

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

(назва теми)

ОРЕНДИ СПОРТИВНОГО ІНВЕНТАРЯ

SPORTS EQUIPMENT RENTAL SOFTWARE

Виконав(ла): студент(ка) 4 курсу, групи КНТ-139

Спеціальності 121 Інженерія програмного

(код і найменування спеціальності)

забезпечення

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

СЛАВНИЙ Б.Ю.

(ПРИЗВИЩЕ та ініціали)

Керівник СКРУПСЬКИЙ С.Ю.

(ПРИЗВИЩЕ та ініціали)

Рецензент ГОЛУБ Т.В.

(ПРИЗВИЩЕ та ініціали)

2023

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

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

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
Сергій СУББОТІН
« » 2023 року

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

СЛАВНОГО Богдана Юрійовича

(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Програмне забезпечення оренди спортивного інвентаря. Sports Equipment Rental Software

керівник проєкту (роботи) к.т.н., доцент СКРУПСЬКИЙ Степан Юрійович,
(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від «04» квітня 2023 року № 87

2. Строк подання студентом проєкту (роботи) 12 червня 2023 року

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

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

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

Слайди презентації

6. Консультанти розділів проєкту (роботи)

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

7. Дата видачі завдання « 11 » квітня 2023 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)

_____ Богдан СЛАВНИЙ
(підпис) (Ім'я ПРИЗВИЩЕ)

Керівник проєкту (роботи)

_____ Степан СКРУПСЬКИЙ
(підпис) (Ім'я ПРИЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 98 с., 26 рис., 2 табл., 3 дод., 11 джерел.

СПОРТ, СПОРТИВНИЙ ІНВЕНТАР, ОРЕНДА, БРОНЮВАННЯ, ВЕБЗАСТОСУНОК.

Об'єкт роботи – процес оренди спортивного інвентаря. Предмет роботи – програмне забезпечення оренди спортивного інвентаря. Мета роботи – розробити програмне забезпечення оренди спортивного інвентаря для підтримки керування відповідними процесами зі сторони господарів і орендарів.

Матеріали, методи та технічні засоби: мова Python, підтримка веброзробки мовами HTML, CSS, Javascript.

Результати. Розроблено програму, яка забезпечує підтримку господарів спортивного інвентаря щодо виставлення спортивного інвентаря у вигляді пропозицій оренди, керування самими цими пропозиціями і бронюваннями оренди спортивного інвентаря, а підтримку інших користувачів щодо пошуку тих пропозицій, які задовольняють їх вимогам, та бронювання оренди спортивного інвентаря за цими пропозиціями.

Висновки. Розроблене програмне забезпечення складається з інтерфейсів для гостя, господаря та авторизованого користувача. Інтерфейси господаря та авторизованого користувача враховують відповідну специфіку та базуються на інтерфейсі гостя. Створені засоби дозволяють керувати орендою спортивного інвентаря, починаючи взагалі від внесення пропозицій, завершуючи програмним переведенням до успішно завершеної оренди або поданням скарги господарем або звичайним авторизованим користувачем, яким є орендар, на бронювання.

Галузь використання – оренда спортивного інвентаря.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 98 pages, 26 figures, 2 tables, 3 appendixes, 11 sources.

SPORTS, SPORTS EQUIPMENT, RENT, RESERVATION, WEBSITE.

The object of the work is the process of renting sports equipment. The subject of the work is software for renting sports equipment. The purpose of the work is to develop sports equipment rental software to support the management of the relevant processes on the part of owners and tenants.

Materials, methods and technical means: Python language, web development support using HTML, CSS, Javascript languages.

The results. A program has been developed which provides support for sports equipment owners to present sports equipment in the form of rental offers, manage these offers and sports equipment rental reservations, and support other users in finding offers that meet their requirements and booking sports equipment rentals based on these offers.

Conclusions. The developed software consists of guest, host and authorized user interfaces. The host and authorized user interfaces take into account the relevant specifics and are based on the guest interface. The created tools allow users to manage the rental of sports equipment, starting from making offers, ending with a transfer of a reservation to a successfully completed rental or submitting a complaint by the owner or a regular authorized user, who is a tenant, on a reservation.

The field of use is the rental of sports equipment.

ЗМІСТ

	С.
Перелік скорочень та умовних познач.....	7
Вступ.....	8
1 Аналіз предметної області.....	9
1.1 Аналіз програмних засобів оренди спортивного інвентаря	9
1.2 Визначення концепції розроблюваної програми	13
1.3 Висновки за розділом 1 та постановка завдань роботи	14
2 Розробка архітектури програми.....	15
2.1 Вибір технологій розробки	15
2.2 Моделювання вимог до програми	16
2.3 Проєктування бази даних	21
2.4 Висновки за розділом 2	36
3 Реалізація програми	37
3.1 Схема функціонування програми.....	37
3.2 Програмна реалізація функцій вебсистеми	38
3.3 Висновки за розділом 3	44
4 Експлуатація та тестування програми	45
4.1 Призначення програми	45
4.2 Умови виконання програми	45
4.3 Виконання і тестування програми.....	45
4.4 Висновки за розділом 4	50
Висновки	51
Перелік джерел посилання	52
Додаток А Технічне завдання	54
Додаток Б Текст програми	59
Додаток В Слайди презентації.....	92

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

БД – база даних;

ПЗ – програмне забезпечення.

ВСТУП

Оренда спортивного інвентаря є достатньо розповсюдженим явищем. Розмір ринку в США зараз становить 1,3 мільярда доларів США, і протягом останніх декількох років ринок стабільно зростає на 5-10%, прогнозований ріст становить 5% принаймні протягом наступних п'яти років: покоління міленіалів є найбільшим демографічним сегментом, який бере участь у цьому ринку, відповідно можна говорити, зважаючи на вік цієї групи, на перспективу подальшого розвитку цієї сфери [1]-[4].

Сервіси прокату велосипедів, обладнання для зимових видів спорту відомі всім. І ці приклади і твердження тільки підкреслюють, що саме оренда є доволі розповсюдженою для них, а вони є як раз прикладами спортивного інвентаря. Однак в такому вигляді, як це відбувається найчастіше, існують певні сервіси прокату, вони є централізованими, надають типові обладнання, у той же час власники подібного інвентаря є джерелом для оренди. Тож це джерело можна застосувати для оренди спортивного інвентаря, але це потребує підтримки за допомогою програмного забезпечення (ПЗ), адже цей ринок є дуже фрагментованим: всі господарі знаходяться в різних місцях, мають дуже різне обладнання. Відповідно ПЗ і має бути розроблено в роботі.

Мета роботи – розробити ПЗ оренди спортивного інвентаря для підтримки керування відповідними процесами зі сторони господарів і орендарів.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз програмних засобів оренди спортивного інвентаря

Оренда спортивного інвентаря є розповсюдженою справою, тому результати пошуку відповідного ПЗ виявились також широкими. Початкова фаза аналізу була сконцентрована на тому ПЗ, що підтримує оренду певної категорії спортивного обладнання. Якщо виділяти з цього напрямку ПЗ, яке підтримує такий варіант оренди в Україні, то потрібно зазначити, що в Україні ці інформаційні технології не розповсюджені. Вебсайти фактично виконують роль інформаційних (рис. 1.1), як наприклад, X-drive [5], а як таких засобів бронювання не мають. Тому повноцінної підтримки оренди спортивного інвентаря на основі ПЗ в Україні на даний момент немає. І це слід вважати проблемою, адже це не сприяє зручності користувачів, яку вони мають в багатьох інших галузях.

У комплект сноуборд входить:
Сноуборд + Боти + Шолом



ЦІНА ПРОКАТУ КОМПЛЕКТІВ

КОМПЛЕКТ ЛИЖ	пів дня*	1 день	2 дні	3 дні	4 дні	5 днів	6 днів	7 днів	+1 день	Застава**
Комплект Лижі Стандарт	180	220	440	620	800	980	1140	1280	140	5500-7000
Комплект Лижі Преміум	240	300	600	860	1120	1380	1620	1890	200	7000-9000
Комплект Лижі Топ	340	420	840	1220	1610	1980	2340	2680	300	9000-11000
Комплект Лижі Стандарт Дитячі	110	150	300	420	530	640	730	820	80	4000-5000
Комплект Лижі Преміум Дитячі	180	220	440	620	800	980	1140	1280	140	5500-7000
КОМПЛЕКТ СНОУБОРД										
Комплект Сноуборд Стандарт	180	220	440	620	800	980	1140	1280	140	5500-7000
Комплект Сноуборд Преміум	240	300	600	860	1120	1380	1620	1890	200	7000-9000
Комплект Сноуборд Топ	340	420	840	1220	1610	1980	2340	2680	300	9000-11000
Комплект Сноуборд Стандарт Дитячий	110	150	300	420	530	640	730	820	80	4000-5000
Комплект Сноуборд Преміум Дитячий	180	220	440	620	800	980	1140	1280	140	5500-7000

Рисунок 1.1 – Вебсайт X-drive

У світі існує достатньо багато вебсайтів, які підтримують роботу окремих сервісів оренди деякої категорії спортивного інвентаря. Однак, оскільки деякі бізнеси є крупними, то деякі вебсайти підтримують дані про цілі мережі. Наприклад, Snowrental [6] містить велику кількість прокатів інвентаря у різних країнах світу (рис. 1.2).

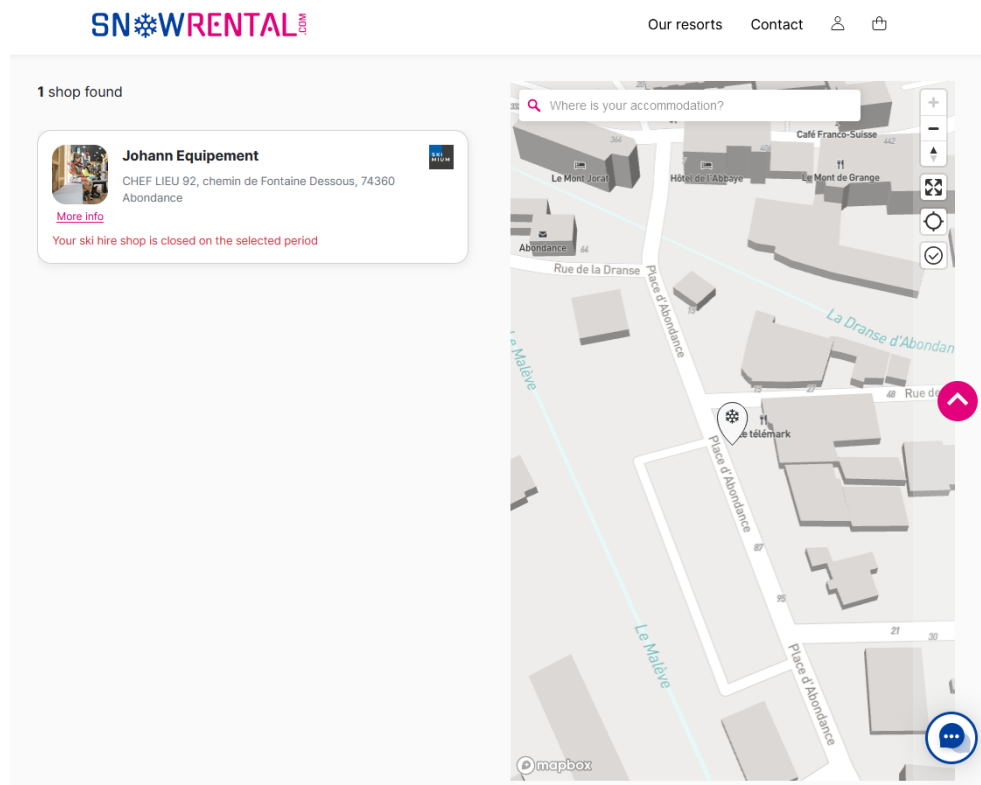


Рисунок 1.2 – Вебсайт Snowrental

Snowrental містить інформацію про курорти мережі, дозволяє виконувати пошук сервісів за відповідним курортом, виконувати бронювання спортивного інвентаря. Однак, наприклад, під час тестового звернення було виявлено, що на даний момент функція бронювання фактично не працює, адже дати пошуку не можна було задати. Пошук організовувався автоматично на наступний тиждень. Тобто слід зазначити, що навіть наявні сервіси можуть бути зіпсовані негативним досвідом роботи з ними.

Checkyeti [7] (рис. 1.3) дозволяє бронювати лижі, сноуборди. Однак ці засоби достатньо переплутані з пошуком загалом активностей, в тому числі,

наприклад, літніх. При цьому сам процес бронювання є достатньо простим, чітко виділеним і зрозумілим. Вебсайт також містить відгуки від користувачів як стосовно програми загалом, так і стосовно конкретних сервісів прокату.

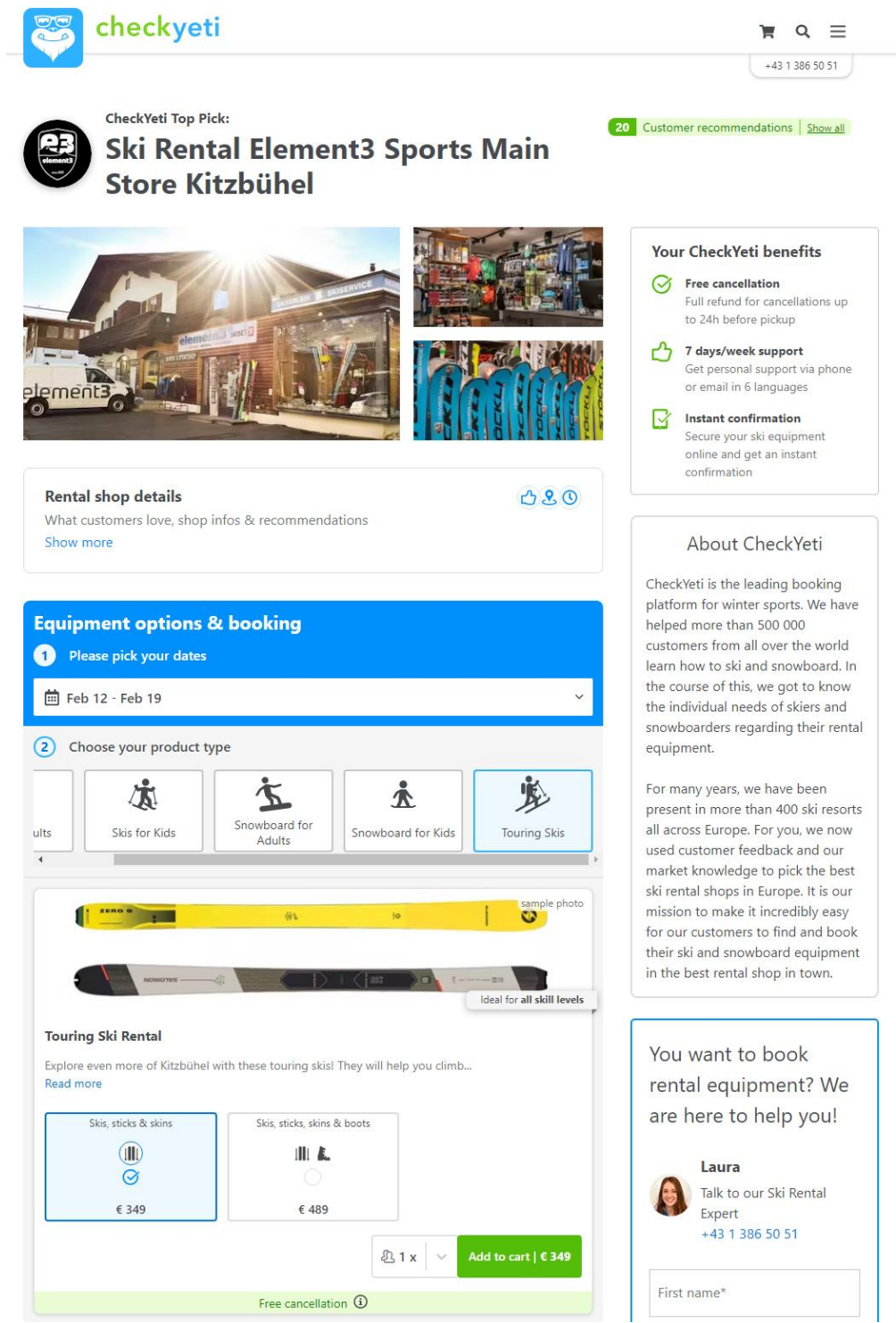


Рисунок 1.3 – Вебсайт Checkyeti

Відмінністю вебзастосунку FriendWithA [8] (рис. 1.4) є те, що він дозволяє бронювати оренду спортивного інвентаря від власників. Тобто це не оренда від сервісів прокату. Вебсайт містить не тільки спортивний інвентар, а він представлений шоломами, велосипедами, лижами, сноубордами та інвентарем для водних видів спорту.

