту замовлення входить і замовлене харчування, додатковий інвентар.

Створення запису про новий простір для проведення подій реалізовано представленням `VenueNewView`. Оновлення запису про простір для проведення подій реалізовано представленням `VenueRenewView`, а параметр `venueslug` визначає текстовий ідентифікатор простору для проведення подій.

Формування переліку вражень за простором для проведення подій реалізовано представленням `VenueRentviewsView`. Параметр `venueslug` задає текстовий ідентифікатор простору для проведення подій. Відбираються ті враження, що сформовані за замовленнями стосовно цього простору для проведення подій.

Формування переліку вражень за представником простору для проведення подій реалізовано представленням `HostRentviewsView`. Параметр `hostname` задає логін представника просторів для проведення подій, за яким потрібно визначити враження. Виділяються всі враження, які стосуються просторів для проведення подій, представником яких є цей користувач.

Формування переліку вражень за відвідувачем-орендарем реалізовано представленням `VisitantRentviewsView`. Параметр `visitantname` задає логін відвідувача-орендаря, а `mode` – те, які саме враження треба виділити. Якщо значення `mode` нульове, то це враження, які створені цим відвідувачем-орендарем, а якщо 1, то це враження, які створені про цього відвідувача-орендаря.

Формування переліку просторів для проведення подій представника реалізовано представленням `HostVenuesView` з параметром `hostname`, що задає логін представника просторів для проведення подій.

Формування переліку просторів для проведення подій, відзначених як улюблених, реалізовано представленням `FavoriteVenuesView`. Простори

визначаються за користувачем, який є відвідувачем-орендарем та авторизувався відповідним чином.

Демонстрування сторінки представника простору для проведення подій реалізовано представленням `HostProfileView` з параметром `hostname`, що визначає логін такого представника. На цю сторінку в шаблон вибираються дані, які стосуються вражень про простори даного представника, вражень, які створені цим представником, але тільки 5 останніх та про його простори.

Замовлення оренди простору для проведення подій реалізовано представленням `HoldbookView`. Параметри задаються через запит.

Демонстрування сторінки відвідувача-орендаря реалізовано представленням `VisitantProfileView`, параметр `visitantname` якого визначає логін відвідувача-орендаря. За цією сторінкою вибираються дані самого відвідувача-орендаря, 5 останніх вражень, які були визначені цим відвідувачем-орендарем, які стосувались цього відвідувача-орендаря.

Пошук простору для проведення подій реалізовано представленням `VenuesSearchView`. Для підтримки роботи відповідної форми представлення заповнює сторінку даними про:

- перелік міст, де простори можуть знаходитись;
- перелік зручностей, які можуть просторами надаватися;
- перелік типів подій, що можна обирати;
- перелік додаткового інвентаря;
- перелік варіантів харчування.

Керування особистими даними відвідувачем-орендарем реалізовано представленням `VisitantChangeView`, а представником простору для проведення подій – `HostChangeView`. Відповідно за даними авторизації перевіряються права доступу.

Для пошуку задаються параметри, які включають:

- місто, де має знаходитися простір для проведення подій;
- кількість людей, на яку буде відбуватися оренда;
- кількість годин, на яку буде відбуватися оренда;

- потрібна мінімальна та максимальна площі простору для проведення подій;
- мінімальна кількість приміщень у просторі для проведення подій;
- мінімальна та максимальна оцінки простору для проведення подій;
- перелік номерів зручностей, які мають бути доступні за простором для проведення подій;
- перелік номерів варіант харчування, які мають бути доступні за простором для проведення подій;
- перелік номерів додаткового інвентаря, який має бути доступним за простором для проведення подій;
- номер типу події, яка має проводитися в просторі;
- дата і час початку оренди простору для проведення подій;
- максимальна вартість оренди простору для проведення подій.

Висловлення враження відвідувачем-орендарем про простір для проведення подій за результатами оренди та висловлення враження представником простору для проведення подій про відвідувача-орендаря за результатами оренди реалізовано представленням `RentviewPublishView`. Параметр `holdbookid` задає номер замовлення оренди простору для проведення подій. Висловлення враження можливо тільки, якщо користувач авторизувався як відвідувач-орендар або представник простору для проведення подій, які стосуються заданого номера замовлення оренди. Якщо це не так, то доступ до висловлення враження блокується.

Відзначення простору для проведення подій як улюбленого та видалення такого відзначення реалізовано представленням `VenueFavoriteView`. Якщо користувач є відвідувачем-орендарем і не відзначав як улюблений цей простір для проведення подій, то він визначається як улюблений. Якщо це не так, але користувач є відвідувачем-орендарем, то простір вилучається з улюблених.

Керування створеним замовленням оренди простору для проведення подій реалізовано представленням `HoldbookSubmitView`. Параметр `holdbookid`

задає номер замовлення, а state – потрібний стан. Відбувається перевірка поточного стану, можливість переходу та наявність прав для відповідних дій над цим замовленням.

3.3 Висновки за розділом 3

До програми забезпечується універсальний варіант доступу, який в залежності від параметрів доступу, визначених обліковим записом користувача, дій користувача направляється відповідним чином за схемою функціонування програми. Реалізовані одиниці програми презентовані відповідними описами за створеним програмним кодом.

4 ЕКСПЛУАТАЦІЯ ТА ТЕСТУВАННЯ ПРОГРАМИ

4.1 Призначення програми

Програма має призначене використання у вигляді забезпечення потреб представників просторів для проведення подій та відвідувачів-орендарів. Представники просторів для проведення подій мають можливість використовувати її для визначення власних просторів для проведення подій, керування даними про них та доступність для оренди, створеними відвідувачами-орендарями замовленнями оренди просторів для проведення подій. Відвідувачі-орендарі мають можливість шукати такий простір, який дозволить задовольнити їхні вимоги щодо проведення події, замовляти оренду такого простору.

4.2 Умови виконання програми

Врахування потужності щодо запитів, які мають забезпечуватися для роботи користувачів з програмою, стосовно кількості на секунду, є важливим. За варіанта доступу з 50 запитами на секунду мінімально технічна сторона повинна включати: процесор не менше Intel Xeon E5440 з частотою 2.83 ГГц, оперативна пам'ять – 12 Гб, жорсткий диск – 40 Гб простору.

Сервер повинен забезпечувати не тільки параметри технічної складової, але і можливість роботи з інтерпретатором Python версії 3, засобами створення вебдодатків на основі Django, доступ до системи керування базами даних MySQL.

4.3 Експлуатація програми

Експлуатація програми розпочинається з гостя, якщо раніше користувач не використовував програму, тобто не звертався за адресою додатку на вебсервері. Якщо це так, то користувачу доступна єдина базова

сторінка з пошуку просторів для проведення подій, яка також демонструє і результати, а з якої можна переміщатися до сторінок самих просторів, обраних з пошуку.

Відвідувач-орендар повинен вже авторизуватися перед початком роботи з програмою. Після цього його меню окремо включає перехід до пошуку за допомогою лупи, але також і розділи замовлень, улюблених просторів для проведення подій та особистої інформації (рис. 4.1).



Рисунок 4.1 – Верхня частина сторінки простору для проведення подій

На сторінці простору для проведення подій згори відображається набір зображень простору, між якими можна перемикатися стрілками, та кнопки для переходу до відповідної частини сторінки. Сторінка також вміщує опис простору текстом (рис. 4.2), перелік всіх зручностей, які наявні, загальних параметрів, кнопку для визначення простору улюбленим, перелік доступного додаткового інвентаря, перелік доступних варіантів харчування, календар для відображення зайнятих та вільних дат і часу на ці дати, а також 5 вражень, які

останніми залишили відвідувачі-орендарі. Вище відображаються тільки фотографії 3 останніх авторів вражень та посилання для переходу на сторінку зі всіма враженнями.