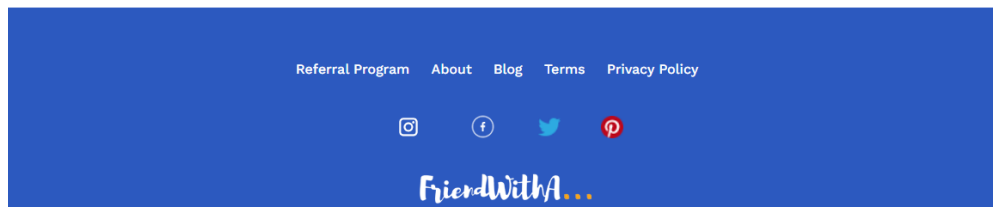
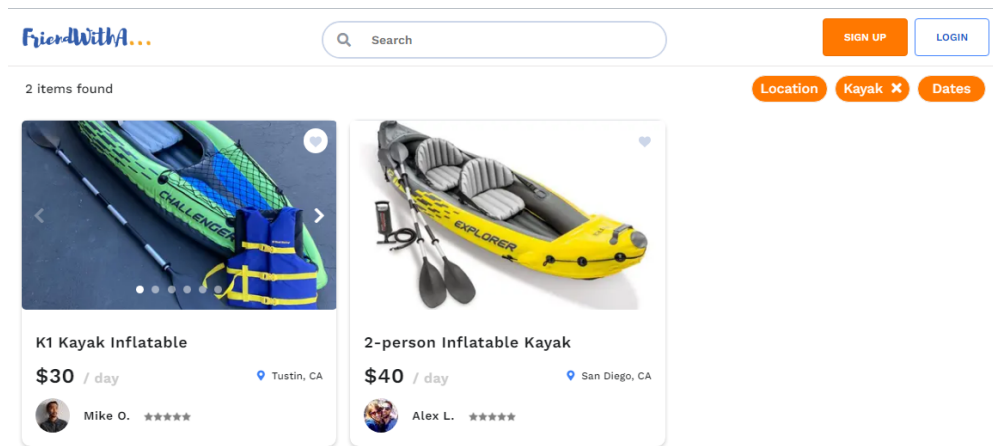


Рисунок 1.4 – Пошук спортивного обладнання в FriendWithA

FriendWithA має достатньо особливу концепцію в своїй сфері, що виділяє його серед всіх аналогів. Містяться також відгуки, що є важливими для всіх вебсайтів, пов'язаних з орендою, але відгуки формуються про самих господарів, а не про конкретний інвентар. Відповідно і оцінка визначена для господаря (рис. 1.5). Це не сприяє зручності роботи, а зі збільшенням наявної кількості пропозицій у господаря це призводить до того, що відгуки фактично не представляють інформації про те, яка ж саме користь від спортивного інвентаря.

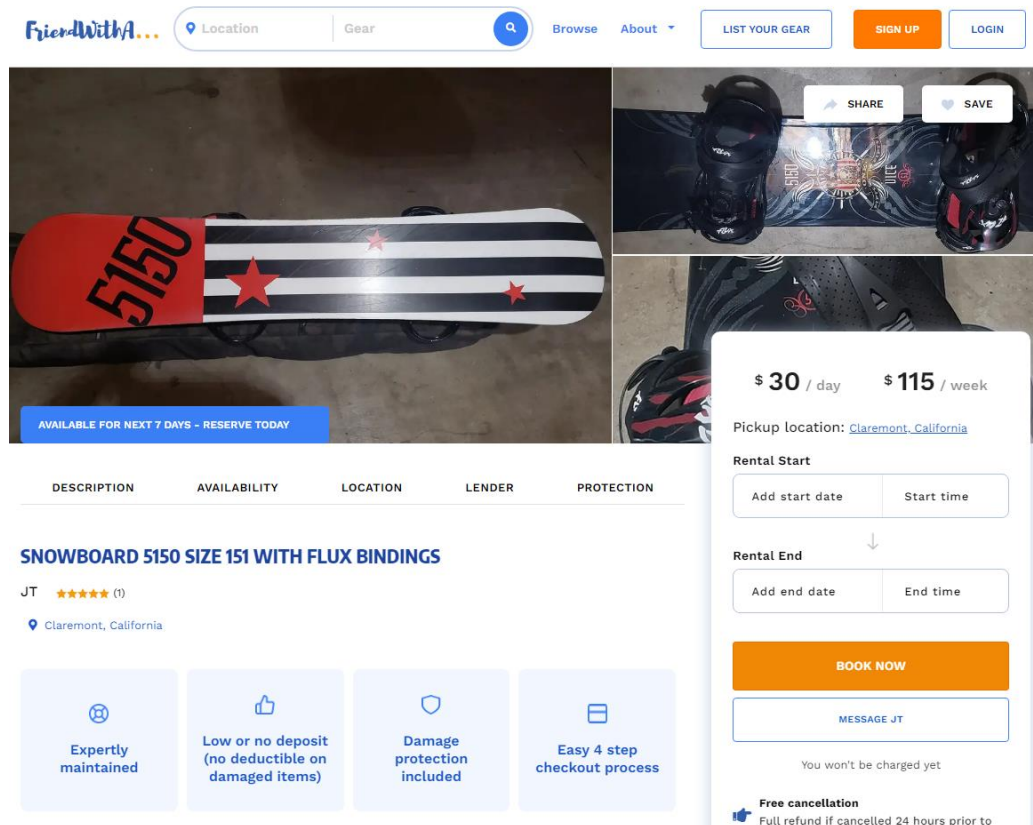


Рисунок 1.5 – Сторінка оголошення спортивного обладнання в FriendWithA

1.2 Визначення концепції розроблюваної програми

Серед того ПЗ, яке проаналізовано, виокремлено Checkyeti, що є потужним засобом стосовно роботи з сервісами прокату, та FriendWithA, який реалізує роботи з господарями спортивного інвентаря (табл. 1.1).

Програма, яку заплановано розробити, потенційно може використовуватися для роботи з пунктами прокату, але основним джерелом самого спортивного інвентаря є її господарі, що аналогічно FriendWithA. Відмінностями є те, що передбачено створення відгуків і про сам спортивний інвентар зокрема, а не тільки про господарів. Також потенційно можна дозволити оренду спортивного інвентаря і на строк від 1 години, це має бути передбачено під час проєктування і розробки. У цих аналогів же мінімальним строком є доба.

Таблиця 1.1 – Результати аналізу ПЗ оренди спортивного інвентаря

Критерій	Checkyeti	FriendWithA	Моя програма
Джерело оренди спортивного інвентаря	Пункти прокату	Господарі	Господарі, можливо пункти прокату
Відгуки про господарів/пункти оренди спортивного інвентаря	—	+	+
Відгуки про сам спортивний інвентар	—	—	+
Бронювання оренди спортивного інвентаря	+	+	+
Створення закладок	—	+	+
Мінімальний строк оренди	День	День	Година

1.3 Висновки за розділом 1 та постановка завдань роботи

Проведено аналіз оренди спортивного інвентаря, розглянуто як аналоги розробки Checkyeti і FriendWithA, виконано порівняння їх між собою та запропоновано удосконалення існуючих засобів у програмі, яку заплановано розробити, для чого визначено завдання:

- аналіз програмних аналогів і визначення концепції програми;
- проєктування ПЗ оренди спортивного інвентаря;
- реалізація вебзастосунку оренди спортивного інвентаря.

2 РОЗРОБКА АРХІТЕКТУРИ ПРОГРАМИ

2.1 Вибір технологій розробки

Технології веброзробки було проаналізовано в ширшому ракурсі, а для детального порівняння виокремлено мови програмування Python [9] і Ruby [10] (табл. 2.1).

Подальшу реалізацію вирішено виконувати мовою Python, врахувавши, що вона має більше готових бібліотек, що дозволяє менше створювати нового коду, а більше використовувати готового, відлагодженого коду для створення нових рішень, більше того Python має більше засобів обробки даних і меншу складність навчання, тобто будь-які зміни долаються простіше в підсумку.

Таблиця 2.1 – Порівняння мов програмування для реалізації ПЗ

Критерій	Python	Ruby
Тип мови програмування	Високорівнева	Високорівнева
Призначення мови програмування	Загальне	Загальне
Виконання веброзробки	Так	Так
Парадигми програмування	Мультипарадигмальна	Мультипарадигмальна
Доступність готових бібліотек	Широка	Порівняно менша
Засоби обробки даних	Високо поширені	Менша поширеність
Складність навчання	Низька	Вища

2.2 Моделювання вимог до програми

Гість (рис. 2.1) є користувачем, який не має або ще не використав свій обліковий запис для роботи з програмою.

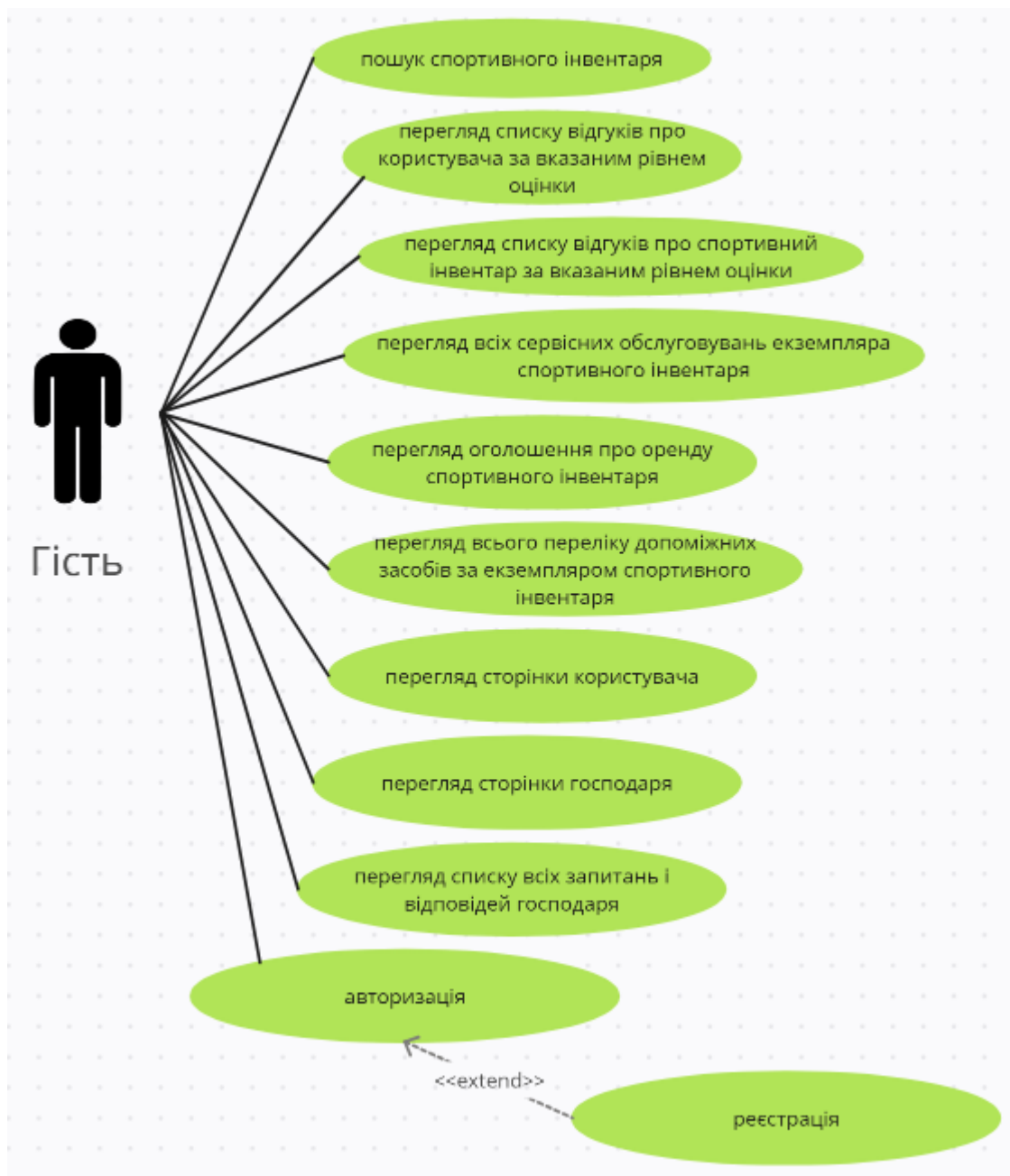


Рисунок 2.1 – Діаграма прецедентів гостя

Гість має доступ до такого набору сценаріїв:

– пошук спортивного інвентаря;

- перегляд списку відгуків про користувача за вказаним рівнем оцінки;
- перегляд списку відгуків про спортивний інвентар за вказаним рівнем оцінки;
- перегляд всіх сервісних обслуговувань екземпляра спортивного інвентаря;
- перегляд оголошення про оренду спортивного інвентаря;
- перегляд всього переліку допоміжних засобів за екземпляром спортивного інвентаря;
- перегляд сторінки користувача;
- перегляд сторінки господаря;
- перегляд списку всіх запитань і відповідей господаря;
- авторизація.

Гостю доступний тільки один альтернативний сценарій – це реєстрація, який застосовується до авторизації в тому випадку, коли гість не має облікового запису взагалі.

Якщо користувач авторизувався, то він може виконувати одну з двох ролей. Ролью, в якій він може додавати власні оголошення про оренду спортивного інвентаря, є господар. Господар має доступ до такого набору сценаріїв:

- створення відгуку про користувача, який орендував спортивний інвентар;
- перегляд отриманих сум за оренду протягом інтервалу часу;
- створення оголошення про оренду спортивного інвентаря;
- перегляд всіх своїх оголошень про оренду спортивного інвентаря;
- створення відповіді на запитання;
- перегляд отриманих запитань;
- визначення базової вартості спортивного інвентаря;
- визначення базової вартості допоміжного засобу;
- визначення сезонності для спортивного інвентаря господаря;

- визначення коефіцієнтів сезонності для спортивного інвентаря;
- визначення коефіцієнтів сезонності для допоміжного засобу;
- перегляд очікуваних оренд всього спортивного інвентаря господаря;
- перегляд календаря оренд екземпляру спортивного інвентаря господаря;

- перегляд історії оренд всього спортивного інвентаря господаря;
- визначення періоду ремонту екземпляру спортивного інвентаря;
- визначення сервісного обслуговування спортивного інвентаря;
- визначення політики відміни бронювань;
- визначення допоміжних засобів для оренди спортивного інвентаря.

Альтернативними сценаріями для господаря є:

- внесення змін до оголошення про оренду спортивного інвентаря, що реалізується альтернативно створенню оголошення про оренду спортивного інвентаря у випадку, якщо інформація у оголошенні не відповідає дійсності;

- коригування коефіцієнтів сезонності для спортивного інвентаря реалізується альтернативно визначенню коефіцієнтів сезонності для спортивного інвентаря у випадку, якщо необхідні значення більше не відповідають дійсності;

- коригування коефіцієнтів сезонності для допоміжного засобу реалізується альтернативно визначенню коефіцієнтів сезонності для допоміжного засобу у випадку, якщо необхідні значення більше не відповідають дійсності;

- коригування періоду ремонту спортивного інвентаря реалізується альтернативно визначенню періоду ремонту екземпляру спортивного інвентаря, якщо було внесено некоректні дані;

- коригування політики відміни бронювань реалізується альтернативно визначенню політики відміни бронювань у випадку, якщо господар вирішив змінити застосовувані правила для обслуговування бронювань (рис. 2.2).

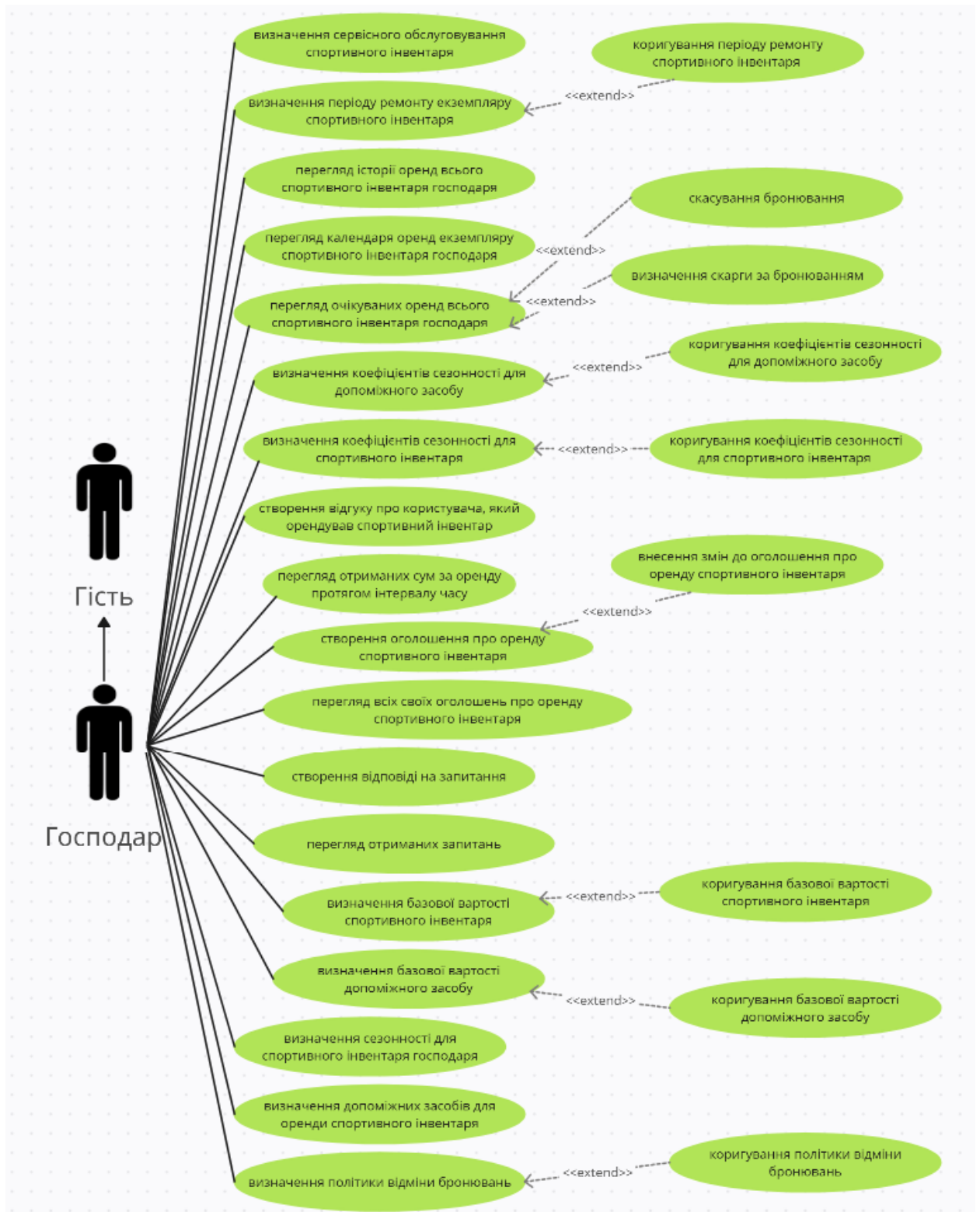


Рисунок 2.2 – Діаграма прецедентів господаря

Також альтернативними сценаріями для господаря є:

– скасування бронювання реалізується альтернативно перегляду очікуваних оренд всього спортивного інвентаря господаря у тому випадку,

якщо господар не має довіри до користувача, який забронював оренду його спортивного інвентаря, з тих чи інших причин або якщо він не зможе забезпечити виконання цієї оренди;

- визначення скарги за бронюванням реалізується альтернативно перегляду очікуваних оренд всього спортивного інвентаря господаря у тому випадку, якщо господар не отримав назад свій інвентар, отримав його невчасно або не втому стані, в якому надавав його в оренду;

- коригування базової вартості спортивного інвентаря реалізується альтернативно визначенню базової вартості спортивного інвентаря у випадку, якщо господар бажає змінити загальні умови оренди;

- коригування базової вартості допоміжного засобу реалізується альтернативно визначенню базової вартості допоміжного засобу у випадку, якщо господар бажає змінити загальні умови оренди.

Авторизований користувач, який не потребує виставлення власного спортивного інвентаря в оренду, має доступ до такого набору сценаріїв:

- створення відгуку про спортивний інвентар, що був у оренді;
- бронювання оренди спортивного інвентаря;
- перегляд закладок оголошень про оренду спортивного інвентаря;
- створення запитання до господаря;
- перегляд своїх запитань до господарів;
- створення закладки на оголошення про оренду спортивного інвентаря;
- визначення доступу до власного бронювання іншому користувачу;
- перегляд власних бронювань оренди спортивного інвентаря;
- перегляд відкритих користувачу бронювань.

Альтернативними сценаріями для авторизованого користувача є:

- видалення створеної закладки на оголошення про оренду спортивного інвентаря реалізується альтернативно створенню закладки на оголошення про оренду спортивного інвентаря у випадку, якщо

авторизований користувач створив закладку помилково або більше не бажає повертатися до цього оголошення;

- визначення скарги за бронюванням реалізується альтернативно бронюванню оренди спортивного інвентаря у випадку, якщо користувач не отримав у оренду спортивний інвентар, отримав його невчасно або не в тому стані, в якому очікував, або не зміг його повернути через провину господаря;
- скасування бронювання реалізується альтернативно бронюванню оренди спортивного інвентаря у випадку, якщо користувач змінив плани і більше не може або не бажає виконати оренду вказаним чином (рис. 2.3).

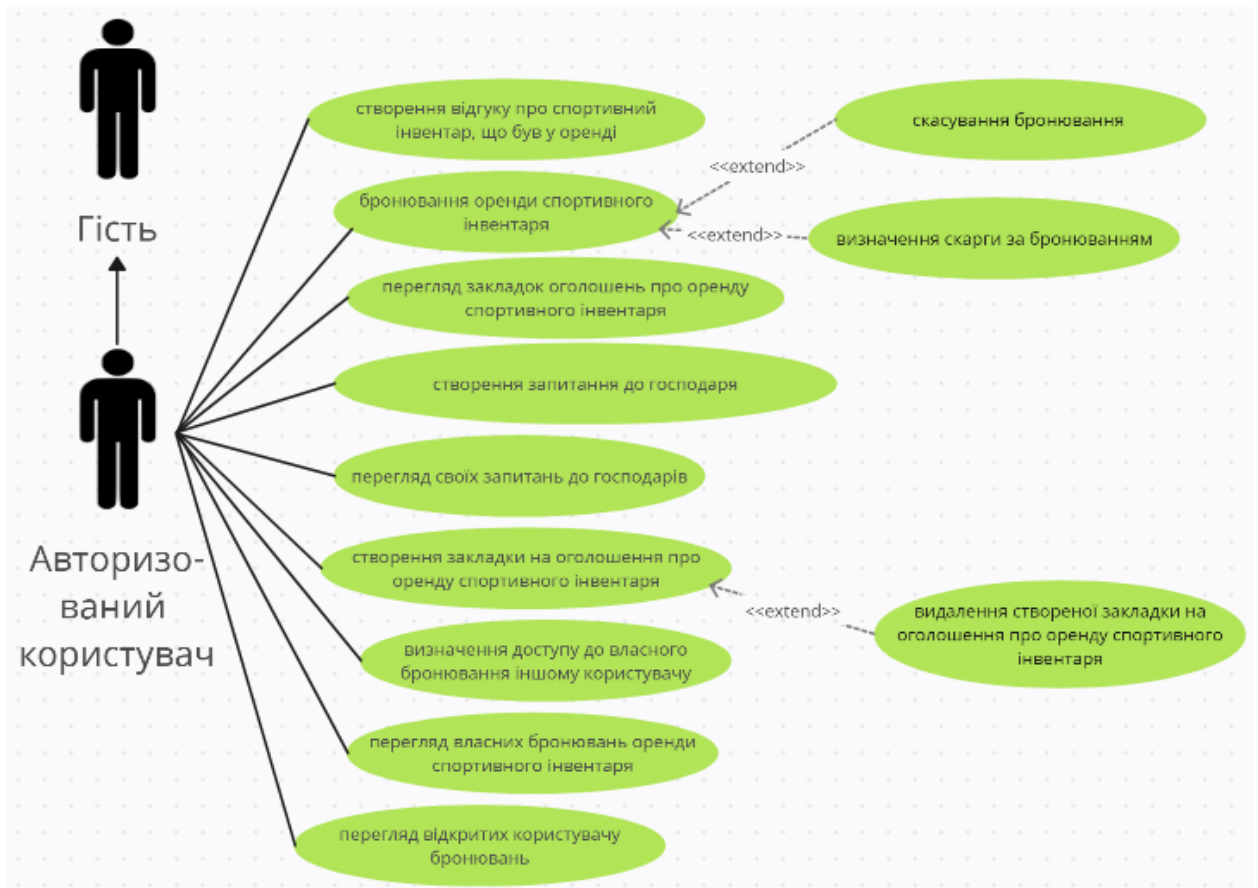


Рисунок 2.3 – Діаграма прецедентів авторизованого користувача

2.3 Проектування бази даних

База даних (БД) у програмі має застосовуватися для підтримки сценаріїв, які були змодельовані у підрозділі 2.2, відповідно потрібно

враховувати основні сутності, які при цьому виявляються.

Наявність спортивного інвентаря в програмі потребує деякого впорядкування. Першим чинником впорядкування є види спорту, адже на результуючому вебсайті може бути представлений різний інвентар, а не маючи такого розподілу, важко буде знайти те, що потрібно користувачу. Окрім того це важливим маркер того, хто саме і що саме потенційно може шукати на вебсайті.

Види спорту каталогізовані в таблиці Sport з полями id для визначення ідентифікатора виду спорту та name для назви виду спорту.

Будь-який екземпляр інвентаря має свої власні характеристики і звичайно, що мова йде не про загальний спортивний інвентар, а саме про конкретні екземпляри. А конкретні екземпляри мають різних виробників. Орендується не загальний сноуборд, а сноуборд конкретного виробника, особливо для інвентаря, який коштує не мінімальну суму, це може бути важливою характеристикою.

Виробники спортивного інвентаря каталогізовані в таблиці Brand з полями id для визначення ідентифікатора виробника спортивного інвентаря та name для назви виробника спортивного інвентаря.

Увесь спортивний інвентар поділений на категорії. Категорії – це фактично як раз певні узагальнення, які не мають виробників, але вже виражають певну сутність спортивного інвентаря. Наприклад, категорією є сноуборд.

Категорії спортивного інвентаря каталогізовані в таблиці SportequipmentCategory з полями id для визначення ідентифікатора категорії спортивного інвентаря, name для назви категорії спортивного інвентаря та sport для зв'язування з таблицею Sport, щоб визначити, до якого виду спорту належить ця категорія.

Однак далі спортивний інвентар може певним чином ще розділятися, ще не вводячи конкретного виробника для певного екземпляра. Наприклад,

якщо спортивним інвентарем є велосипед, то гірський, шосейний велосипед – це і є тип категорії спортивного інвентаря.

Типи категорій спортивного інвентаря каталогізовані в таблиці `SportequipmentCategoryType`, яка включає поля:

- `id` – ідентифікатор типу категорій спортивного інвентаря;
- `sportequipmentcategory` – категорія спортивного інвентаря, до якої належить цей тип категорії: визначає зв'язок з таблицею `SportequipmentCategory`, яка може мати декілька записів у цій таблиці, кожен з яких визначає конкретний тип конкретної категорії спортивного інвентаря;
- `name` – назва типу категорії спортивного інвентаря.

Сам спортивний інвентар, вже представлений конкретними екземплярами, надається господарями в оренду. Тобто це вже найнижчий рівень, на якому мова йде про конкретний фізичний об'єкт – конкретний сноуборд, велосипед або штангу. Такі дані про спортивний інвентар представлені в таблиці `Sportequipment`, яка включає поля:

- `id` – ідентифікатор спортивного інвентаря (конкретного екземпляру фактично, який в подальшому називається коротко – спортивний інвентар);
- `host` – господар, який володіє цим спортивним інвентарем: визначає зв'язок з таблицею `Host`, яка може мати декілька записів у цій таблиці, кожен з яких визначає конкретний екземпляр спортивного інвентаря, яким володає господар;
- `brand` – виробник цього екземпляру спортивного інвентаря: визначає зв'язок з таблицею `Brand`, яка може мати декілька записів у цій таблиці, кожен з яких визначає конкретний екземпляр спортивного інвентаря, який виготовлений цим виробником;
- `sportequipmentcategorytype` – тип категорії, яку представляє цей екземпляр спортивного інвентаря: визначає зв'язок з таблицею `SportequipmentCategoryType`, яка може мати декілька записів у цій таблиці, кожен з яких визначає конкретний екземпляр цього типу категорії спортивного інвентаря;

- title – назва, яка використовується для загального текстового представлення цього екземпляра спортивного інвентаря;
- fulldescription – повна інформація про цей екземпляр спортивного інвентаря: сам господар визначає, що саме слід внести в це поле;
- review_mark – оцінка, яку було обчислено за отриманими відгуками на цей екземпляр спортивного інвентаря через усереднення оцінок у них;
- address – адреса місця, де знаходиться цей екземпляр спортивного інвентаря, визначена текстовими даними;
- address_x – координата широти місця, де знаходиться цей екземпляр спортивного інвентаря, визначена текстовими даними;
- address_y – координата довготи місця, де знаходиться цей екземпляр спортивного інвентаря, визначена текстовими даними;
- closed – оголошення закрито, бронювання за ним здійснюватися не може (якщо значення True).

Кожен екземпляр спортивного обладнання може мати не одну, а декілька фотографій, хоча в найгіршому випадку їх може бути 0. Відповідно зберігати фотографії потрібно окремо від самого екземпляру спортивного обладнання. Для цього було виділено таблицю `SportequipmentPhoto`, яка включає поля:

- id – ідентифікатор фотографії спортивного інвентаря;
- sportequipment – спортивний інвентар, який має цю фотографію: визначає зв'язок з таблицею `Sportequipment`, яка може мати декілька фотографій у цій таблиці, кожна з яких визначає цей конкретний екземпляр спортивного інвентаря;
- photo – безпосередньо фотографія спортивного обладнання.

Спортивне обладнання може мати певні допоміжні засоби. Наприклад, до велосипеда необхідний шолом. Але фактично окремо орендувати шолом з одного боку недоцільно, а з іншого – незручно. Тому фактично ці допоміжні засоби існують в певній зв'язці зі спортивним обладнанням і не представляються у вебдодатку в результаті як окреме

оголошення. Через це немає необхідності зберігати ці допоміжні засоби в тій самій таблиці, що і спортивний інвентар. Відповідно допоміжні засоби спортивного інвентаря в загальному вигляді (або той самий шолом) представлені в таблиці `Helper`, яка включає поля:

- `id` – ідентифікатор допоміжного засобу для спортивного інвентаря;
- `sportequipmentcategory` – категорія спортивного інвентаря, яку представляє цей допоміжний засіб: визначає зв'язок з таблицею `SportequipmentCategory`, яка може мати декілька записів у цій таблиці, кожен з яких визначає конкретний допоміжний засіб;
- `host` – господар, який володіє цим допоміжним засобом: визначає зв'язок з таблицею `Host`, яка може мати декілька записів у цій таблиці, кожен з яких визначає конкретний набір (екземплярів фактично може бути декілька) допоміжних засобів, який має цей господар;
- `photo` – фотографія допоміжного засобу: достатньо однієї фотографії для ілюстрації;
- `description` – інформація про цей допоміжний засіб;
- `quantity` – кількість наявних екземплярів: допоміжні засоби є дрібними елементами, тому немає потреби фіксувати, який саме екземпляр кому саме має бути переданий, це не дуже важливо під час бронювання оренди, важливо отримати захисний шолом, а не те, щоб він був конкретним шоломом з 5 доступних.

Допоміжні засоби не важливі самі по собі, вони надаються на додачу до спортивного інвентаря, тобто користувач замовляє спортивний інвентар. А вже під нього отримує певні допоміжні засоби. Тому для кожного екземпляру допоміжного засобу користувач має встановити відповідний допоміжний засіб, який з ним можна і який може бути доцільно з ним замовити. Дані про відповідність допоміжних засобів екземплярам спортивного інвентаря знаходяться в таблиці `SportequipmentHelper`, яка включає поля:

- `id` – ідентифікатор зв'язування між допоміжним засобом і спортивним інвентарем;

- `helper` – допоміжний засіб, який відповідає цьому спортивному інвентарю: визначає зв'язок з таблицею `Helper`, яка може мати декілька записів у цій таблиці, кожен з яких визначає конкретне зв'язування;

- `sportequipment` – спортивний інвентар, якому відповідає цей допоміжний засіб: визначає зв'язок з таблицею `Sportequipment`, яка може мати декілька записів у цій таблиці, кожен з яких визначає конкретне зв'язування.