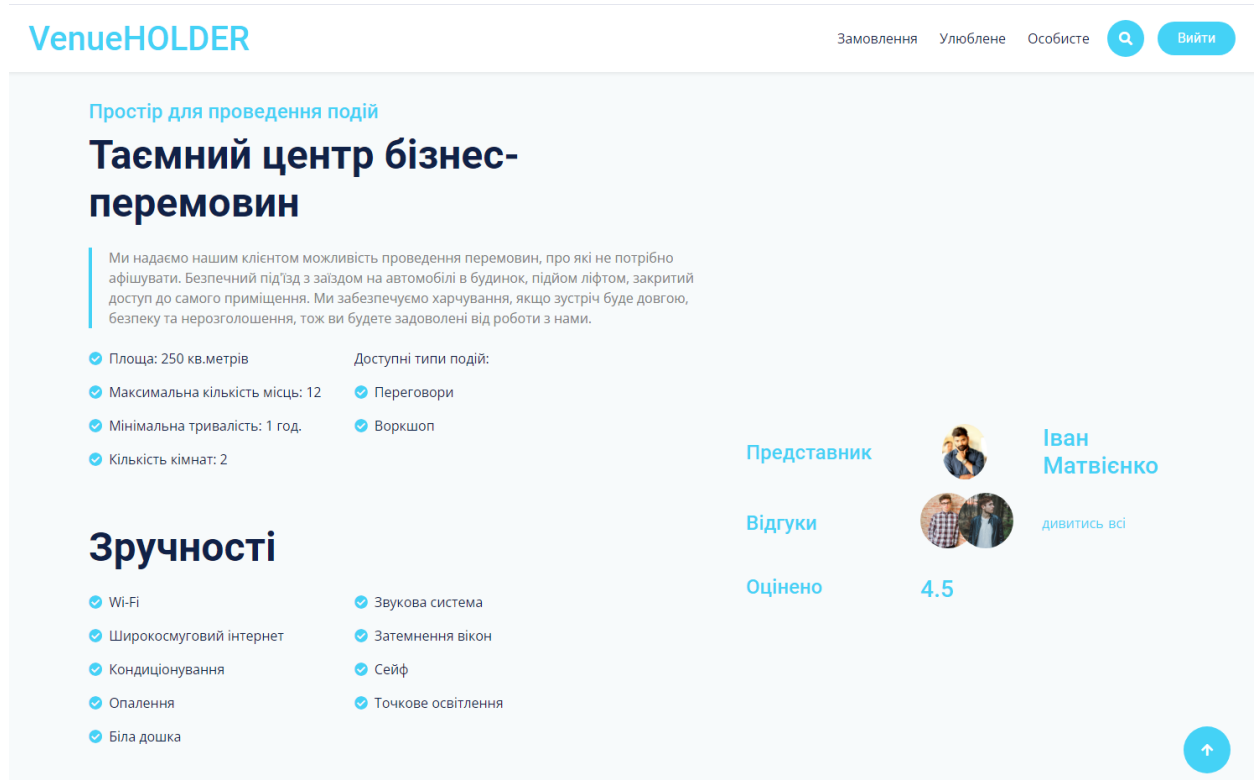


Рисунок 4.2 – Друга частина сторінки простору для проведення подій

На сторінці з замовленнями оренди просторів для проведення подій відвідувач-орендар може переглянути, які замовлення він створив, в якому вони стані, а також виконати деякі дії. Одна з них – відкрити повну інформацію про замовлення, друга – скасувати замовлення, якщо воно ще не виконувалось, третє – перейти за кнопкою, яка в такому випадку відобразиться, на сторінку для додавання враження про відвідування простору для проведення подій.

На сторінці самого простору для проведення подій можна переглянути доступний час для оренди, дізнатись про вартість оренди, що також відображається і під час пошуку просторів для проведення подій. За допомогою кнопки замовити можна перейти на сторінку, де таке замовлення формується (рис. 4.3). На цій сторінці простір вже є вибраним, однак потрібно задати додаткові параметри, що включають кількість відвідувачів події, час

оренди, тип події. Після вибору типу події можна обрати тип тривалості оренди, який визначений для такого типу події. Це впливає на доступний для вибору часу завершення оренди, тобто ці дані мають бути узгоджені.

VenueHOLDER Замовлення Улюблене Особисте Вийти

Обраний простір для проведення подій

Світла зала



Адреса
Житомир, вул. Збанацька, 14

Контактний номер
+380601010100

Вартість
10000 грн.

Тип події
Конференція

Тип тривалості
Півдня

Дата початку оренди
04.06.2024

Час початку оренди
09:00

Дата закінчення оренди
04.06.2024

Час закінчення оренди
13:00

Додаткове повідомлення

Кількість людей
25

Інвентар
Стілець

5

Вартість: 0 грн.

Додати

Харчування
Фуршет

Вартість: не обрано

Додати

Створити замовлення

Рисунок 4.3 – Завершення створення замовлення оренди простору для проведення подій

Під час створення замовлення можна додати додатковий інвентар, який потрібен під час проведення події. Він обирається з переліку за доступними можливостями для цього простору. Після вибору потрібно зазначити кількість, тоді розраховується вартість додаткового інвентаря сумарно та зліва сумарна вартість всієї оренди. Щоб додати інвентар, потрібно натиснути на відповідну кнопку. Після додавання у переліку інвентаря з'являється позиція, а біля неї хрестик для видалення. Так само можна додати потрібне харчування. Далі

необхідно обов'язково створити замовлення, щоб воно оформилось і зокрема відобразилось представнику простору для проведення подій.

Представник простору для проведення подій може авторизуватись через форму авторизації, доступну зі сторінки авторизації відвідувача-орендаря для переходу за окремим посиланням, але йому в підсумку тоді відображається своя функціональність, організована через меню. Меню містить розділ просторів, куди входить перехід до всіх просторів для проведення подій, які належать цьому представнику (рис. 4.4), а також можливість переходу до створення нового простору.