Фактично таблиця `SportequipmentHelper` необхідна для реалізації зв'язку, який має тип багато-до-багатьох між відповідно таблицями `Sportequipment` і `Helper`.

Бронювання оренди спортивного інвентаря є однією з головних функцій цієї програми, а основні характеристики бронювання знаходяться в таблиці `Booking`, яка включає поля:

- `id` – ідентифікатор бронювання оренди спортивного інвентаря;

- `sportequipment` – спортивний інвентар, який заброньовано в цьому бронюванні: визначає зв'язок з таблицею `Sportequipment`, яка може мати декілька записів у цій таблиці, кожен з яких визначає конкретне бронювання цього екземпляра спортивного інвентаря;

- `status` – стан бронювання: бронювання не вважається одразу виконаним, тобто воно не передбачає одразу обов'язкове виконання оренди, адже воно може бути скасовано, може щось трапитись, що фактично не зможуть зустрітись обидві сторони, тому саме через стан бронювання і відслідковується те, що сталося з бронюванням;

- `user` – користувач, який забронював цей спортивний інвентар: визначає зв'язок з таблицею `User`, яка може мати декілька записів у цій таблиці, кожен з яких визначає конкретне бронювання цього користувача;

- `price` – вартість оренди спортивного інвентаря за вказаних параметрів;

- `paid` – сума, яку за це бронювання було сплачено;

- account – рахунок, з якого було сплачено ці кошти за бронювання;
- booked_date – дата, коли заброньовано цей спортивний інвентар;
- start_date – дата, з якої починається оренда спортивного інвентаря;
- end_date – дата, до якої триває оренда спортивного інвентаря;
- phone – номер телефону для підтримки користувача;

Користувач, який створив бронювання може надати інформацію про нього іншому користувачу. Це може бути корисним, наприклад, тоді, коли друзі разом подорожують і орендують спортивний інвентар разом, або в подібних ситуаціях. Таке надання інформації зберігається в таблиці BookingShare, яка включає поля:

- id – ідентифікатор надання інформації про бронювання користувачем, який створив це бронювання, іншому користувачу;
- booking – бронювання, яким поділився користувач, який його створив: визначає зв'язок з таблицею Booking, яка може мати декілька записів у цій таблиці, кожен з яких визначає факт того, що ним поділився користувач, який його створив, з іншим користувачем, а таких користувачів може бути в тому числі і декілька;
- user – користувач, якому надано доступ до даних бронювання.

Внесені до бронювання допоміжні засоби знаходяться в таблиці BookingHelper, яка включає поля:

- id – ідентифікатор доданої позиції з допоміжним засобом до бронювання;
- booking – бронювання, до якого входить цей допоміжний засіб: визначає зв'язок з таблицею Booking, яка може мати декілька записів у цій таблиці, кожен з яких визначає окремий допоміжний засіб, доданий до бронювання;
- helper – конкретний допоміжний засіб, доданий до бронювання: визначає зв'язок з таблицею Helper, яка може мати декілька записів у цій таблиці, кожен з яких визначає окреме бронювання, в яке допоміжний засіб з

цього набору (адже в вихідних даних це може бути декілька екземплярів, які розглядаються як повністю однакові) був доданий;

- quantity – кількість екземплярів цього допоміжного засобу, який було заброньовано.

Політики відміни бронювань визначаються в таблиці Cancellation, яка включає поля:

- id – ідентифікатор політики відміни бронювання;
- name – назва політики відміни бронювання;
- days – кількість днів, за яку до настання початку оренди можна скористатися цією політикою;
- percent – відсоток з суми бронювання, який повертається в такому випадку користувачу, який створив бронювання.

Застосування конкретних політик відміни бронювання до конкретного екземпляру спортивного інвентаря зберігається в таблиці SportequipmentCancellation, яка включає поля:

- id – ідентифікатор застосування політики відміни до бронювання;
- sportequipment – спортивний інвентар, до якого застосовується ця політика відміни: визначає зв'язок з таблицею Sportequipment, яка може мати декілька записів у цій таблиці, кожен з яких визначає окрему політику відміни, що може застосовуватися до бронювання цього спортивного інвентаря;
- cancellation – політика відміни бронювання, яка застосовується: визначає зв'язок з таблицею Cancellation, яка може мати декілька записів у цій таблиці, кожен з яких визначає окреме застосування політики відміни для конкретного спортивного інвентаря.

Дані про доставку фіксуються для зручності господаря. Якщо він використовує роботу служб доставки, то в цій таблиці може бути збережено дані про те, куди бажає доставити своє замовлення користувач. Це не передбачає визначення наявних служб доставки або подібної інформації. Користувач має в такому випадку лише чітко зазначити, куди і

використовуючи яку службу доставки має доставити господар це бронювання для реалізації оренди. Відповідно така доставка може і не вимагатися в конкретному випадку. Або взагалі бути забороненою фактично з боку господаря. Тому це потребує виділення окремої таблиці Delivery, хоча її зв'язок з бронюванням в загальному випадку один-до-одного. Поля таблиці:

- id – ідентифікатор замовлення доставки за бронюванням;
- booking – бронювання, до якого відноситься ця доставка: визначає зв'язок з таблицею Booking;
- service – назва служби доставки, яка має бути використана;
- department – відділення служби доставки, на яку має бути відправлено спортивний інвентар.

Користувачі програми разом зі своїми обліковими записами зберігаються в таблиці User, яка включає поля:

- id – ідентифікатор користувача в програмі;
- login – логін, який в цьому обліковому запису застосовується;
- name – ім'я користувача (те, яке використовується в житті);
- password – пароль користувача;
- review_mark – оцінка, яку було обчислено за отриманими відгуками щодо цього користувача через усереднення оцінок у них.

Дані господарів спортивного інвентаря зберігаються в таблиці Host, яка включає поля:

- id – ідентифікатор господаря;
- login – логін господаря;
- name – ім'я користувача (те, яке використовується в житті);
- password – пароль користувача;
- review_mark – оцінка, яку було обчислено за отриманими відгуками щодо всіх екземплярів спортивного інвентаря цього користувача через усереднення оцінок у них;
- phone – номер телефону для підтримки господаря;

– account – банківський рахунок, на який мають переводитися кошти господарю за оренду його спортивного інвентаря;

– delivery_used – можливість використання служб доставки для даного господаря;

– last_visit – дата останньої авторизації господаря у програмі.

Підхід до ціноутворення в програмі базується на сезонності. Тобто під час різних сезонів вартість оренди спортивного інвентаря може бути різною. Взимку велосипеди однозначно непопулярні, тому якщо на вулиці, наприклад, не сніг, можна спробувати стимулювати оренду велосипедів зниженням вартості. Це звичайно не призведе до літніх масштабів оренди, але це загалом є логічним економічним станом.

Тож у програмі визначається декілька типів сезонів, а далі вони мають бути співставлені з конкретними датами, адже зима в різних країнах, наприклад, може починатися в різний час. Відповідно і для різного обладнання один сезон може бути найпопулярнішим, а для іншого – навпаки. Наприклад, сноуборди взимку популярні, а велосипеди – ні. А взимку ситуація змінюється навпаки.

Типи сезонів визначають фактично назви сезонів для представлення в програмі. Типи сезонів знаходяться в таблиці Seasontype, яка включає поля:

– id – ідентифікатор типу сезону;

– title – назва типу сезону.

Відповідно сам конкретний сезон застосовується вже у термінах категорій спортивного інвентаря. Тобто він визначається господарем стосовно певної групи спортивного інвентаря, який входить до однієї категорії. Для всієї такої групи встановлюється один і той самий проміжок дат. Тобто таким чином фактично визначається в термінах цінових пропозицій початок і закінчення зими, наприклад. Відповідні ці дані у вигляді визначення сезону знаходяться в таблиці Season, яка включає поля:

– id – ідентифікатор сезону;

– `seasontype` – тип сезону, до якого відноситься цей сезон: визначає зв'язок з таблицею `Seasontype`, яка може мати декілька записів у цій таблиці, кожен з яких визначає окремий варіант визначення цього типу сезону за проміжком його дат;

– `sportequipmentcategory` – категорія спортивного інвентаря, для якої визначено цей тип сезону: визначає зв'язок з таблицею `SportequipmentCategory`, яка може мати декілька записів у цій таблиці, кожен з яких визначає окремий сезон для цієї категорії спортивного обладнання;

– `host` – господар, який визначив ці правила для свого спортивного інвентаря: визначає зв'язок з таблицею `Host`, яка може мати декілька записів у цій таблиці, кожен з яких визначає окремий сезон для окремої категорії спортивного інвентаря;

– `start_date` – дата початку дії цього сезону;

– `end_date` – дата закінчення дії цього сезону.

Відповідно система ціноутворення в програмі передбачає, що після визначення періоду, до якого належить кожен сезон, потрібно встановити базову вартість оренди для кожного спортивного інвентаря за певної тривалості такого орендування. А потім для кожного сезону зазначається певний коефіцієнт, який цю базову вартість або пропорційно збільшує для всіх ситуацій, або пропорційно зменшує. Це дозволяє в підсумку формувати певні загальні правила, а далі визначати їх особливості, зменшуючи обсяг роботи для господаря.

Базова вартість оренди спортивного інвентаря визначається в таблиці `BasePrice`, яка включає поля:

– `id` – ідентифікатор запису про базову вартість оренди деякого спортивного інвентаря;

– `sportequipment` – спортивний інвентар, для якого визначена ця базова вартість: визначає зв'язок з таблицею `Sportequipment`, яка може мати декілька записів у цій таблиці, кожен з яких визначає базову вартість оренди цього спортивного інвентаря за певних умов;

- unit – одиниця вимірювання тривалості оренди (місяць, день тощо);
- quantity – мінімальна кількість одиниць часу, за якої застосовується ця базова вартість;

- price – значення вартості цієї одиниці оренди.

Дані про сезонні коефіцієнти, які застосовуються господарем для свого спортивного інвентаря, знаходяться в таблиці SeasonCoefficient, яка включає поля:

- id – ідентифікатор сезонного коефіцієнту;
- sportequipment – спортивний інвентар, якого стосується цей коефіцієнт: визначає зв'язок з таблицею Sportequipment, яка може мати декілька записів у цій таблиці, кожен з яких визначає коефіцієнт для конкретного типу сезону;
- seasontype – тип сезону, за якого для даного спортивного інвентаря діє цей коефіцієнт: визначає зв'язок з таблицею Seasontype, яка може мати декілька записів у цій таблиці, кожен з яких визначає коефіцієнт, який застосовується для різного спортивного інвентаря за цього типу сезону;
- coefficient – значення коефіцієнта, яке має бути застосовано за таких умов.

Для допоміжних засобів також використовується подібна логіка ціноутворення, але вона є дещо спрощеною. Ця спрощеність визначає, що коефіцієнти застосовуються не для конкретного екземпляру спортивного інвентаря, а для категорії спортивного інвентаря загалом.

Базова вартість оренди допоміжного засобу визначається в таблиці HelperBasePrice, яка має поля:

- id – ідентифікатор базової вартості допоміжного засобу;
- helper – допоміжний засіб, для якого застосовується ця базова вартість: визначає зв'язок з таблицею Helper, яка може мати декілька записів у цій таблиці, кожен з яких визначає коефіцієнт, який застосовується для різного спортивного інвентаря за різних параметрів оренди;
- unit – одиниця тривалості (день, тиждень, місяць);

– quantity – мінімальна кількість одиниць тривалості оренди, за якої застосовується ця вартість;

– price – вартість однієї одиниці тривалості оренди за цих умов.

Дані про сезонні коефіцієнти, які застосовуються господарем для своїх допоміжних засобів спортивного інвентаря, знаходяться в таблиці HelperSeasonCoefficient, яка включає поля:

– id – ідентифікатор сезонного коефіцієнту допоміжного засобу;

– sportequipmentcategory – категорія спортивного інвентаря, якої стосується цей коефіцієнт: визначає зв'язок з таблицею SportequipmentCategory, яка може мати декілька записів у цій таблиці, кожен з яких визначає коефіцієнт для конкретного типу сезону;

– seasontype – тип сезону, за якого для даного допоміжного засобу діє цей коефіцієнт: визначає зв'язок з таблицею Seasontype, яка може мати декілька записів у цій таблиці, кожен з яких визначає коефіцієнт, який застосовується для різного допоміжного засобу за цього типу сезону;

– coefficient – значення коефіцієнта, яке має бути застосовано за таких умов.

За спортивним інвентарем може визначатися його обслуговування, щоб продемонструвати його стан, підтвердити його певними фактами. Відповідні дані знаходяться в таблиці Service, яка включає поля:

– id – ідентифікатор виконаного обслуговування спортивного інвентаря;

– sportequipment – спортивний інвентар, обслуговування якого виконувалось: визначає зв'язок з таблицею Sportequipment, яка може мати декілька записів у цій таблиці, кожен з яких визначає конкретне обслуговування цього спортивного інвентаря;

– service_date – дата обслуговування;

– detail – опис тих сервісних дій або загалом обслуговування, яке замовлялось;

– document – документ, який підтверджує факт робіт.

Часом спортивний інвентар може знаходитися в ремонті, на обслуговування або просто господар не працює в ці дні. Всі такі випадки, тобто ситуації, коли спортивний інвентар недоступний для оренди, визначаються в таблиці Repair, яка включає поля:

- id – ідентифікатор ремонту (недоступності спортивного інвентаря);
- sportequipment – спортивний інвентар, ремонт якого виконувався або буде виконуватися: визначає зв'язок з таблицею Sportequipment, яка може мати декілька записів у цій таблиці, кожен з яких визначає конкретний ремонт цього спортивного інвентаря;
- start_date – дата початку ремонту;
- end_date – дата закінчення ремонту.

Відгуки про користувачів знаходяться в таблиці ReviewUser, яка включає поля:

- id – ідентифікатор відгуку про користувача;
- user – користувач, про якого цей відгук: визначає зв'язок з таблицею User, яка може мати декілька записів у цій таблиці, кожен з яких визначає конкретний відгук про цього користувача;
- host – господар, який написав цей відгук: визначає зв'язок з таблицею Host, яка може мати декілька записів у цій таблиці, кожен з яких визначає конкретний відгук про різних користувачів;
- mark – оцінка;
- review_date – дата створення відгуку;
- review_message – текст відгуку.

Відгуки про спортивний інвентар знаходяться в таблиці ReviewEquipment, яка включає поля:

- id – ідентифікатор відгуку про спортивний інвентар;
- user – користувач, який створив цей відгук: визначає зв'язок з таблицею User, яка може мати декілька записів у цій таблиці, кожен з яких визначає конкретний відгук про різний спортивний інвентар;

- sportequipment – спортивний інвентар, відгук про який створено: визначає зв'язок з таблицею Sportequipment, яка може мати декілька записів у цій таблиці, кожен з яких визначає конкретний відгук від різних господарів про цей спортивний інвентар;
- mark – оцінка;
- review_date – дата створення відгуку;
- review_message – текст відгуку (рис. 2.4).

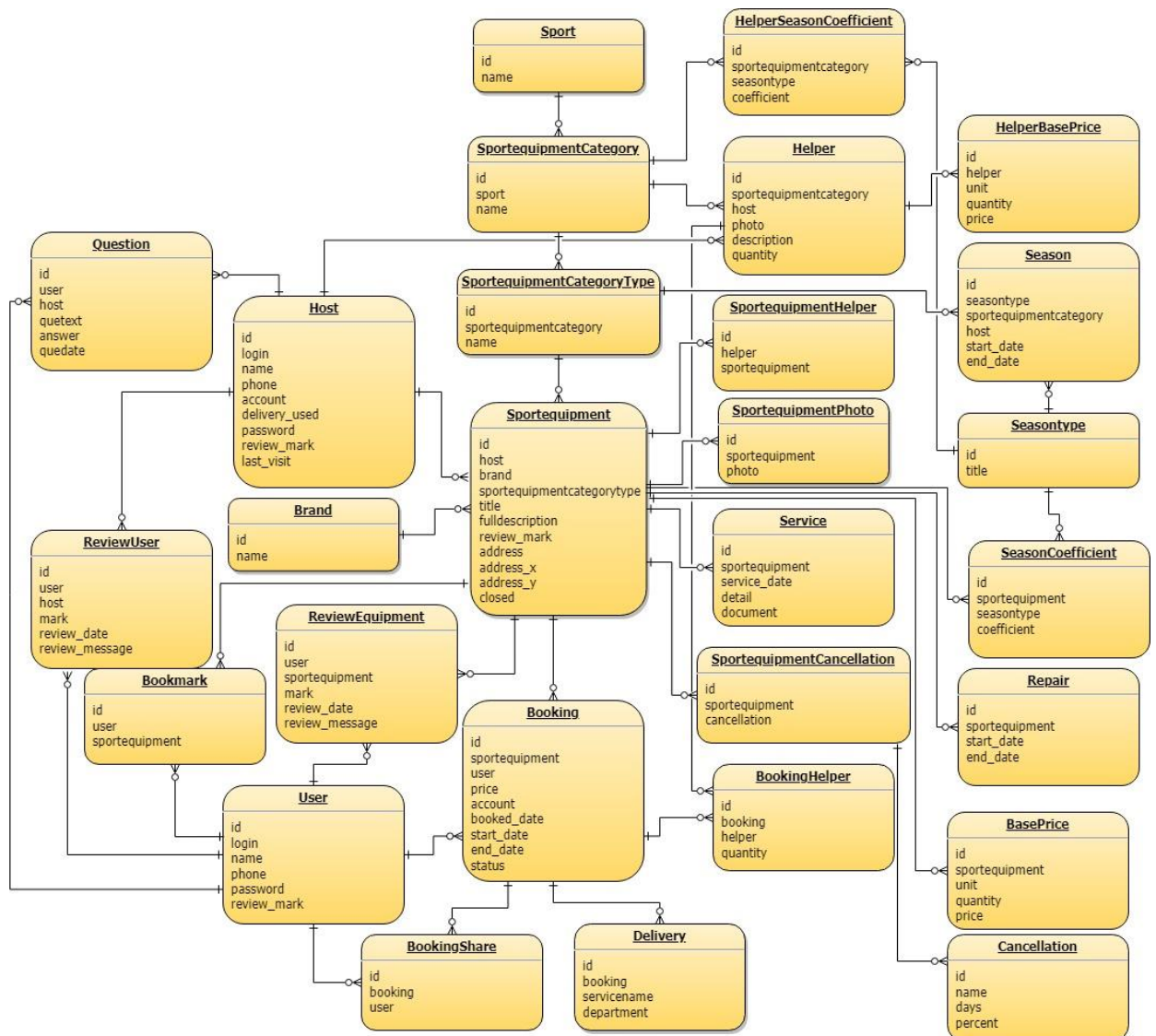


Рисунок 2.4 – Схема БД

Користувачі можуть створювати закладки, щоб зберегти і повернутися до певного спортивного інвентаря з часом. Відповідні закладки

зберігаються в таблиці Bookmark, яка включає поля:

- id – ідентифікатор закладки;
- user – користувач, що створив закладку: визначає зв'язок з таблицею User, яка може мати декілька записів у цій таблиці, кожен з яких визначає конкретну закладку стосовно різного спортивного інвентаря;
- sportequipment – спортивний інвентар, який збережено в цій закладці: визначає зв'язок з таблицею Sportequipment, яка може мати декілька записів у цій таблиці, кожен з яких визначає конкретну закладку від конкретного користувача, а користувачі можуть бути різними.

Також користувачі можуть з господарями обмінюватися запитаннями і відповідями. Для цього створена таблиця Question, яка включає поля:

- id – ідентифікатор запитання;
- user – користувач, який створив запитання або якому можливо воно адресовано;
- host – господар, якому адресовано запитання або який можливо створив його;
- quetext – текст запитання;
- answer – текст відповіді;
- quedate – дата створення запитання.

2.4 Висновки за розділом 2

У другому розділі розв'язано питання проектування ПЗ, охоплюючи вибір мови python для реалізації програми, MySQL для підтримки роботи з БД, виділення сутностей предметної області та зв'язків між ними як структури БД.

3 РЕАЛІЗАЦІЯ ПРОГРАМИ

3.1 Схема функціонування програми

Загальна послідовність роботи з програмою приведена на рис. 3.1.

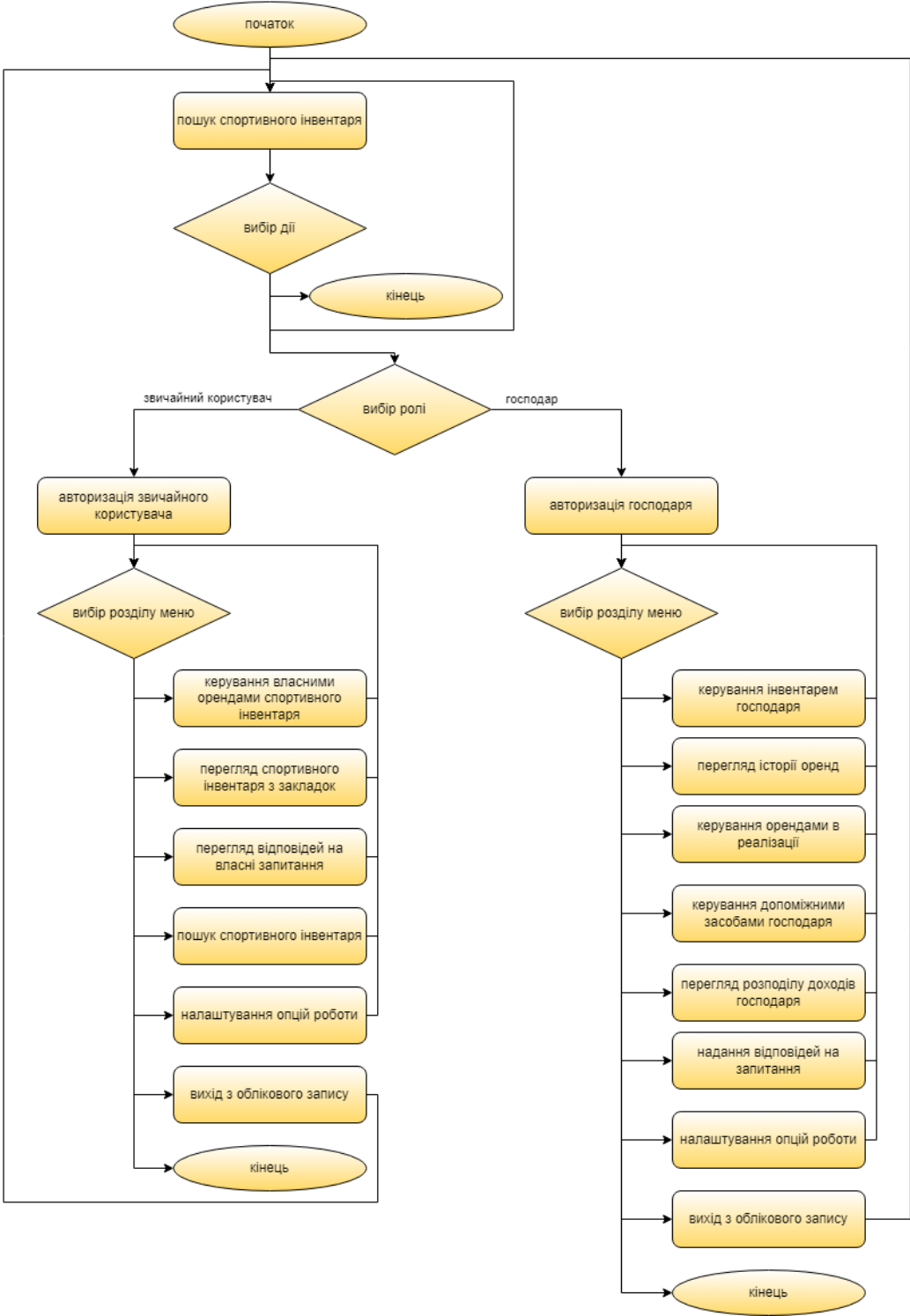


Рисунок 3.1 – Схема функціонування програми

3.2 Програмна реалізація функцій вебсистеми

Реалізацію програми виконано. Потрібні функції створені. Реалізацію забезпечено з використанням фреймворку django. Функції програми у реалізованому коді представлені нижче. Для створення макетів вебсторінок застосовано шаблон [11].

У подальшому описі акцент зроблено на функціях представлень, адже саме від них залежить витягання даних з БД і внесення їх до шаблонів сторінок, тобто вони визначаються користувацькі функції, які потрібно було реалізувати.

Функція `sportequipment_search` реалізує пошук спортивного інвентаря. Дана функція для виконання пошуку містить ряд параметрів:

- список можливих виробників спортивного інвентаря у параметрі `brands`;
- список можливих видів спорту, для яких призначений спортивний інвентар, у параметрі `sports`;
- список категорій спортивного інвентаря у параметрі `categories`;
- список типів категорій спортивного інвентаря у параметрі `categorytypes`;
- адресу пошуку через її широту в параметрі `addx` і довготу в параметрі `addy`, а також радіус пошуку в метрах у параметрі `radius`;
- мінімальну середню оцінку за відгуками орендарів у параметрі `reviewlevel`;
- мінімальну вартість оренди спортивного інвентаря у параметрі `priceFrom`;
- максимальну вартість оренди спортивного інвентаря у параметрі `priceTo`;
- дату початку оренди спортивного інвентаря у параметрі `dateFrom`;
- дату завершення оренди спортивного інвентаря у параметрі `dateTo`.

Під час реалізації пошуку відбувається відбір спортивного інвентаря, що відповідає вказаним параметром. Серед параметрів присутня координата, яка визначає адресу пошуку. Для неї відбувається перевірка відстані між адресою, вказаною для спортивного інвентаря та в параметрах пошуку. Якщо вона менше ніж задана у параметрах пошуку відстань, то цей екземпляр залишається в результатах.

Окремим етапом визначається відбір того інвентаря, який відповідає датам і вартості. Для цього спочатку за кожним раніше відібраним інструментом перевіряється, чи вказано на деякий проміжок з цих дат ремонт. Якщо такий ремонт вказано, то цей екземпляр вилючається з результатів пошуку. Якщо ремонт відсутній, то далі перевіряється, що цей спортивний інвентар не був заброньований на ці дати. Під час пошуку перевіряється тільки сам основний інвентар. Допоміжні засоби, для яких вказується кількість, перевіряються під час оформлення замовлення. Звичайно, що під час виконання замовлення ці перевірки вже відбуваються не циклічно для всіх екземплярів спортивного інвентаря, а саме для конкретного інвентаря, а вже перевірки для допоміжних засобів забезпечуються за кожним вказаним записом. Оформлення замовлення реалізовано функцією `new_booking`, яка містить в якості параметра номер спортивного інвентаря, за яким створюється бронювання, в параметрі `sportequipmentid`.

Після того, як було встановлено, що бронювання оренди цього спортивного інвентаря на вказані дати немає, відбувається обчислення вартості такої оренди. Обчислення вартості виконується з розбиттям на проміжки, тобто враховується варіант, що оренда може охоплювати декілька сезонів. Тоді визначається проміжок дат для відповідного сезону, який в нього входить, виконується обчислення мінімальної вартості за ним, обираючи серед розрахунку на основі погодинного, щотижневого, місячного та річного тарифів. Після цього виконується обчислення вартості на наступному секторів, що відповідає наступному сезону. Сума всіх секторів

визначається як вартість оренди. Вона вноситься в результати, а цей екземпляр спортивного інвентаря додається до результатів, якщо отримане значення потрапляє у вказаний у параметрах пошуку інтервал.

Функція `user_reviews` реалізує перегляд списку відгуків про користувача за вказаним рівнем оцінки. Рівень оцінки визначається параметрами нижньої оцінки – `markbottom` та верхньої межі оцінки – `marktop`. Логін користувача, для якого відбираються відгуки, вказується в параметрі `userlogin`. Аналогічна функція перегляду списку відгуків про спортивний інвентар за вказаним рівнем оцінки реалізована функцією `sportequipment_reviews`. У неї додатковим параметром є номер екземпляру спортивного інвентаря – `sportequipmentid`.

Перегляд всіх сервісних обслуговувань екземпляра спортивного інвентаря реалізований функцією `sportequipment_services`, в параметрі якої `sportequipmentid` вказується номер екземпляру спортивного інвентаря.

Перегляд всього переліку допоміжних засобів за екземпляром спортивного інвентаря реалізований функцією `sportequipment_services`, в параметрі якої `sportequipmentid` вказується номер екземпляру спортивного інвентаря.

Перегляд списку всіх запитань і відповідей господаря реалізований функцією `host_answers`, де логін господаря вказується в параметрі `hostlogin`. Всі запитання відсортовуються за датою створення. Тобто запитання з відповідями надаються у відкритому режимі, так що кожен бажаючий може побачити цю інформацію. Перейти до цього переліку можна зі сторінки господаря.

Перегляд оголошення про оренду спортивного інвентаря створений функцією `sport_equipment`, в параметрі якої `sportequipmentid` вказується номер екземпляру спортивного інвентаря. У підсумку витягнуті дані заповнюють шаблон сторінки `equipment.html`, в якому через наступні ключі доступні наступні дані:

- `sportequipment` – повні дані оголошення про оренду спортивного інвентаря, номер якого було вказано (сюди також додаються фотографії, дані господаря, типу категорії спортивного обладнання, відповідної категорії, відповідного виду спорту, інформація про виробника);
- `reviews` – перелік з 3 останніх відгуків про цей інвентар;
- `booked` – інформація про те, чи міститься у закладці авторизованого користувача (якщо він авторизований) цей спортивний інвентар;
- `services` – перелік з 3 останніх обслуговувань;
- `baseprices` – дані базової вартості оренди спортивного інвентаря;
- `coefficients` – коефіцієнти сезонності для спортивного інвентаря;
- `seasons` – дані сезонів, встановлені для цього спортивного інвентаря;
- `helpers` – дані 5 можливих допоміжних засобів для цього спортивного інвентаря;
- `bookings_finished` – кількість повністю завершених бронювань оренди (оренд);
- `last_rental` – дата останньої оренди (завершеного бронювання);
- `last_booking` – дата останнього створеного замовлення;
- `cancellations` – політики скасування бронювання.

Усі бронювання мають набір станів, до яких входять: 0 – створене, сплачене бронювання, 1 – скасоване бронювання, 2 – бронювання, за яким подано скаргу, 3 – виконане коректно бронювання. Бронювання автоматично за запуском кожного дня скрипта переводиться в стан коректно виконаного, якщо після 3 днів після його завершення не було подано скаргу.

Перегляд сторінки користувача виконується за допомогою функції `user_view`, яка має параметр `userlogin`, що визначає логін цього користувача. Окрім даних самого користувача також повертаються відгуки про нього від господарів.

Перегляд сторінки господаря реалізований функцією `host`, яка має параметр `hostlogin`, що визначає логін господаря. Окрім даних господаря до шаблону сторінки передаються також відгуки про спортивний інвентар цього

господаря та перелік оголошень про оренду спортивного інвентаря, які були створені цим господарем.

Перегляд всіх своїх оголошень про оренду спортивного інвентаря господарем реалізований функцією `own_sportequipment`, яка не потребує параметрів, адже дані про господаря містяться за даними авторизації.

Перегляд отриманих запитань реалізований функцією `own_questions`, яка витягає всі запитання, які були адресовані господарю, який авторизувався. При спробі звернення з облікового запису, який не належить господарю, результати отримані не будуть.

Перегляд очікуваних оренд всього спортивного інвентаря господаря реалізується функцією `waited_bookings`, яка відбирає всі бронювання оренди спортивного інвентаря, які стосуються інвентаря, що належить господарю, що авторизувався, знаходяться в стані 0 (створене бронювання) та мають розпочатися пізніше поточної дати. Усі бронювання впорядковуються за датою початку. Відповідно перегляд історії оренд всього спортивного інвентаря господаря реалізований функцією `history_bookings`, в якій відбираються бронювання, які знаходяться в стані скарги або виконаних і які вже завершилися.

Перегляд календаря оренд екземпляру спортивного інвентаря господаря виконується на основі функції `sportequipment_bookings`, яка має параметр `sportequipmentid`, який визначає ідентифікатор самого екземпляру спортивного інвентаря.

Перегляд отриманих сум за оренду протягом інтервалу часу підтримується функцією `host_bookings_analyse` яка має параметри `dateFrom` і `dateTo`, що визначають дати початку і завершення оренд, за якими має бути відібрано дані. Відбираються всі бронювання, які було виконано коректно, тобто вони значить завершилися і можна враховувати їх суму. Всі бронювання розділяються на місяці зі вказаного часового проміжку. Тобто формується період з відповідної кількості місяців. За кожний місяць

визначається сума, яку було отримано з оренди за кожним екземпляром спортивного інвентаря, який є у господаря.