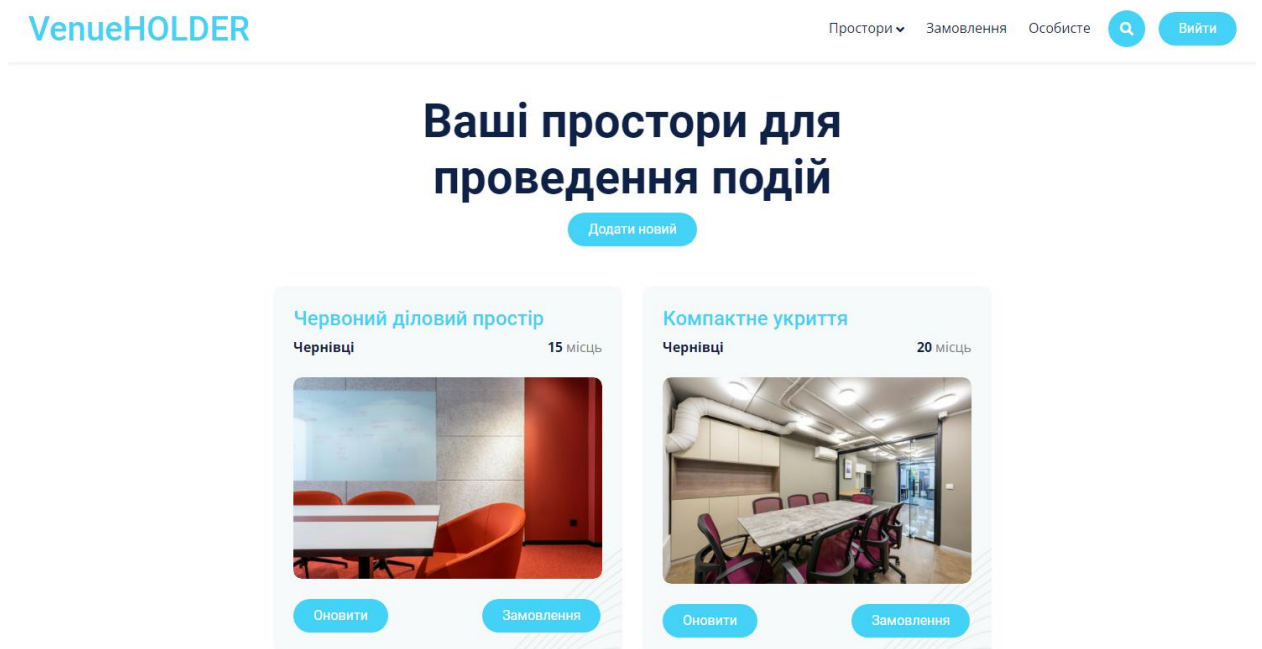


Рисунок 4.4 – Сторінка просторів для проведення подій представника у вигляді, який йому відображається

За сторінки просторів для проведення подій представника він може відкрити будь-який з цих просторів у режимі редагування через кнопку оновлення під відповідним простором для проведення подій, перейти до замовлень за цим простором, обираючи, в якому порядку їх відобразити, керуючи ними (підтверджуючи, скасовуючи, визначаючи фактичну оренду). Через розділ меню замовлень відображаються замовлення не стосовно

обраного простору для проведення подій, а стосовно всіх просторів для проведення подій представника, який авторизувався.

4.4 Тестування програми

Тестування необхідно було для того, щоб пересвідчитись, що повноцінно реалізовано в програмі те, що було заплановано. Оскільки планування представлялось сценаріями, які представляли кожну окрему функціональну можливість, то саме за ними і відбувалось тестування (табл. 4.1). Перевірка є успішною, результати є такими, якими очікувались.

Таблиця 4.1 – Результати тестування програми пошуку простору для проведення подій

Функціональна можливість	Результат
1	2
Створення запису про новий простір для проведення подій	+
Оновлення запису про простір для проведення подій	+
Формування переліку просторів для проведення подій представника	+
Пошук простору для проведення подій	+
Демонстрування сторінки простору для проведення подій	+
Замовлення оренди простору для проведення подій	+
Авторизація представника просторів для проведення подій	+
Авторизація відвідувача-орендаря	+
Реєстрація представника просторів для проведення подій	+
Реєстрація відвідувача-орендаря	+
Керування створеним замовленням оренди простору для проведення подій	+

Продовження таблиці 4.1

1	2
Формування переліку замовлень оренди простору для проведення подій, створених відвідувачем-орендарем	+
Формування переліку замовлень оренди простору для проведення подій за простором для проведенній подій	+
Формування переліку замовлень оренди простору для проведення подій за представником просторів для проведення подій	+
Керування особистими даними представником простору для проведення подій	+
Керування особистими даними відвідувачем-орендарем	+
Висловлення враження відвідувачем-орендарем про простір для проведення подій за результатами оренди	+
Висловлення враження представником простору для проведення подій про відвідувача-орендаря за результатами оренди	+
Формування переліку вражень за простором для проведення подій	+
Формування переліку вражень за представником простору для проведення подій	+
Демонстрування сторінки представника простору для проведення подій	+
Демонстрування сторінки відвідувача-орендаря	+
Формування переліку вражень за відвідувачем-орендарем	+
Відзначення простору для проведення подій як улюбленого	+
Видалення відзначення простору для проведення подій як улюбленого	+

Продовження таблиці 4.1

1	2
Формування переліку просторів для проведення подій, відзначених як улюблених	+
Керування своїм створеним замовленням оренди простору для проведення подій	+
Демонстрування повного складу замовлення оренди простору для проведення подій	+

4.5 Висновки за розділом 4

Програма може експлуатуватися і відвідувачами-орендарями, і представниками просторів для проведення подій, зважаючи на те, що усі функції успішно виконані під час тестування, інтерфейс є простим у застосуванні, структуруючи основні функціональні можливості через меню, а додаткові представляючи відповідними кнопками, що фактично працюють за логікою контекстного меню.

ВИСНОВКИ

У роботі виконано завдання аналізу проблеми розробки програми пошуку простору для проведення подій: забезпечено детальний огляд програм пошуку простору для проведення подій, включаючи Storefront, We Are Spaces, Splacer, Hire Space, виділяючи різні підходи до забезпечення роботи з просторами для проведення подій. Серед критеріїв порівняння виділено варіанти періоду оренди, що забезпечуються, підтримувані варіанти харчування під час оренди, доступність додаткового інвентаря під час оренди, можливість визначення вражень від просторів для проведення подій.

Виконано завдання спроектувати програмне забезпечення пошуку простору для проведення подій, включаючи кроки з вибору засобів, які підтримували проєктування і розробку (Python, Django, MySQL), сценарії роботи, структуру БД.

Виконано завдання реалізувати програмне забезпечення пошуку простору для проведення подій, що забезпечує потреби представників просторів для проведення подій та відвідувачів-орендарів. Представники просторів для проведення подій мають можливість використовувати її для визначення власних просторів для проведення подій, керування даними про них та доступність для оренди, створеними відвідувачами-орендарями замовленнями оренди просторів для проведення подій. Відвідувачі-орендарі мають можливість шукати такий простір, який дозволить задовольнити їхні вимоги щодо проведення події, замовляти оренду такого простору.

Розроблена програма дозволяє не тільки шукати приміщення для проведення подій, а працювати з ними саме як з просторами за рахунок вибору потрібного додаткового інвентаря для проведення подій, потрібних варіантів харчування.

Усе це сукупно дозволило досягти мету розробити програмне забезпечення пошуку простору, який міг би забезпечити необхідну підтримку проведення подій.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Storefront. 10,000+ pop-up shops, showrooms, event venues to rent [Electronic resource]. – Access mode : <https://www.thestorefront.com/>.
2. Find awesome event venues & spaces for rent in Singapore | We Are Spaces [Electronic resource]. – Access mode : <https://wearespaces.com/>.
3. Splacer. Find creative venues and corporate spaces for your next event [Electronic resource]. – Access mode : <https://www.splacer.co/>.
4. Hire Space. All the best venues for your event all in one place [Electronic resource]. – Access mode : <https://hirespace.com/>.
5. Dinder, M. Becoming an Enterprise Django Developer [Text] : Discover best practices, tooling, and solutions for writing and organizing Django applications in production / M. Dinder. – Birmingham : Packt Publishing, 2022. – 526 p.
6. Django Web Framework (Python) [Electronic resource]. – Access mode : <https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django>.
7. Ullenboom, C. Spring Boot 3 and Spring Framework 6 [Text] / C. Ullenboom. – Boston : Rheinwerk Computing, 2023. – 934 p.
8. Hinkula, J. Full Stack Development with Spring Boot 3 and React – Fourth Edition [Text] : Build modern web applications using the power of Java, React, and TypeScript / J. Hinkula. – Birmingham : Packt Publishing, 2023. – 454 p.
9. Investa – Investment Website Template [Electronic resource]. – Access mode : <https://htmlcodex.com/investment-website-template/>.