Функція `new_service` реалізує визначення сервісного обслуговування спортивного інвентаря. Сам спортивний інвентар визначається номером `sportequipmentid`, який вказується у параметрі. Це функція створення запису в БД, тому вона як і будь-які подібні функції, які для зменшення опису не виносились відповідно у цей розділ, реалізована через створення відповідної форми, до якої виконуються звернення у функції. У даному випадку це форма класу `ServiceForm`. Також перед внесенням даних відбуваються необхідні перевірки. У даному випадку це перевірка про те, що авторизований користувач є господарем і що цей господар є господарем конкретно вказаного за номером у параметрі спортивного інвентаря. Якщо це не вірно, то форма для внесення даних не надається.

Будь-які функції, пов'язані з коригуванням даних, реалізуються на основі відповідних форм, які створені для внесення даних. Тому знову ж таки для уникнення дублювання принципів створення ці описи не виносились у даний розділ.

Перегляд закладок оголошень про оренду спортивного інвентаря виконано функцією `bookmarks`, яка не має параметрів, адже виводяться всі створені закладки за авторизованим користувачем. Якщо користувач має обліковий запис господаря, то дані відповідно не будуть виведені. Якщо він є просто авторизованим користувачем, то відбираються всі створені закладки. Саме створення закладки реалізовано функцією `new_bookmark` з визначенням номеру спортивного інвентаря `sportequipmentid`, закладку за яким потрібно створити. Вилучення закладки відбувається за допомогою функції `delete_bookmark`, в якій в параметрі `bookmarked` визначається номер закладки.

Перегляд своїх запитань до господаря виконується функцією `own_questions`.

Перегляд власних бронювань оренди спортивного інвентаря реалізовано функцією `own_bookings`, у якій для авторизованого користувача відбираються всі бронювання, які він створив. Перегляд відкритих користувачу бронювань виконується функцією `own_shares`. За подібною логікою реалізовано інші представлення для програмного застосунку.

3.3 Висновки за розділом 3

ПЗ програмно реалізовано. Представлено порядок, за яким користувачі різних ролей працюють з програмою, та описано ті функції, які були створені в програмі задля забезпечення функціональних вимог.

4 ЕКСПЛУАТАЦІЯ ТА ТЕСТУВАННЯ ПРОГРАМИ

4.1 Призначення програми

Програма забезпечує підтримку господарів спортивного інвентаря щодо виставлення спортивного інвентаря у вигляді пропозицій оренди, керування самими цими пропозиціями і бронюваннями оренди спортивного інвентаря, а підтримку інших користувачів щодо пошуку тих пропозицій, які задовольняють їх вимогам, та бронювання оренди спортивного інвентаря за цими пропозиціями.

4.2 Умови виконання програми

Вебсервер має бути забезпечений процесором у мінімальній конфігурації Intel Core 2 або відповідним еквівалентом, оперативною пам'яттю з доступом не менше ніж до 4 Гб, жорстким диском з бажаним доступним обсягом не менше ніж 100 Гб, інтерпретатором мови python та системою керування БД MySQL.

4.3 Виконання і тестування програми

Робота з програмою розпочинається з пошуку спортивного інвентаря. Це основна сторінка, на якій потрібно в формі пошуку задати виробника або, додаючи послідовно, можливих виробників спортивного інвентаря, види спорту, для яких інвентар призначений, категорії спортивного інвентаря (також можна долати декілька), типи категорій, адресу пошуку (можна вибрати точку на мапі, натискаючи відповідну кнопку), радіус пошуку відносно цієї точки, мінімальну та максимальну вартість оренди, дату і час початку та дату і час закінчення оренди. Тоді на сторінці під формою відображаються результати пошуку (рис. 4.1).

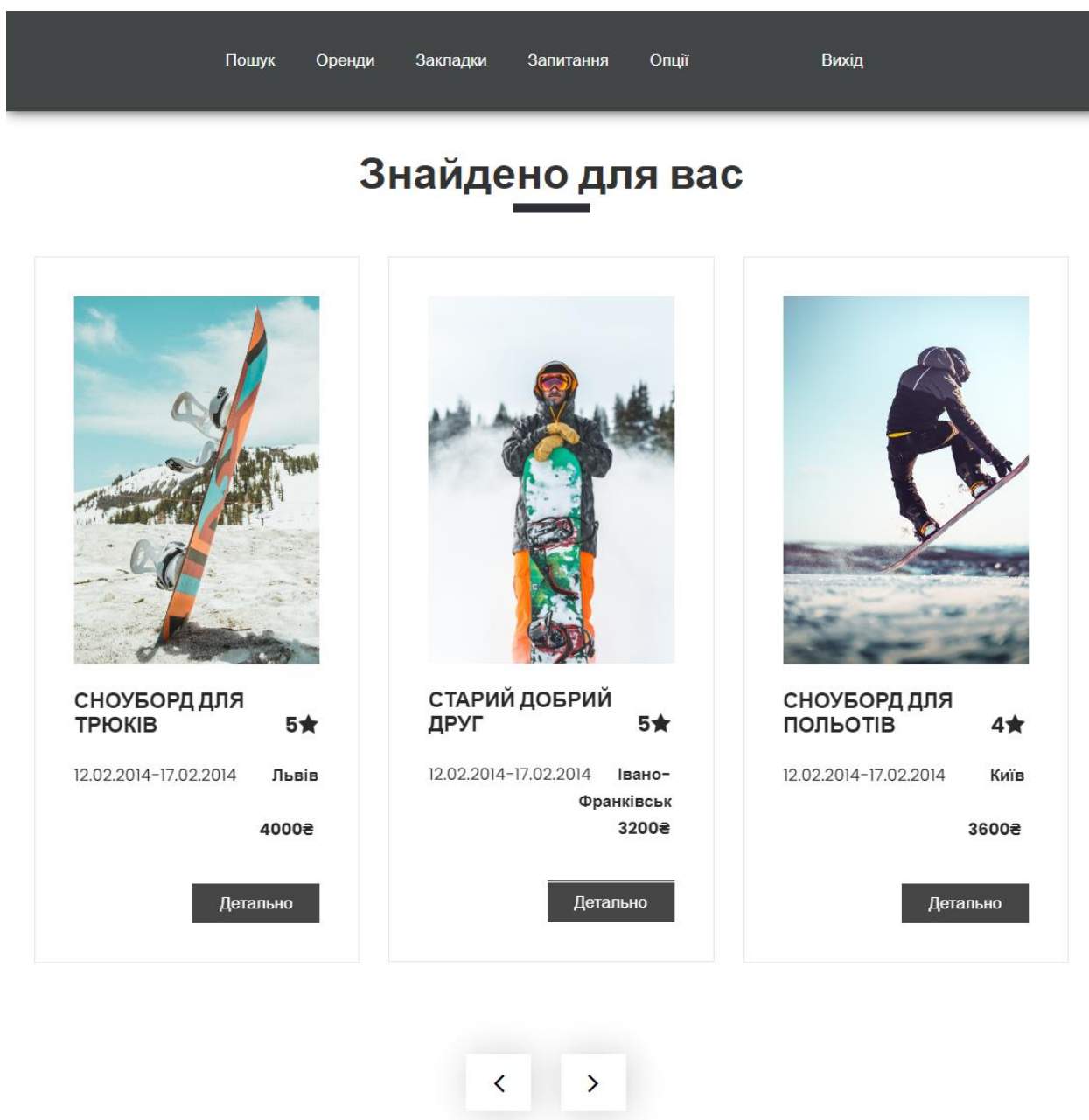


Рисунок 4.1 – Сторінка з пошуком спортивного інвентаря в частині результатів пошуку

У результатах пошуку кожне оголошення про оренду спортивного інвентаря представлено фотографією, назвою, середньою оцінкою, періодом оренди, містом, де знаходиться цей інвентар, вартістю оренди. Натикаючи на «Детально», можна відкрити сторінку зі всією інформацією про цей спортивний інвентар (рис. 4.2).

Пошук
Оренди
Закладки
Запитання
Опції
Вихід

Червоні стріли

Лижі в прекрасному стані. Їм лише рік. Сам використовую.
Буду радий поділитися з людьми, хто звик берегти речі.

Замовити
Зберегти

Горні лижі



Виробник

DPS



Оцінка

5



Оренд

3



Востаннє заброньовано

03.06.2023



Остання оренда

02.06.2023



Бронювань

5

Рисунок 4.2 – Сторінка оголошення про оренду спортивного інвентаря

На сторінці спортивного інвентаря згори представлено назву оголошення, перший абзац опису (повний опис розташований нижче). Також доступні кнопки для замовлення та збереження в закладках цього спортивного інвентаря. Відображаються фотографії, між якими можна

перемикатися. Далі виводиться інформація про виробника цього спортивного інвентаря, про середню оцінку, про кількість оренд (тобто завершених бронювань), кількість оформлених бронювань, дату, коли востаннє було створено бронювання, і дату, коли завершилось останнє фактично виконане бронювання. Ще нижче знаходяться дані про господаря, звідки можна перейти до його сторінки, адреса його розташування, зображена на мапі, дані про сервісне обслуговування цього спортивного інвентаря, таблиця з інформацією про періоди вартості цього інвентаря, 5 варіантів допоміжних засобів до цього інвентаря з кнопкою для відкриття сторінки з повним набором варіантів (при замовленні можна додати зі сторінки оформлення замовлення будь-який потрібний або декілька з доступних), варіанти політики відміни бронювання та 3 останні відгуки (рис. 4.3) про цей інвентар (під ними є кнопка для відкриття сторінки зі всіма відгуками).

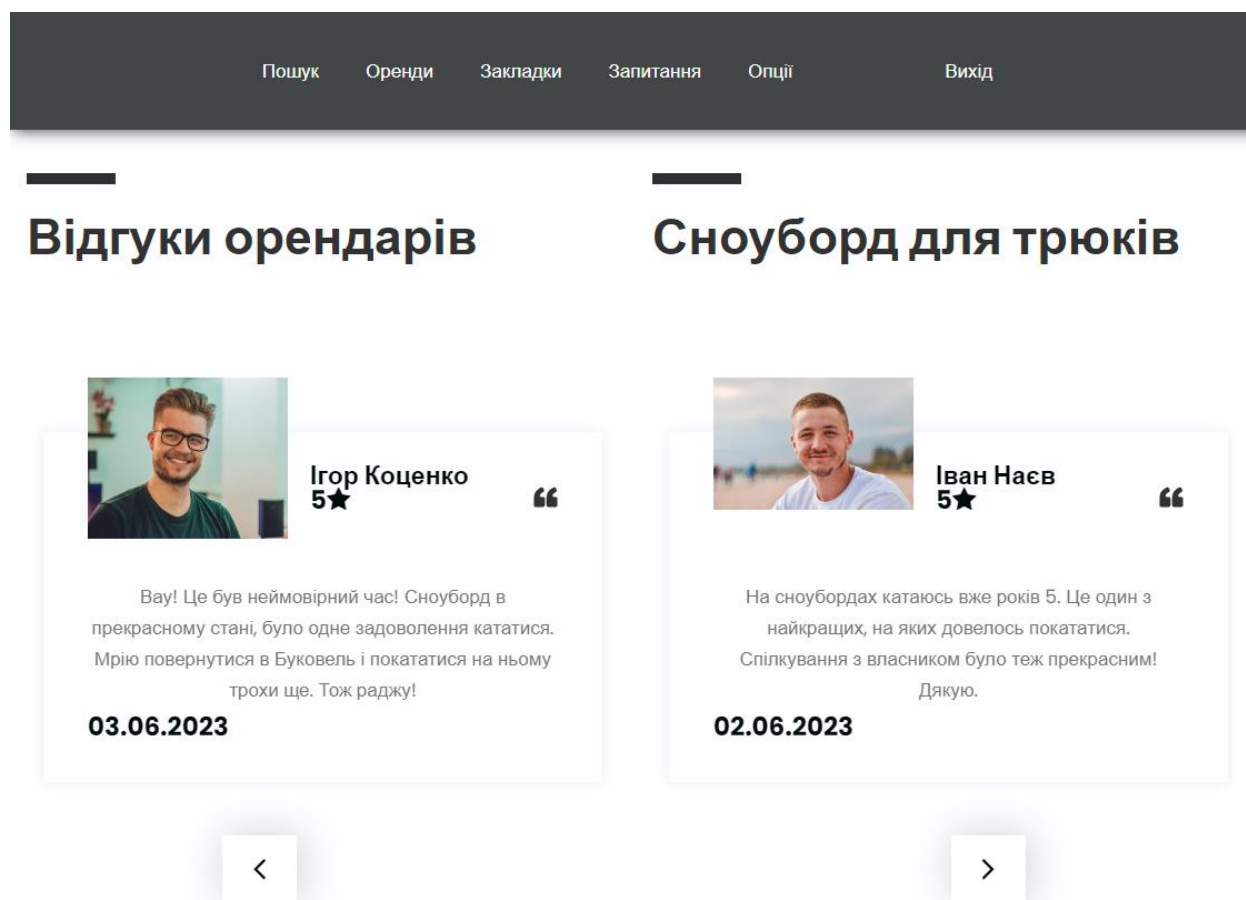


Рисунок 4.3 – Сторінка відгуків орендарів про спортивний інвентар

Якщо спортивний інвентар вже додано до закладок, то відображається кнопка «Прибрати з закладок». Переглянути самі збережені оголошення можна з розділу «Закладки».

Розділ «Оренди» дозволяє переглянути оформлені власні оренди спортивного інвентаря. З цієї ж сторінки за виконаними орендами доступна кнопка для створення відгуку про господаря окремо і про сам інвентар окремо. Якщо оренда не виконана, то її можна скасувати.

З налаштування опцій є можливість змінити дані власного профілю фактично.

Зі сторінки господаря наявна кнопка «Запитати». Переглянути відповіді можна з розділу «Запитання».

Для господаря варіант меню змінюється (рис. 4.4). Він містить зокрема розділ «Доходи» для перегляду суми виконаних оренд за період часу за всім власним спортивним інвентарем.

Пошук	Інвентар	Оренди	Історія	Засоби	Доходи	Запитання	Опції	Вихід
-------	----------	--------	---------	--------	--------	-----------	-------	-------

Доходи за вашим спортивним інвентарем			
01.05.2023	06.06.2023	ГОТОВО	
	Травень 2023	Червень 2023	Разом
Горні лижі DPS #2	2600€	5200€	7800€
Горні лижі Rossignol #3	1800€	4000€	5800€

Рисунок 4.4 – Сторінка перегляду власних доходів господарем

З розділу «Інвентар» господар може керувати власними оголошеннями. В тому числі можна змінювати оголошення, додавати нове. За кожним оголошенням зокрема доступна можливість переглянути календар бронювань, керувати базовою вартістю, сезонністю та коефіцієнтами сезонності. Зі сторінки самого інвентаря господар може додати допоміжні засоби до нього, тобто те, що він буде надавати вибрати до комплекту оренди.

З розділу «Оренди» доступні бронювання, які ще не виконані за всім спортивним інвентарем господаря. З розділу «Історія» доступні відповідно бронювання, які завершилися. Керування допоміжними засобами доступне в меню «Засоби».

Відповісти на запитання, які поставлені були, можна з меню «Запитання».

За описаною логікою було проведено тестування ПЗ з перевіркою його роботи під різними браузерами та щодо виконання функцій. Які було заплановані для реалізації. Усі тести під керуванням браузерів Chrome і Microsoft Edge були завершені успішно.

4.4 Висновки за розділом 4

ПЗ реалізовано і описано в частині взаємодії користувача як орендаря і користувача як господаря з програмою.

ВИСНОВКИ

У представленій дипломній кваліфікаційній роботі розроблено ПЗ оренди спортивного інвентаря, що направлено на підтримку керування відповідними процесами зі сторони господарів і орендарів. Проектування і розробку виконано на основі створення вебзастосунку. Для реалізації цих етапів спочатку виконано аналіз стану проблему на поточний момент, базуючись на аналізі програмних аналогів, які головним чином є вебзастосунками підтримки роботи сервісів конкретної категорії спортивного інвентаря.

Розроблено програму, яка забезпечує підтримку господарів спортивного інвентаря щодо виставлення спортивного інвентаря у вигляді пропозицій оренди, керування самими цими пропозиціями і бронюваннями оренди спортивного інвентаря, а підтримку інших користувачів щодо пошуку тих пропозицій, які задовольняють їх вимогам, та бронювання оренди спортивного інвентаря за цими пропозиціями.

Виконано і описано рішення, які забезпечили проектування БД, моделювання сценаріїв роботи користувачів, вибір python як основної мови програмування, реалізацію основних програмних складових, створення інтерфейсу програми.