ДОДАТОК А
Технічне завдання

Вступ

Простори для проведення подій є особливими місцями, які не зовсім відповідають звичайному житлу. Тому при виконанні оренди таких просторів необхідно це враховувати, надаючи можливість задати, визначити, застосувати ті особливості, які характерні для просторів для проведення подій. Наприклад, простори можуть орендуватися для проведення подій на специфічну тривалість. Якщо для житла це або місяць (певна кількість місяців), або день (певна кількість днів), то для просторів для проведення подій це може бути повна доба, половина доби, погодинна оренда. До того ж може додаватися спеціальне обладнання, яке необхідне, щоб провести весілля, конференцію. Тобто це призводить до того, що орендується не просто приміщення, а саме простір для проведення подій. Основна мета – забезпечити повноцінне проведення події необхідного типу в орендованому для цього приміщенні. Необхідність врахування всього цього і диктує потребу в розробці окремого програмного забезпечення. З такими рішеннями в Україні є достатньо багато проблем на даний момент.

A.1 Підстави для розробки

Наказом № 168 від 24 квітня 2024 р. за Національним університетом «Запорізька політехніка» визначено тему «Програмне забезпечення пошуку простору для проведення подій» для виконання дипломної кваліфікаційної роботи.

A.2 Призначення розробки

Програма має призначене використання у вигляді забезпечення потреб представників просторів для проведення подій та відвідувачів-орендарів. Представники просторів для проведення подій повинні мати можливість

використовувати її для визначення власних просторів для проведення подій, керування даними про них та доступність для оренди, створеними відвідувачами-орендарями замовленнями оренди просторів для проведення подій. Відвідувачі-орендарі повинні мати можливість шукати такий простір, який дозволить задовольнити їхні вимоги щодо проведення події, замовляти оренду такого простору.

A.3 Основні вимоги до програми

A.3.1 Вимоги до функціональних характеристик

Функціональні характеристики програми пошуку простору для проведення подій повинні включати:

- створення запису про новий простір для проведення подій;
- оновлення запису про простір для проведення подій та додаткові параметри, які його стосуються, включаючи доступні зручності, типи доступних для проведення подій, доступний додатковий інвентар, часові інтервали роботи простору, доступні варіанти харчування, базові ставки оренди;
- формування переліку просторів для проведення подій представника;
- пошук простору для проведення подій;
- демонстрування сторінки простору для проведення подій, включаючи визначення параметрів оренди;
- замовлення оренди простору для проведення подій;
- авторизація і реєстрація представника просторів для проведення подій та відвідувача-орендаря;
- керування створеним замовленням оренди простору для проведення подій;
- формування переліку замовлень оренди простору для проведення подій, створених відвідувачем-орендарем, за простором для проведення подій або за представником просторів для проведення подій;

- керування особистими даними відвідувачем-орендарем або представником простору для проведення подій;
- висловлення враження відвідувачем-орендарем про простір для проведення подій за результатами оренди;
- висловлення враження представником простору для проведення подій про відвідувача-орендаря за результатами оренди;
- формування переліку вражень за простором для проведення подій;
- формування переліку вражень за представником простору для проведення подій;
- демонстрування сторінки представника простору для проведення подій;
- демонстрування сторінки відвідувача-орендаря;
- формування переліку вражень за відвідувачем-орендарем;
- відзначення простору для проведення подій як улюбленого та видалення такого відзначення;
- формування переліку просторів для проведення подій, відзначених як улюблених.

A.3.2 Умови експлуатації

Експлуатація можлива тільки за використання сервера, на якому можливо встановити і в подальшому встановлено засоби, які включають інтерпретатор Python версії 3, засоби створення вебдодатків на основі Django, забезпечено доступ до системи керування базами даних MySQL з можливістю програмного створення бази даних засобами фреймворку для розробки вебдодатків Django.

A.3.3 Вимоги до складу та параметрів технічних засобів

Технічні засоби визначені на рівні, що має забезпечувати приблизно 50

запитів на секунду:

- процесор не менше Intel Xeon E5440 з частотою 2.83 ГГц;
- оперативна пам'ять – 12 Гб;
- жорсткий диск – 40 Гб простору.

A.4 Порядок контролю та приймання

Виконання дипломної кваліфікаційної роботи бакалавра має бути закінчено у восьмижневий термін з розподіленням за етапами, що визначено завданням на роботу у відповідному календарному плані. Після виконання даного плану реалізується процедура захисту роботи. Захист відбувається перед екзаменаційною комісією.

ДОДАТОК Б
Текст програми

```

from django.utils.decorators import method_decorator
from .models import Venue
from .models import Visitant
from .models import Host
from .models import Venuephoto
from .models import Venueamenity
from .models import Venueaddon
from .models import Venuefoodoption
from django.contrib.auth.mixins import LoginRequiredMixin
from .models import Priceapproach
from .models import Rentview
from .models import Typevenue
from django.views import View
from .models import Favorite
from .models import Holdbook
from django.shortcuts import render
from django.views.decorators.csrf import csrf_exempt as
csrf_exempt
from django.http import HttpResponseRedirect

def VenueView(request, venueslug):
    venues = Venue.objects.filter(placeslug =
venueslug).select_related("host").select_related("holdcity")
    loggeduser = None
    if request.user.is_active:
        visitant = Visitant.objects.filter(djangouser =
request.user).prefetch_related("favorites")
        if not visitant:
            host = Host.objects.filter(djangouser =
request.user)
            if host:
                host = host[0]
                loggeduser = host
                loggeduser.type = "host"
            else:
                visitant = visitant[0]
                loggeduser = visitant
                loggeduser.type = "visitant"
    if not venues:
        return render(
            request,
            "venueholder/problem.html",
            {

```

```

        "error": "Такий простір для проведення
подій нам невідомий."
    }
)
venue = venues[0]
venue.amenities = Venueamenity.objects.filter(venue =
venue).select_related("amenity")
venue.photos = Venuephoto.objects.filter(venue = venue)
venue.types = Typevenue.objects.filter(venue =
venue).select_related("venuetype")
venue.addons = Venueaddon.objects.filter(venue =
venue).select_related("addon")
venue.foodoptions =
Venuefoodoption.objects.filter(priceapproach__typevenue__venue =
venue).select_related("foodoption").select_related("priceapproac
h").select_related("priceapproach__typevenue").select_related("p
riceapproach__typevenue__venue")
venue.rentviews = Rentview.objects.filter(
    holdbook__priceapproach__typevenue__venue = venue,
    hostorvisitant=
False).select_related("holdbook__visitant").order_by("-
viewdate")[:5]
    import datetime
    comparison_time = datetime.datetime.now()
    venue.times = Venuetime.objects.filter(
        venue = venue,
        slottimestart__gte =
comparison_time).order_by("slottimestart")
    venue.prices =
Priceapproach.objects.filter(typevenue__venue =
venue).select_related("typevenue").select_related("venuetype")
    if loggeduser and loggeduser.type == "visitant" and (
venue in loggeduser.favorites):
        venue.favorite = True
    else:
        venue.favorite = False
    return render(request, "venueholder/venue.html", venue)

def VisitantHoldbooksView(request, display = 0):
    visitant = None
    if request.user.is_active:
        visitant = Visitant.objects.filter(djangouser =
request.user).prefetch_related("favorites")
        if visitant:

```

```

        visitant = visitant[0]
    else:
        return render(
            request,
            "venueholder/problem.html",
            {
                "error": "Неавторизований користувач не
може використовувати таку функцію."
            }
        )
    if visitant:
        visitantholdbooks =
Holdbook.objects.filter(visitant = visitant)
        import datetime
        comparison_time = datetime.datetime.now()
        if display == 0:
            visitantholdbooks =
visitantholdbooks.filter(starthold__gte =
comparison_time).order_by("starthold")
        elif display == 1:
            visitantholdbooks =
visitantholdbooks.filter(starthold__gte =
comparison_time).order_by("-starthold")
        elif display == 2:
            visitantholdbooks =
visitantholdbooks.filter(starthold__lt =
comparison_time).order_by("-starthold")
        elif display == 3:
            visitantholdbooks =
visitantholdbooks.filter(starthold__lt =
comparison_time).order_by("starthold")
        elif display == 4:
            visitantholdbooks =
visitantholdbooks.order_by("booktime")
        elif display == 5:
            visitantholdbooks =
visitantholdbooks.order_by("-booktime")
        return render(request,
"venueholder/visitantholdbooks.html", visitantholdbooks)
    else:
        return render(
            request,
            "venueholder/problem.html",
            {

```

```

            "error": "Цей тип користувача не може
використовувати таку функцію."
        }
    )

```

```

def HostHoldbooksView(request, display = 0):
    host = None
    if request.user.is_active:
        host = Host.objects.filter(djangouser =
request.user)
    if host:
        host = host[0]
    else:
        return render(
            request,
            "venueholder/problem.html",
            {
                "error": "Цей тип користувача не може
використовувати таку функцію."
            }
        )
    if host:
        hostholdbooks =
Holdbook.objects.filter(priceapproach__typevenue__venue__host =
host)

        import datetime
        comparison_time = datetime.datetime.now()
        if display == 0:
            hostholdbooks =
hostholdbooks.filter(starthold__gte =
comparison_time).order_by("starthold")
        elif display == 1:
            hostholdbooks =
hostholdbooks.filter(starthold__gte =
comparison_time).order_by("-starthold")
        elif display == 2:
            hostholdbooks =
hostholdbooks.filter(starthold__lt =
comparison_time).order_by("-starthold")
        elif display == 3:
            hostholdbooks =
hostholdbooks.filter(starthold__lt =
comparison_time).order_by("starthold")
        elif display == 4:

```

```

        hostholdbooks =
hostholdbooks.order_by("booktime")
        elif display == 5:
            hostholdbooks = hostholdbooks.order_by("-
booktime")
        return render(request,
"venueholder/hostholdbooks.html", hostholdbooks)
    else:
        return render(
            request,
            "venueholder/problem.html",
            {
                "error": "Цей тип користувача не може
використовувати таку функцію."
            }
        )

def VenueHoldbooksView(request, venueslug, display = 0):
    venues = Venue.objects.filter(placeslug =
venueslug).select_related("host").select_related("holdcity")
    if not venues:
        return render(
            request,
            "venueholder/problem.html",
            {
                "error": "Такий простір для проведення
подій нам невідомий."
            }
        )
    host = None
    if request.user.is_active:
        host = Host.objects.filter(djangouser =
request.user)
    if host:
        host = host[0]
    else:
        return render(
            request,
            "venueholder/problem.html",
            {
                "error": "Цей тип користувача не може
використовувати таку функцію."
            }
        )

```

```

        venue = venues[0]
        if host:
            hostholdbooks =
Holdbook.objects.filter(priceapproach__typevenue__venue__host =
host, venue = venue)
            import datetime
            comparison_time = datetime.datetime.now()
            if display == 0:
                hostholdbooks =
hostholdbooks.filter(starthold__gte =
comparison_time).order_by("starthold")
            elif display == 1:
                hostholdbooks =
hostholdbooks.filter(starthold__gte =
comparison_time).order_by("-starthold")
            elif display == 2:
                hostholdbooks =
hostholdbooks.filter(starthold__lt =
comparison_time).order_by("-starthold")
            elif display == 3:
                hostholdbooks =
hostholdbooks.filter(starthold__lt =
comparison_time).order_by("starthold")
            elif display == 4:
                hostholdbooks =
hostholdbooks.order_by("booktime")
            elif display == 5:
                hostholdbooks = hostholdbooks.order_by("-
booktime")

            return render(request,
"venueholder/venueholdbooks.html", hostholdbooks)
        else:
            return render(
                request,
                "venueholder/problem.html",
                {
                    "error": "Цей тип користувача не може
використовувати таку функцію."
                }
            )

def HoldbookView(request, holdbookid):
    holdbook = Holdbook.objects.filter(id =
holdbookid).select_related("visitant").select_related("priceappr

```



```

        holdbook.foodoptions =
Holdbookfoodoption.objects.filter(holdbook =
holdbook).order_by("id")
        return render(request, "venueholder/holdbook.html",
holdbook)

    def VenueRentviewsView(request, venueslug):
        venues = Venue.objects.filter(placeslug =
venueslug).select_related("host").select_related("holdcity")
        if not venues:
            return render(
                request,
                "venueholder/problem.html",
                {
                    "error": "Такий простір для проведення
подій нам невідомий."
                }
            )
        venue = venues[0]
        venue.rentviews = Rentview.objects.filter(
            holdbook__priceapproach__typevenue__venue = venue,
            hostorvisitant =
False).select_related("holdbook").select_related("holdbook__visi
tant").order_by("-viewdate")
        return render(request,
"venueholder/venuerentviews.html", venue)

    def HostRentviewsView(request, hostname):
        hosts = Host.objects.filter(djangouser__username =
hostname)
        if not hosts:
            return render(
                request,
                "venueholder/problem.html",
                {
                    "error": "Такий представник простору для
проведення подій нам невідомий."
                }
            )
        host = hosts[0]
        host.rentviews = Rentview.objects.filter(
            holdbook__priceapproach__typevenue__venue__host =
host,

```

```

        hostorvisitant =
False).select_related("holdbook").select_related("holdbook__visi
tant").order_by("-viewdate")
        return render(request,
"venueholder/hostrentviews.html", host)

def VisitantRentviewsView(request, visitantname, mode = 0):
    visitant = None
    if request.user.is_active:
        visitant = Visitant.objects.filter(djangouser =
request.user)
        if visitant:
            visitant = visitant[0]
        else:
            return render(
                request,
                "venueholder/problem.html",
                {
                    "error": "Неавторизований користувач не
може використовувати таку функцію."
                }
            )
        visitant = visitant[0]
        if mode == 0:
            visitant.rentviews = Rentview.objects.filter(
                holdbook__visitant = visitant,
                hostorvisitant = False).order_by("-viewdate")
            return render(request,
"venueholder/visitantrentviews.html", visitant)
        else:
            visitant.rentviews = Rentview.objects.filter(
                holdbook__visitant = visitant,
                hostorvisitant = True).order_by("-viewdate")
            return render(request,
"venueholder/visitantrentviews.html", visitant)

def HostVenuesView(request, hostname):
    hosts = Host.objects.filter(djangouser__username =
hostname)
    if not hosts:
        return render(
            request,
            "venueholder/problem.html",
            {

```

```

        "error": "Такий представник простору для
првoедення подій нам невідомий."
    }
    )
    host = hosts[0]
    host.venues = Venue.objects.filter(
        host =
host).select_related("holdcity").order_by("id")
    return render(request, "venueholder/hostvenues.html",
host)

def FavoriteVenuesView(request):
    visitant = None
    if request.user.is_active:
        visitant = Visitant.objects.filter(djangouser =
request.user).prefetch_related("favorites").select_related("favo
rites__venue").select_related("favorites__venue__holdcity")
        if visitant:
            visitant = visitant[0]
    else:
        return render(
            request,
            "venueholder/problem.html",
            {
                "error": "Неавторизований користувач не
може використовувати таку функцію."
            }
        )
    return render(request, "venueholder/favvenues.html",
visitant)

def HostProfileView(request, hostname):
    hosts = Host.objects.filter(djangouser__username =
hostname)
    if not hosts:
        return render(
            request,
            "venueholder/problem.html",
            {
                "error": "Такий представник простору для
првoедення подій нам невідомий."
            }
        )
    host = hosts[0]

```

```

        host.venues = Venue.objects.filter(
            host =
host).select_related("holdcity").order_by("id")
        host.rentviews = Rentview.objects.filter(
            holdbook__priceapproach__typevenue__venue__host =
host,
            hostorvisitant =
False).select_related("holdbook").select_related("holdbook__visi
tant").order_by("-viewdate")[:5]
        host.createdrentviews = Rentview.objects.filter(
            holdbook__priceapproach__typevenue__venue__host =
host,
            hostorvisitant =
True).select_related("holdbook").select_related("holdbook__visit
ant").order_by("-viewdate")[:5]
        return render(request, "venueholder/host.html", host)

def VisitantProfileView(request, visitantname):
    visitant = None
    if request.user.is_active:
        visitant = Visitant.objects.filter(djangouser =
request.user)
    if visitant:
        visitant = visitant[0]
    else:
        return render(
            request,
            "venueholder/problem.html",
            {
                "error": "Неавторизований користувач не
може використовувати таку функцію."
            }
        )
    visitant = visitant[0]
    visitant.createdrentviews = Rentview.objects.filter(
        holdbook__visitant = visitant,
        hostorvisitant = False).order_by("-viewdate")[:5]
    visitant.rentviews = Rentview.objects.filter(
        holdbook__visitant = visitant,
        hostorvisitant = True).order_by("-viewdate")[:5]
    return render(request, "venueholder/visitant.html",
visitant)

def VenuesSearchView(request):

```

```

        foundvenues = Venue.objects.all()
        #забезпечення форми пошуку переліком міст, де простори
        можуть знаходитись
        holdcities =
        Holdcity.objects.all().order_by('cityname')
        #забезпечення форми пошуку переліком зручностей, які
        можуть просторами надаватися
        amenities =
        Amenity.objects.all().order_by('captionshort')
        #забезпечення форми пошуку переліком типів подій, що
        можна обирати
        venuetypes =
        Venuetype.objects.all().order_by('activityname')
        addons = Addon.objects.all().order_by('captionshort')
        foodoptions =
        Foodoption.objects.all().order_by('captionshort')
        holdcityname = self.request.GET.get('holdcity', None)
        if holdcityname:
            holdcity = Holdcity.objects.filter(cityname =
holdcityname)
            foundvenues = foundvenues.filter(holdcity__in =
holdcity)
            people = self.request.GET.get('people', None)
            if people:
                people = int(people)
                foundvenues = foundvenues.filter(maxpeople__gte =
people)
            hourstorent = self.request.GET.get('hours', None)
            if hourstorent:
                hourstorent = int(hourstorent)
                foundvenues = foundvenues.filter(minhours__lte =
hourstorent)
            else:
                hourstorent = 1
            sqmeters = self.request.GET.get('metersmin', None)
            if sqmeters:
                sqmeters = int(sqmeters)
                foundvenues = foundvenues.filter(sqmeters__gte =
sqmeters)
            sqmeters = self.request.GET.get('metersmax', None)
            if sqmeters:
                sqmeters = int(sqmeters)
                foundvenues = foundvenues.filter(sqmeters__lte =
sqmeters)

```

```

cabinets = self.request.GET.get('cabinets', None)
if cabinets:
    cabinets = int(cabinets)
    foundvenues = foundvenues.filter(cabinets__gte =
cabinets)
assessment = self.request.GET.get('assessmentmax',
None)
if assessment:
    assessment = float(assessment)
    foundvenues = foundvenues.filter(assessment__lte =
assessment)
assessment = self.request.GET.get('assessmentmin',
None)
if assessment:
    assessment = float(assessment)
    foundvenues = foundvenues.filter(assessment__gte =
assessment)
amenitiesparameters = self.request.GET.get('amenities',
None)
if amenitiesparameters:
    amenitiesparameters =
amenitiesparameters.split(',')
    amenities_foundvenues = []
    for foundvenue in foundvenues:
        venue_amenity_check = True
        for amenity in amenitiesparameters:
            amenity = int(amenity)
            venue_amenity =
Venueamenity.objects.filter(
                venue = foundvenue,
                amenity = amenity
            )
            if not venue_amenity:
                venue_amenity_check = False
        if venue_amenity_check:
            amenities_foundvenues.append(foundvenue)
    foundvenues = amenities_foundvenues
foodoptionsparameters =
self.request.GET.get('foodoptions', None)
if foodoptionsparameters:
    foodoptionsparameters =
foodoptionsparameters.split(',')
    foodoptions_foundvenues = []
    for foundvenue in foundvenues:

```



```

        if type_venue:
            type_foundvenues.append(foundvenue)
    rentdt = self.request.GET.get('time', None)
    if not rentdt:
        return render(
            request,
            "venueholder/problem.html",
            {
                "error": "Пошук обов'язково повинен
визначати час початку оренди."
            }
        )
    from datetime import datetime
    used_format = "%d-%m-%Y %H:%M"
    rentdt_time = datetime.strptime(rentdt, used_format)
    from datetime import timedelta
    rentdt_time_end = rentdt_time + timedelta(hours =
hourstorent)
    time_foundvenues = []
    #відбір просторів для проведення подій, які є вільними
на потрібний час
    for foundvenue in foundvenues:
        times = Venuetime.objects.filter(
            venue = foundvenue,
            slottimestart__lte = rentdt_time,
            slottimeend__gte = rentdt_time_end,
            freeoroccupied = True
        )
        if times:
            time_foundvenues.append(foundvenue)
    price_max = self.request.GET.get('price', None)
    duration_type = self.request.GET.get('long', None)
    if duration_type:
        if duration_type == 0 and (hourstorent < 24):
            return render(
                request,
                "venueholder/problem.html",
                {
                    "error": "Доба триває не менше 24
годин. Такий тариф застосувати неможливо"
                }
            )
        elif duration_type == 1 and ((hourstorent < 4) or
(hourstorent > 12)):

```

```

        return render(
            request,
            "venueholder/problem.html",
            {
                "error": "Строк у півдня триває не
більше 12 годин, але і не менше 4. Такий тариф застосувати
неможливо"
            }
        )
    else:
        duration_type = 2
        price_foundvenues = []
        #відбір просторів для проведення подій, вартість оренди
        яких не перевищує потрібну
        for foundvenue in foundvenues:
            priceapproaches = Priceapproach.objects.filter(
                typevenue__venue = foundvenue,
                typevenue__venuetype = venuetype,
                typevenue__accessed = True
            )
            price_found = False
            for priceapproach in priceapproaches:
                if priceapproach.renttimetype == 0:
                    days = hourstorent // 24
                    days = days + 1
                    calc_price = priceapproach.price * days
                    if calc_price <= price_max:
                        price_found = True
                elif priceapproach.renttimetype == 1:
                    calc_price = priceapproach.price
                    if calc_price <= price_max:
                        price_found = True
                elif priceapproach.renttimetype == 2:
                    calc_price = priceapproach.price *
hourstorent
                    if calc_price <= price_max:
                        price_found = True
            if price_found:
                price_foundvenues.append(priceapproach)
                break
    return render(request, "venueholder/search.html",
        { "found": foundvenues,
          "holdcities": holdcities,
          "amenities": amenities,

```



```

        "error": "Потрібно авторизуватися спершу."
    }
    )
    if request.method == "POST":
        rentview_form = RentviewForm(request.POST)
        if rentview_form.is_valid() and ((loggeduser.type
== "visitant" and holdbook.visitant == loggeduser) or
(loggeduser.type == "host" and
holdbook.priceapproach.typevenue.venue.host == loggeduser)):
            rentview = rentview_form.save(commit=False)
            if loggeduser.type == "visitant":
                rentview.hostorvisitant = False
            else:
                rentview.hostorvisitant = True
            rentview.save()
            return
    HttpResponseRedirect(reverse("venueholder:visitantrentviews",
args=( loggeduser.djangouser.username,)))
    else:
        rentview_form = RentviewForm()
        return render(request,
"venueholder/rentviewpublish.html", {"rentview_form":
rentview_form })

    @method_decorator(csrf, name='dispatch')
    class VenueFavoriteView(LoginRequiredMixin, View):
        def post(self, request, pk):
            venues = Venue.objects.filter(id = pk)
            if venues:
                venue = venues[0]
            else:
                return render(
                    request,
                    "venueholder/problem.html",
                    {
                        "error": "Такий простір для проведення
подій нам невідомий."
                    }
                )
            if request.user.is_active:
                visitant = Visitant.objects.filter.djangouser =
request.user).prefetch_related("favorites")
                if not visitant:
                    return render(

```

```

        request,
        "venueholder/problem.html",
        {
            "error": "Потрібно спочатку
авторизуватися."
        }
    )
    if venue not in visitant.favorites:
        visitant.favorites.add(venue)
        visitant.save()
    else:
        visitant.favorites.remove(venue)
        visitant.save()
    return HttpResponse()

```

ДОДАТОК В
Слайди презентації



Рисунок В.1 – Перший слайд

Об'єкт, предмет, мета роботи

- Об'єкт роботи – процес пошуку простору для проведення подій.
- Предмет роботи – програмне забезпечення пошуку простору для проведення подій.
- Мета роботи – розробити програмне забезпечення пошуку простору, який міг би забезпечити необхідну підтримку проведення події.

2

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

Завдання роботи

- проаналізувати проблему розробки програми пошуку простору для проведення подій;
- спроектувати програмне забезпечення пошуку простору для проведення подій;
- реалізувати програмне забезпечення пошуку простору для проведення подій.

3

Рисунок В.3 – Завдання роботи

Порівняння існуючих аналогів та розроблюваного програмного забезпечення пошуку простору для проведення подій

Характеристика	Storefront	We Are Spaces	Spacer	Hire Space	Розробка
Період оренди	Кількість діб	Погодинно, півдня, доба, кількість діб	Погодинно	–	Погодинно, півдня, доба, кількість діб
Варіанти харчування	–	+	–	–	+
Додатковий інвентар	–	–	–	–	+
Враження від просторів для проведення подій	+	–	+	–	+

4

Рисунок В.4 – Порівняння існуючих аналогів та розроблюваного програмного забезпечення пошуку простору для проведення подій

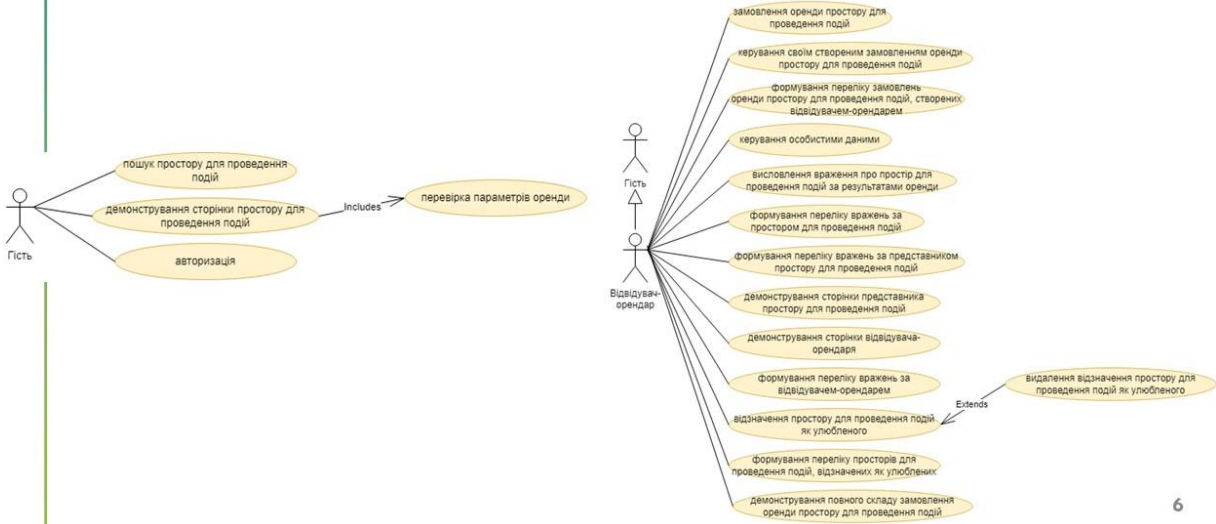
Порівняння фреймворків розробки програми пошуку простору для проведення подій

Критерій	Django	Spring Boot
Фокус розробки	Швидкість, керування даними	Масштабність
Підхід до побудови	Містить усі необхідні засоби одразу	Модульний
Мова програмування	Python	Java

5

Рисунок В.5 – Порівняння фреймворків розробки програми пошуку простору для проведення подій

Діаграми варіантів використання гостем та відвідувачем-орендарем програми пошуку простору для проведення подій



6

Рисунок В.6 – Діаграми варіантів використання гостем та відвідувачем-орендарем програми пошуку простору для проведення подій

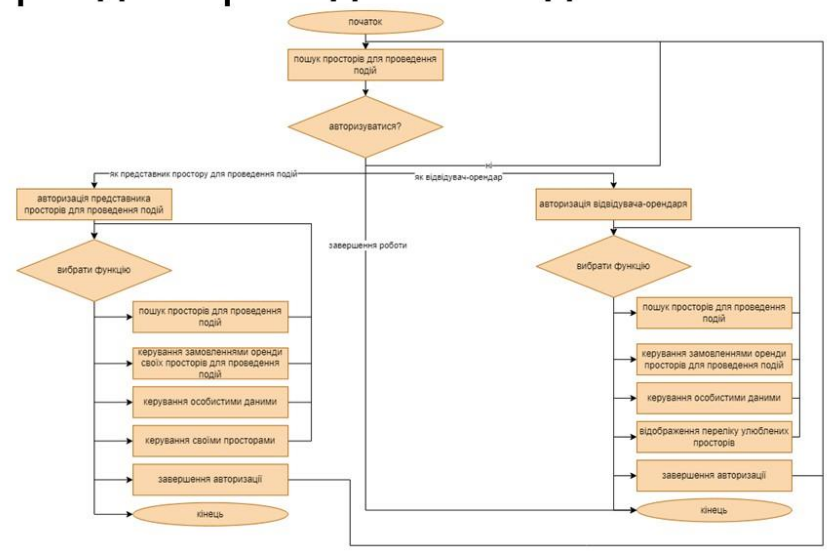


Схема бази даних програми пошуку просторів для проведення подій



Рисунок В.8 – Схема бази даних програми пошуку просторів для проведення подій

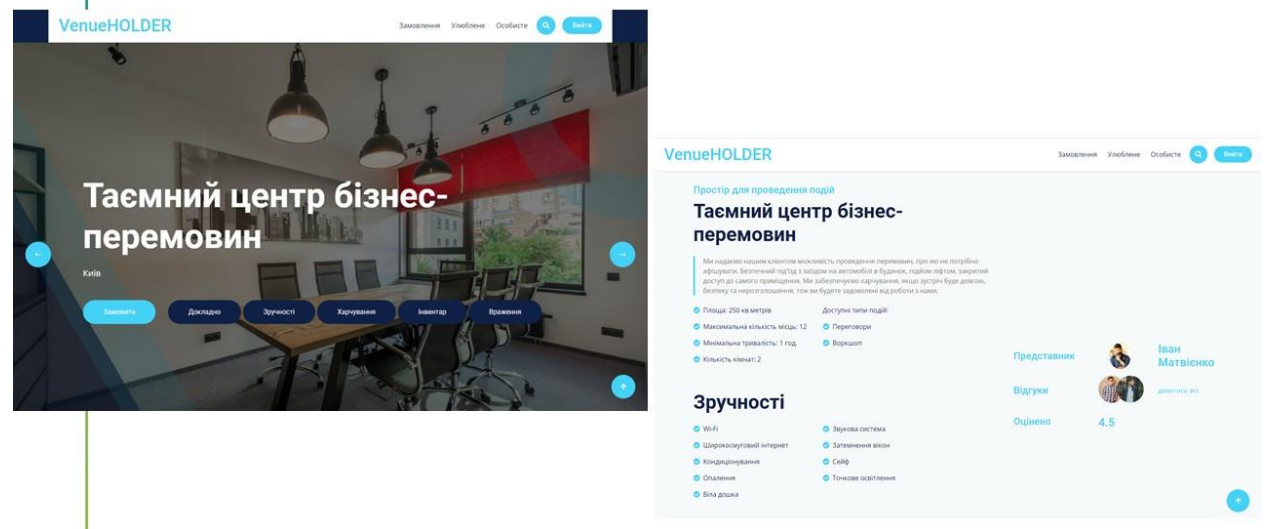
Схема функціонування програми пошуку просторів для проведення подій



9

Рисунок В.9 – Схема функціонування програми пошуку просторів для проведення подій

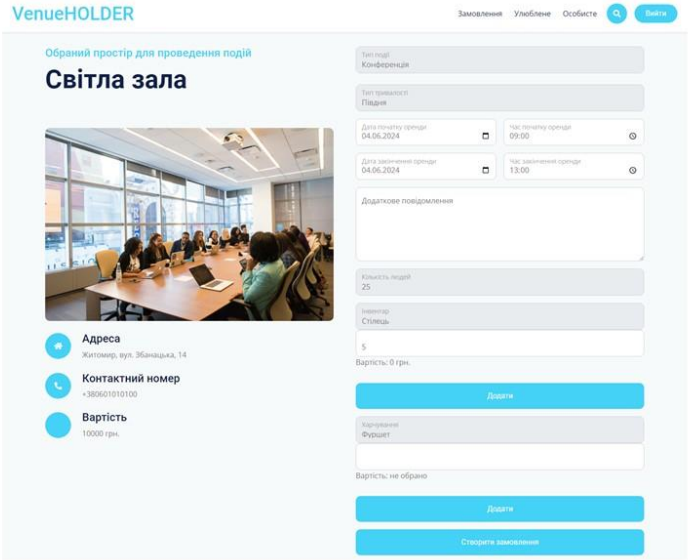
Сторінка простору для проведення подій



10

Рисунок В.10 – Сторінка простору для проведення подій

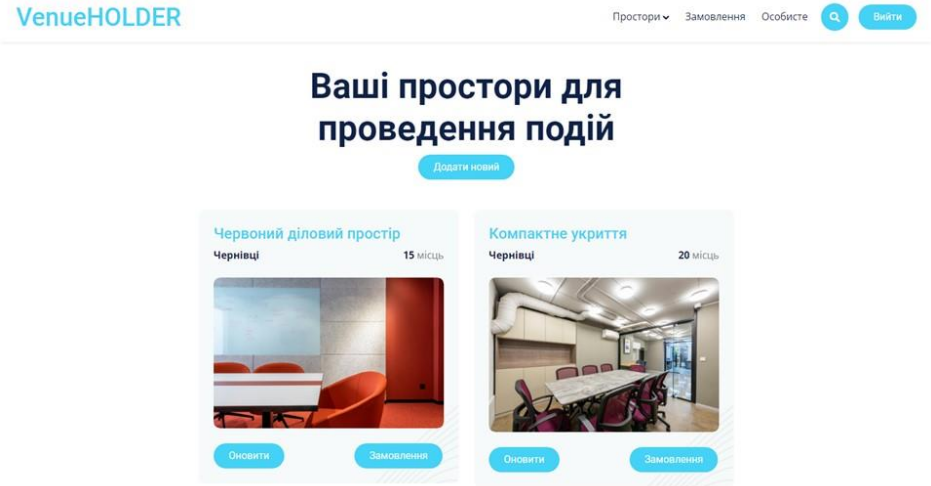
Створення замовлення оренди простору для проведення подій



11

Рисунок В.11 – Створення замовлення оренди простору для проведення подій

Сторінка просторів для проведення подій представника



12

Рисунок В.12 – Сторінка просторів для проведення подій представника

Результати тестування програми пошуку простору для проведення подій

Функціональна можливість	Результат	Функціональна можливість	Результат
Створення запису про новий простір для проведення подій	+	Формування переліку замовлень оренди простору для проведення подій, створених відвідувачем-орендарем	+
Оновлення запису про простір для проведення подій	+	Формування переліку замовлень оренди простору для проведення подій за простором для проведення подій	+
Формування переліку просторів для проведення подій представника	+	Формування переліку замовлень оренди простору для проведення подій за представником просторів для проведення подій	+
Пошук простору для проведення подій	+	Керування особистими даними представником простору для проведення подій	+
Демонстрування сторінки простору для проведення подій	+	Керування особистими даними відвідувачем-орендарем	+
Замовлення оренди простору для проведення подій	+	Висловлення враження відвідувачем-орендарем про простір для проведення подій за результатами оренди	+
Авторизація представника просторів для проведення подій	+	Висловлення враження представником простору для проведення подій про відвідувача-орендаря за результатами оренди	+
Авторизація відвідувача-орендаря	+	Формування переліку вражень за простором для проведення подій	+
Реєстрація представника просторів для проведення подій	+	Формування переліку вражень за представником простору для проведення подій	+
Формування переліку просторів для проведення подій, відзначених як улюблених	+	Демонстрування сторінки представника простору для проведення подій	+
Керування своїм створеним замовленням оренди простору для проведення подій	+	Демонстрування сторінки відвідувача-орендаря	+
Демонстрування повного складу замовлення оренди простору для проведення подій	+	Формування переліку вражень за відвідувачем-орендарем	+
Реєстрація відвідувача-орендаря	+	Відзначення простору для проведення подій як улюбленого	+
Керування створеним замовленням оренди простору для проведення подій	+	Видалення відзначення простору для проведення подій як улюбленого	+

Рисунок В.13 – Результати тестування програми пошуку простору для проведення подій

Висновки

- У роботі виконано завдання аналізу проблеми розробки програми пошуку простору для проведення подій: забезпечено детальний огляд програм пошуку простору для проведення подій, включаючи Storefront, We Are Spaces, Splacer, Hire Space, виділяючи різні підходи до забезпечення роботи з просторами для проведення подій. Серед критеріїв порівняння виділено варіанти періоду оренди, що забезпечуються, підтримувані варіанти харчування під час оренди, доступність додаткового інвентаря під час оренди, можливість визначення вражень від просторів для проведення подій.
- Виконано завдання спроектувати програмне забезпечення пошуку простору для проведення подій, включаючи кроки з вибору засобів, які підтримували проектування і розробку (Python, Django, MySQL), сценарії роботи, структуру бази даних.
- Виконано завдання реалізувати програмне забезпечення пошуку простору для проведення подій, що забезпечує потреби представників просторів для проведення подій та відвідувачів-орендарів. Представники просторів для проведення подій мають можливість використовувати її для визначення власних просторів для проведення подій, керування даними про них та доступність для оренди, створеними відвідувачами-орендарями замовленнями оренди просторів для проведення подій. Відвідувачі-орендарі мають можливість шукати такий простір, який дозволить задовольнити їхні вимоги щодо проведення події, замовляти оренду такого простору.

Рисунок В.14 – Висновки