Розроблене програмне забезпечення складається з інтерфейсів для гостя, господаря та авторизованого користувача. Інтерфейси господаря та авторизованого користувача враховують відповідну специфіку та базуються на інтерфейсі гостя. Створені засоби дозволяють керувати орендою спортивного інвентаря, починаючи взагалі від внесення пропозицій, завершуючи програвим переведенням до успішно завершеної оренди або поданням скарги господарем або звичайним авторизованим користувачем, яким є орендар, на бронювання.

Усі завдання, винесені на дипломну кваліфікаційну роботу, виконано.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. What is the 10 year outlook of the US sporting and recreational goods rental market? [Electronic resource]. – Access mode : <https://askwonder.com/research/10-year-outlook-us-sporting-recreational-goods-rental-market-v8igxpqle>.
2. How to make money on renting sports equipment? [Electronic resource]. – Access mode : <https://novoempreendedorismo.com/en/how-to-make-money-on-renting-sports-equipment/>.
3. Henkel, R. Renting Instead of Buying [Electronic resource] : How Sports Equipment Rental Opens up New Target Groups / R. Henkel. – Access mode : <https://www.ispo.com/en/companies/renting-instead-buying-how-sports-equipment-rental-opens-new-target-groups>.
4. Ikhide, J. Renting Sports Equipment [Electronic resource] : How to Expand Your Target Market / J. Ikhide. – Access mode : <https://www.daysmartrecreation.com/blog/renting-sports-equipment-how-to-expand-your-target-market>.
5. Ski rental X-drive [Electronic resource]. – Access mode : <https://x-drive.ua/en/>.
6. Ski hire online – ski and snowboard rental Snowrental [Electronic resource]. – Access mode : <https://www.snowrental.com/en>.
7. CheckYeti [Electronic resource] : Book Outdoor Activities in 1500+ Holiday Destinations. – Access mode : <https://www.checkyeti.com/>.
8. FriendWithA [Electronic resource] : The Online Marketplace for Gear Rentals. – Access mode : <https://friendwitha.com/>.
9. Python in a Nutshell [Text] : A Desktop Quick Reference / Alex Martelli, Anna Ravenscroft, Steve Holden et al. – Massachusetts : O'Reilly Media, 2023. – 735 p.
10. Copeland, D. B. Sustainable Web Development with Ruby on Rails [Text] : Practical Tips for Building Web Applications that Last / D. B. Copeland. – New York : David Bryant Copeland, 2020. – 472 p.

11. Hightech Free Website Template. Free CSS Templates. Free CSS [Electronic resource]. – Access mode : <https://www.free-css.com/free-css-templates/page291/hightech>.

ДОДАТОК А
Технічне завдання

Вступ

Оренда спортивного інвентаря є достатньо розповсюдженим явищем. Сервіси прокату велосипедів, обладнання для зимових видів спорту відомі всім. І ці приклади тільки підкреслюють, що саме оренда є доволі розповсюдженою для них, а вони є як раз прикладами спортивного інвентаря. Однак в такому вигляді, як це відбувається найчастіше, існують певні сервіси прокату, вони є централізованими, надають типові обладнання, у той же час власники подібного інвентаря є джерелом для оренди. Тож це джерело можна застосувати для оренди спортивного інвентаря, але це потребує підтримки за допомогою ПЗ, адже цей ринок є дуже фрагментованим: всі господарі знаходяться в різних місцях, мають дуже різне обладнання. Відповідно ПЗ і має бути розроблено в роботі.

А.1 Підстави для розробки

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

А.2 Призначення розробки

Програма, яка розробляється, в підсумку має забезпечувати підтримку господарів спортивного інвентаря щодо виставлення їх у вигляді пропозицій оренди, керування самими цими пропозиціями і бронюваннями оренди спортивного інвентаря, а підтримку інших користувачів щодо пошуку тих пропозицій, які задовольняють їх вимоги, та бронювання оренди спортивного інвентаря за цими пропозиціями.

A.3 Основні вимоги до програми

A.3.1 Вимоги до функціональних характеристик

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

- пошук спортивного інвентаря;
- створення відгуку про спортивний інвентар, що був у оренді;
- створення відгуку про користувача, який орендував спортивний інвентар;
- перегляд списку відгуків про користувача за вказаним рівнем оцінки;
- перегляд списку відгуків про спортивний інвентар за вказаним рівнем оцінки;
- бронювання оренди спортивного інвентаря;
- визначення допоміжних засобів для оренди спортивного інвентаря;
- скасування бронювання;
- перегляд отриманих господарем сум за оренду протягом інтервалу часу;
- створення оголошення про оренду спортивного інвентаря;
- внесення змін до оголошення про оренду спортивного інвентаря;
- перегляд всіх своїх оголошень про оренду спортивного інвентаря;
- перегляд закладок оголошень про оренду спортивного інвентаря;
- створення запитання до господаря;
- створення відповіді на запитання господарем;
- перегляд отриманих господарем запитань;
- перегляд своїх запитань до господарів;
- створення закладки на оголошення про оренду спортивного інвентаря;
- видалення створеної закладки на оголошення про оренду спортивного інвентаря;

- визначення базової вартості спортивного інвентаря або допоміжного засобу;
- коригування базової вартості спортивного інвентаря або допоміжного засобу;
- визначення сезонності для спортивного інвентаря господаря;
- визначення коефіцієнтів сезонності для спортивного інвентаря або допоміжного засобу;
- коригування коефіцієнтів сезонності для спортивного інвентаря або допоміжного засобу;
- перегляд очікуваних оренд всього спортивного інвентаря господаря;
- перегляд календаря оренд екземпляру спортивного інвентаря господаря;
- перегляд історії оренд всього спортивного інвентаря господаря;
- визначення періоду ремонту екземпляру спортивного інвентаря;
- коригування періоду ремонту спортивного інвентаря;
- визначення сервісного обслуговування спортивного інвентаря;
- перегляд всіх сервісних обслуговувань екземпляра спортивного інвентаря;
- визначення доступу до власного бронювання іншому користувачу;
- перегляд власних бронювань оренди спортивного інвентаря;
- перегляд відкритих користувачу бронювань;
- визначення політики відміни бронювань;
- коригування політики відміни бронювань;
- перегляд оголошення про оренду спортивного інвентаря;
- перегляд всього переліку допоміжних засобів за екземпляром спортивного інвентаря;
- визначення скарги за бронюванням;
- перегляд сторінки користувача;
- перегляд сторінки господаря;
- перегляд списку всіх запитань і відповідей господаря;

- авторизація і реєстрація користувачів і господарів.

A.3.2 Вимоги до складу та параметрів технічних засобів

Апаратні вимоги до серверної сторони:

- процесор у мінімальній конфігурації Intel Core 2 або відповідний еквівалент;
- оперативна пам'ять з доступом не менше ніж до 4 Гб;
- жорсткий диск з бажаним доступним обсягом не менше ніж 100 Гб.

A.4 Порядок контролю та приймання

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

ДОДАТОК Б
Текст програми

Б.1 Текст файла `guestviews.py`

```

from .models import Sport, Brand, SportequipmentCategory,
SportequipmentCategoryType, Sportequipment, SportequipmentPhoto, Helper,
SportequipmentHelper, HelperBasePrice, HelperSeasonCoefficient, Booking,
BookingShare, BookingHelper, Cancellation, SportequipmentCancellation,
Delivery, User, Host, Seasontype, Season, BasePrice, SeasonCoefficient, Service,
Repair, ReviewUser, ReviewEquipment, Bookmark, Question

from django.shortcuts import render

import haversine

from haversine import Unit

from datetime import datetime, timedelta

from dateutil import relativedelta


def sportequipment_search(request, brands = [], sports = [], categories = [],
categorytypes = [], addx = None, addy = None, radius = 100, reviewlevel = 0,
priceFrom = None, priceTo = None, dateFrom = None, dateTo = None):

    sportequipments =
Sportequipment.objects.select_related("sportequipmentcategorytype").select_relate
d("sportequipmentcategorytype__sportequipmentcategory").select_related("sporte
quipmentcategorytype__sportequipmentcategory__sport").prefetch_related("photo
s").select_related("host").select_related("brand").all()

    if brands:
        brands_id = []
        brands = brands.split(".")
        for brand_id in brands:
            brands_id.append(int(brand_id))
        sportequipments = sportequipments.filter(brand__in = brands_id)

    if sports:
        sports_id = []

```

```

    sports = sports.split(".")
    for sport_id in sports:
        sports_id.append(int(sport_id))
    sportequipments =
sportequipments.filter(sportequipmentcategorytype__sportequipmentcategory__sp
ort__in = sports_id)
    if categories:
        categories_id = []
        categories = categories.split(".")
        for category_id in categories:
            categories_id.append(int(category_id))
        sportequipments =
sportequipments.filter(sportequipmentcategorytype__sportequipmentcategory__in
= categories_id)
        if categorytypes:
            categorytypes_id = []
            categorytypes = categorytypes.split(".")
            for category_id in categorytypes:
                categorytypes_id.append(int(category_id))
            sportequipments = sportequipments.filter(sportequipmentcategorytype
= categorytypes_id)
            if reviewlevel:
                sportequipments = sportequipments.filter(review_mark__gte =
reviewlevel)
            if addx and addy:
                geolocated = []
                coordinate_center = (addx, addy)
                for sportequipment in sportequipments:
                    sportequipment_center = (sportequipment.address_x,
sportequipment.address_y)

```

```

        dist = haversine.haversine(coordinate_center,
sportequipment_center, unit=Unit.KILOMETERS)
        if dist <= radius / 1000:
            geolocated.append(sportequipment)
        sportequipments = geolocated
    dtnow = datetime.now()
    if dateFrom:
        dateFrom = datetime.strptime(dateFrom, '%m/%d/%y %H:%M')
    if not dateFrom or (dateFrom < dtnow):
        dateFrom = dtnow
    if dateTo:
        dateTo = datetime.strptime(dateTo, '%m/%d/%y %H:%M')
    if not dateTo or (dateTo < dtnow) or (dateTo <= dateFrom):
        dateTo = dateFrom + timedelta(days = 1)
    dated = []
    for sportequipment in sportequipments:
        repairs = Repair.objects.filter(start_date__gte = dateFrom,
start_date__lte = dateTo)
        if repairs:
            continue
        repairs = Repair.objects.filter(end_date__gte = dateFrom,
end_date__lte = dateTo)
        if repairs:
            continue
        repairs = Repair.objects.filter(start_date__lte = dateFrom,
end_date__gte = dateTo)
        if repairs:
            continue
        books = Booking.objects.filter(start_date__gte = dateFrom,
start_date__lte = dateTo, status__in = [0, 2])

```

```

    if books:
        continue
        books = Booking.objects.filter(end_date__gte = dateFrom,
end_date__lte = dateTo, status__in = [0, 2])
    if books:
        continue
        books = Booking.objects.filter(start_date__lte = dateFrom,
end_date__gte = dateTo, status__in = [0, 2])
    if books:
        continue
        category =
sportequipment.sportequipmentcategorytype.sportequipmentcategory
        host = sportequipment.host
        price_hours = 0
        price_weeks = 0
        price_days = 0
        price_months = 0
        price_years = 0
        season = Season.objects.filter(sportequipmentcategory = category,
host = host, start_date__lte = dateFrom, end_date__gte = dateFrom)
        while True:
            if season.end_date >= dateTo:
                duration = relativedelta.relativedelta(dateTo, dateFrom)
                coefficient = SeasonCoefficient.objects.get(sportequipment =
sportequipment, seasontype = season.seasontype)
                hours = duration.hours
                baseprice = BasePrice.objects.filter(sportequipment =
sportequipment, unit = 0, quantity__lte = hours).order_by("-quantity")[0]
                price_new = baseprice.price * hours
                price_hours = price_hours + price_new

```

```

    days = duration.days
    baseprice = BasePrice.objects.filter(sportequipment =
sportequipment, unit = 1, quantity__lte = days).order_by("-quantity")[0]
    price_new = baseprice.price * days
    price_days = price_days + price_new
    weeks = duration.weeks
    baseprice = BasePrice.objects.filter(sportequipment =
sportequipment, unit = 2, quantity__lte = weeks).order_by("-quantity")[0]
    price_new = baseprice.price * weeks
    price_weeks = price_weeks + price_new
    months = duration.months
    baseprice = BasePrice.objects.filter(sportequipment =
sportequipment, unit = 3, quantity__lte = months).order_by("-quantity")[0]
    price_new = baseprice.price * months
    price_months = price_months + price_new
    years = duration.years
    baseprice = BasePrice.objects.filter(sportequipment =
sportequipment, unit = 4, quantity__lte = years).order_by("-quantity")[0]
    price_new = baseprice.price * years
    price_years = price_years + price_new
    price = min([price_hours, price_days, price_weeks,
price_months, price_years])
    condition = True
    if priceFrom and price < priceFrom:
        condition = False
    if priceTo and price > priceTo:
        condition = False
    if condition:
        sportequipment.price = price
        dated.append(sportequipment)

```

```

    sportequipments = dated
    break
else:
    duration = relativedelta.relativedelta(season.end_date, dateFrom)
    coefficient = SeasonCoefficient.objects.get(sportequipment =
sportequipment, seasontype = season.seasontype)
    hours = duration.hours
    baseprice = BasePrice.objects.filter(sportequipment =
sportequipment, unit = 0, quantity__lte = hours).order_by("-quantity")[0]
    price_new = baseprice.price * hours
    price_hours = price_hours + price_new
    days = duration.days
    baseprice = BasePrice.objects.filter(sportequipment =
sportequipment, unit = 1, quantity__lte = days).order_by("-quantity")[0]
    price_new = baseprice.price * days
    price_days = price_days + price_new
    weeks = duration.weeks
    baseprice = BasePrice.objects.filter(sportequipment =
sportequipment, unit = 2, quantity__lte = weeks).order_by("-quantity")[0]
    price_new = baseprice.price * weeks
    price_weeks = price_weeks + price_new
    months = duration.months
    baseprice = BasePrice.objects.filter(sportequipment =
sportequipment, unit = 3, quantity__lte = months).order_by("-quantity")[0]
    price_new = baseprice.price * months
    price_months = price_months + price_new
    years = duration.years
    baseprice = BasePrice.objects.filter(sportequipment =
sportequipment, unit = 4, quantity__lte = years).order_by("-quantity")[0]
    price_new = baseprice.price * years

```

```

        price_years = price_years + price_new
        dateFrom = season.end_date
        season = Season.objects.filter(sportequipmentcategory =
category, host = host, start_date__lte = dateFrom, end_date__gte = dateFrom)
    if request.user:
        for sportequipment in sportequipments:
            bookmark = Bookmark.objects.get(user = request.user,
sportequipment = sportequipment)
            if bookmark:
                sportequipment.bookmark = True
            else:
                sportequipment.bookmark = False
    data = {}
    data["sportequipments"] = sportequipments
    return render(request, "wesports/search.html", data)

def user_reviews(request, userlogin, markbottom = 0, marktop = 5):
    try:
        user = User.objects.filter(login = userlogin)
    except User.DoesNotExist:
        user = None
    if user:
        userreviews = ReviewUser.objects.filter(user = user, mark__gte =
markbottom, mark__lte = marktop).order_by("-review_date")
        data = {}
        data["reviews"] = userreviews
        return render(request, "wesports/userreviews.html", data)
    else:
        return render(request, "wesports/userreviews.html", [])

```

```

def sportequipment_reviews(request, sportequipmentid, markbottom = 0,
marktop = 5):
    try:
        sportequipment = Sportequipment.objects.filter(login =
sportequipmentid)
    except Sportequipment.DoesNotExist:
        sportequipment = None
    if sportequipment:
        sportequipmentreviews =
ReviewEquipment.objects.filter(sportequipment = sportequipment, mark__gte =
markbottom, mark__lte = marktop).order_by("-review_date")
        data = {}
        data["reviews"] = sportequipmentreviews
        return render(request, "wesports/eqreviews.html", data)
    else:
        return render(request, "wesports/eqreviews.html", [])

def sportequipment_services(request, sportequipmentid):
    try:
        sportequipment = Sportequipment.objects.filter(login =
sportequipmentid)
    except Sportequipment.DoesNotExist:
        sportequipment = None
    if sportequipment:
        sportequipmentservices = Service.objects.filter(sportequipment =
sportequipment).order_by("-service_date")
        data = {}
        data["services"] = sportequipmentreviews
        return render(request, "wesports/services.html", data)
    else:

```

```

        return render(request, "wesports/services.html", [])

def sportequipment_services(request, sportequipmentid):
    try:
        sportequipment = Sportequipment.objects.filter(login =
sportequipmentid)
    except Sportequipment.DoesNotExist:
        sportequipment = None
    if sportequipment:
        helpers =
SportequipmentHelper.objects.select_related("helper").select_related("helper__hos
t").select_related("helper__sportequipmentcategory").filter(sportequipment =
sportequipment).distinct("helper")
        data = {}
        data["helpers"] = helpers
        return render(request, "wesports/helpers.html", data)
    else:
        return render(request, "wesports/helpers.html", [])

def host_answers(request, hostlogin):
    try:
        host = Host.objects.filter(login = hostlogin)
    except Host.DoesNotExist:
        host = None
    if host:
        answers = Question.objects.filter(host = host).order_by("-quedate")
        data = {}
        data["answers"] = answers
        return render(request, "wesports/answers.html", data)
    else:

```

```

return render(request, "wesports/answers.html", [])

def sport_equipment(request, sportequipmentid):
    try:
        sportequipment =
Sportequipment.objects.select_related("sportequipmentcategorytype").select_relate
d("sportequipmentcategorytype__sportequipmentcategory").select_related("sporte
quipmentcategorytype__sportequipmentcategory__sport").prefetch_related("photo
s").select_related("host").select_related("brand").filter(id = sportequipmentid).get()
    except Sportequipment.DoesNotExist:
        sportequipment = None
    if sportequipment:
        data = {}
        data["sportequipment"] = sportequipment
        sportequipmentreviews =
ReviewEquipment.objects.filter(sportequipment = sportequipment, mark__gte =
markbottom, mark__lte = marktop).order_by("-review_date")[:3]
        data["reviews"] = sportequipmentreviews
        sportequipmentservices = Service.objects.filter(sportequipment =
sportequipment).order_by("-service_date")[:3]
        if request.user:
            booked = Bookmark.objects.filter(sportequipment =
sportequipment, user = user)
            if booked:
                booked = True
            else:
                booked = False
        else:
            booked = False
        data["services"] = sportequipmentservices

```

```

data["booked"] = booked

baseprices = BasePrice.objects.filter(sportequipment =
sportequipment)

data["baseprices"] = baseprices

coefficients =
SeasonCoefficient.objects.select_related("seasontype").filter(sportequipment =
sportequipment)

data["coefficients"] = coefficients

seasons =
Season.objects.select_related("seasontype").select_related("sportequipmentcategory").select_related("host").filter(sportequipmentcategory =
sportequipment.sportequipmentcategorytype.sportequipmentcategory)

data["seasons"] = seasons

helpers =
SportequipmentHelper.objects.select_related("helper").select_related("helper__host").select_related("helper__sportequipmentcategory").filter(sportequipment =
sportequipment).distinct("helper")[:5]

data["helpers"] = helpers

bookings = Booking.objects.filter(sportequipment = sportequipment,
state__in = [0, 2, 3]).count()

bookings_finished = Booking.objects.filter(sportequipment =
sportequipment, state = 3).count()

last_rental = Booking.objects.filter(sportequipment = sportequipment,
state = 3).order_by("-end_date")[0]

last_booking = Booking.objects.filter(sportequipment =
sportequipment).order_by("-booked_date")[0]

data["bookings"] = bookings

data["finished"] = bookings_finished

data["last_rental"] = last_rental.end_date

data["last_booking"] = last_booking.booked_date

```

```

        cancellations =
SportequipmentCancellation.objects.select_related("cancellation").filter(sportequip
ment = sportequipment).distinct("cancellation")
        data["cancellations"] = cancellations
        return render(request, "wesports/equipment.html", data)
    else:
        return render(request, "wesports/equipment.html", [])

```

```

def user_view(request, userlogin):
    try:
        user= User.objects.get(login = userlogin)
    except User.DoesNotExist:
        user= None
    if user:
        data = {}
        data["user"] = user
        reviews = ReviewUser.objects.filter(user = user).order_by("-
review_date")[:3]
        data["reviews"] = reviews
        return render(request, "wesports/user.html", data)
    else:
        return render(request, "wesports/user.html", [])

```

```

def host(request, hostlogin):
    try:
        host = Host.objects.get(login = hostlogin)
    except Host.DoesNotExist:
        host = None
    if host:
        data = {}

```

```

        data["host"] = host
        reviews = ReviewEquipment.objects.filter(sportequipment __host =
host).order_by("-review_date")[:3]
        data["reviews"] = reviews
        sportequipments = Sportequipment.objects.filter(host = host, closed =
False).order_by("-id")[:3]
        data["sportequipments"] = sportequipments
        return render(request, "wesports/host.html", data)
    else:
        return render(request, "wesports/host.html", [])

```

Б.2 Текст файла hostviews.py

```

from .models import Sport, Brand, SportequipmentCategory,
SportequipmentCategoryType, Sportequipment, SportequipmentPhoto, Helper,
SportequipmentHelper, HelperBasePrice, HelperSeasonCoefficient, Booking,
BookingShare, BookingHelper, Cancellation, SportequipmentCancellation,
Delivery, User, Host, Seasontype, Season, BasePrice, SeasonCoefficient, Service,
Repair, ReviewUser, ReviewEquipment, Bookmark, Question

from django.shortcuts import render, redirect
from dateutil import relativedelta

from .forms import ReviewUserForm, ServiceForm, RepairForm,
SeasonCoefficientForm, SeasonCoefficientForm, SeasonForm,
SportequipmentForm, HelperBasePriceForm

def own_sportequipment(request):
    try:
        host = Host.objects.get(login = request.user.login)
    except Host.DoesNotExist:
        host = None

```

```

    if host:
        data = {}
        sportequipments = Sportequipment.objects.filter(host =
host).order_by("-id")
        data["sportequipments"] = sportequipments
        return render(request, "wesports/equipments.html", data)
    else:
        return render(request, "wesports/equipments.html", [])

def own_questions(request):
    try:
        host = Host.objects.get(login = request.user.login)
    except Host.DoesNotExist:
        host = None
    if host:
        data = {}
        questions = Question.objects.filter(host = host).order_by("-id")
        data["questions"] = questions
        return render(request, "wesports/questions.html", data)
    else:
        return render(request, "wesports/questions.html", [])

def waited_bookings(request):
    try:
        host = Host.objects.get(login = request.user.login)
    except Host.DoesNotExist:
        host = None
    if host:
        data = {}
        dtnow = datetime.now()

```

```

        bookings = Booking.objects.filter(sportequipment__host = host, state
= 0, start_date__gte = dtnow).order_by("start_date")
        data["bookings"] = bookings
        return render(request, "wesports/waitedbookings.html", data)
    else:
        return render(request, "wesports/waitedbookings.html", [])

```

```

def history_bookings(request):
    try:
        host = Host.objects.get(login = request.user.login)
    except Host.DoesNotExist:
        host = None
    if host:
        data = {}
        dtnow = datetime.now()
        bookings = Booking.objects.filter(sportequipment__host = host,
state__in = [2,3], end_date__lte = dtnow).order_by("-start_date")
        data["bookings"] = bookings
        return render(request, "wesports/historybookings.html", data)
    else:
        return render(request, "wesports/historybookings.html", [])

```

```

def sportequipment_bookings(request, sportequipmentid):
    try:
        sportequipment = Sportequipment.objects.get(id = sportequipmentid)
    except Sportequipment.DoesNotExist:
        sportequipment = None
    if sportequipment:
        data = {}

```

```

        bookings = Booking.objects.filter(sportequipment =
sportequipment).order_by("-start_date")
        data["bookings"] = bookings
        return render(request, "wesports/equipbookings.html", data)
    else:
        return render(request, "wesports/equipbookings.html", [])

def host_bookings_analyse(request, dateFrom, dateTo):
    if dateFrom:
        dateFrom = datetime.strptime(dateFrom, '%m/%d/%y %H:%M')
    if dateTo:
        dateTo = datetime.strptime(dateTo, '%m/%d/%y %H:%M')
    try:
        host = Host.objects.get(login = request.user.login)
    except Host.DoesNotExist:
        host = None
    if host:
        sportequipments = Sportequipment.objects.filter(host = host)
        data = {}
        bookings = Booking.objects.filter(sportequipment__host = host,
start_date__gte = dateFrom, end_date__lte = dateTo, status = 3)
        bookings_value = 0
        for booking in bookings:
            bookings_value = bookings_value + booking.price
        data["sum"] = bookings_value
        data["sportequipments"] = sportequipments
        equipment_sums = []
        dateToTemp = dateFrom + relativedelta.relativedelta(months=1)
        while dateToTemp < dateTo:
            equipment_sum = []

```

```

        for sportequipment in sportequipments:
            bookings = Booking.objects.filter(sportequipment =
sportequipment, start_date__gte = dateFrom, end_date__lte = dateTo, status = 3)
            bookings_value = 0
            for booking in bookings:
                bookings_value = bookings_value + booking.price
            equipment_sum.append(bookings_value)
            dateToTemp = dateToTemp + relativedelta.relativedelta(months=1)
            equipment_sums.append(equipment_sum)
        return render(request, "wesports/analyse.html", data)
    else:
        return render(request, "wesports/analyse.html", [])

def new_user_review(request, userlogin):
    try:
        host = Host.objects.get(login = request.user.login)
    except Host.DoesNotExist:
        host = None
    try:
        user = User.objects.get(login = userlogin)
    except User.DoesNotExist:
        user = None
    booked = False
    if host and user:
        booked = Booking.objects.filter(sportequipment__host = host, user =
user)
    if booked:
        if request.method == "POST":
            reviewuserform = ReviewUserForm(request.POST)
            if reviewuserform.is_valid():

```

```

        reviewuser = reviewuserform.save(commit=False)
        reviewuser.user = user
        reviewuser.host = host
        reviewuser.save()
        reviews = ReviewUser.objects.filter(user = user)
        sum = 0
        for userreview in reviews:
            sum = sum + userreview.mark
            n = n + 1
        user.review_mark = sum / n
        user.save()
        redirect("userreviews", userlogin)
    else:
        reviewuserform = ReviewUserForm()
    else:
        reviewuserform = None
    data = {}
    data["form"] = reviewuserform
    return render(request, "wesports/newuserreview.html", data)

def new_service(request, sportequipmentid):
    try:
        host = Host.objects.get(login = request.user.login)
    except Host.DoesNotExist:
        host = None
    try:
        sportequipment = Sportequipment.objects.select_related("host").get(id
= sportequipmentid)
    except Sportequipment.DoesNotExist:
        sportequipment = None

```

```

if host and sportequipment and sportequipment.host == host:
    if request.method == "POST":
        serviceform = ServiceForm(request.POST)
        if serviceform.is_valid():
            service = serviceform.save(commit=False)
            service.sportequipment = sportequipment
            service.save()
            redirect("sportequipment", sportequipment.id)
        else:
            serviceform = ServiceForm()
    else:
        serviceform = None
    data = {}
    data["form"] = serviceform
    return render(request, "wesports/newservice.html", data)

def new_repair(request, sportequipmentid):
    try:
        host = Host.objects.get(login = request.user.login)
    except Host.DoesNotExist:
        host = None
    try:
        sportequipment = Sportequipment.objects.select_related("host").get(id
= sportequipmentid)
    except Sportequipment.DoesNotExist:
        sportequipment = None
    if host and sportequipment and sportequipment.host == host:
        if request.method == "POST":
            repairform = RepairForm(request.POST)
            if repairform.is_valid():

```

```

        repair = repairform.save(commit=False)
        repair.sportequipment = sportequipment
        repair.save()
        redirect("sportequipment", sportequipment.id)
    else:
        repairform = RepairForm()
    else:
        repairform = None
    data = {}
    data["form"] = repairform
    return render(request, "wesports/newrepair.html", data)

```

```

def give_answer(request, questionid, answer):
    try:
        host = Host.objects.get(login = request.user.login)
    except Host.DoesNotExist:
        host = None
    try:
        question = Question.objects.get(id = questionid)
    except Question.DoesNotExist:
        question = None
    if host and question and question.host == host:
        question.answer = answer
        question.save()
        redirect("hostanswers", host.login)

```

```

def change_repair(request, repairid):
    try:
        host = Host.objects.get(login = request.user.login)
    except Host.DoesNotExist:

```

```

        host = None
    try:
        repair =
Repair.objects.select_related("sportequipment").select_related("sportequipment__
host").get(id = repairid)
    except Repair.DoesNotExist:
        repair = None
    if host and repair and repair.sportequipment.host == host:
        if request.method == "POST":
            repairform = RepairForm(request.POST, instance = repair)
            if repairform.is_valid():
                repair = repairform.save(commit=False)
                repair.sportequipment = sportequipment
                repair.save()
                redirect("sportequipment", sportequipment.id)
            else:
                repairform = RepairForm(None, instance = repair)
        else:
            repairform = None
    data = {}
    data["form"] = repairform
    return render(request, "wesports/newrepair.html", data)

def new_baseprice(request, sportequipmentid):
    try:
        host = Host.objects.get(login = request.user.login)
    except Host.DoesNotExist:
        host = None
    try:

```

```

        sportequipment = Sportequipment.objects.select_related("host").get(id
= sportequipmentid)
    except Sportequipment.DoesNotExist:
        sportequipment = None
    if host and sportequipment and sportequipment.host == host:
        if request.method == "POST":
            basepriceform = BasePriceForm(request.POST)
            if basepriceform.is_valid():
                baseprice = basepriceform.save(commit=False)
                baseprice.sportequipment = sportequipment
                baseprice.save()
                redirect("sportequipment", sportequipment.id)
            else:
                basepriceform = BasePriceForm()
        else:
            basepriceform = None
    data = {}
    data["form"] = basepriceform
    return render(request, "wesports/newbaseprice.html", data)

def change_baseprice(request, basepriceid):
    try:
        host = Host.objects.get(login = request.user.login)
    except Host.DoesNotExist:
        host = None
    try:
        baseprice =
BasePrice.objects.select_related("sportequipment").select_related("sportequipment
__host").get(id = basepriceid)
    except BasePrice.DoesNotExist:

```

```

        baseprice = None
    if host and baseprice and baseprice.sportequipment.host == host:
        if request.method == "POST":
            basepriceform = BasePriceForm(request.POST, instance =
baseprice)

            if basepriceform.is_valid():
                basepriceform.save()
                redirect("sportequipment", sportequipment.id)
            else:
                basepriceform = BasePriceForm(None, instance = baseprice)
        else:
            basepriceform = None
    data = {}
    data["form"] = basepriceform
    return render(request, "wesports/newbaseprice.html", data)

def new_coefficient(request, sportequipmentid):
    try:
        host = Host.objects.get(login = request.user.login)
    except Host.DoesNotExist:
        host = None
    try:
        sportequipment = Sportequipment.objects.select_related("host").get(id
= sportequipmentid)
    except Sportequipment.DoesNotExist:
        sportequipment = None
    if host and sportequipment and sportequipment.host == host:
        if request.method == "POST":
            coefform = SeasonCoefficientForm(request.POST)
            if coefform.is_valid():

```

```

        coefficient = coefform.save(commit=False)
        coefficient.sportequipment = sportequipment
        coefficient.save()
        redirect("sportequipment", sportequipment.id)
    else:
        coefform = SeasonCoefficientForm()
    else:
        coefform = None
    data = {}
    data["form"] = coefform
    return render(request, "wesports/newcoeff.html", data)

def change_coefficient(request, coefficientid):
    try:
        host = Host.objects.get(login = request.user.login)
    except Host.DoesNotExist:
        host = None
    try:
        coefficient =
SeasonCoefficient.objects.select_related("sportequipment").select_related("sporteq
uipment__host").get(id = coefficientid)
    except SeasonCoefficient.DoesNotExist:
        coefficient = None
    if host and sportequipment and coefficient.sportequipment.host == host:
        if request.method == "POST":
            coefform = SeasonCoefficientForm(request.POST, instance
=coefficient)
            if coefform.is_valid():
                coefform.save()
                redirect("sportequipment", sportequipment.id)

```

```

        else:
            coefform = SeasonCoefficientForm(None, instance = coefficient)
    else:
        coefform = None
    data = {}
    data["form"] = coefform
    return render(request, "wesports/newcoeff.html", data)

def new_season(request, sportequipmentcategory):
    try:
        host = Host.objects.get(login = request.user.login)
    except Host.DoesNotExist:
        host = None
    try:
        category =
SportequipmentCategory.objects.select_related("sport").get(id =
sportequipmentcategory)
    except SportequipmentCategory.DoesNotExist:
        category = None
    if host:
        if request.method == "POST":
            seasonform = SeasonForm(request.POST)
            if seasonform.is_valid():
                season= season.save(commit=False)
                season.sportequipmentcategory = sportequipmentcategory
                season.host = host
                season.save()
                redirect("sportequipment", sportequipment.id)
            else:
                seasonform = SeasonForm()

```

```

else:
    seasonform = None
data = {}
data["form"] = seasonform
return render(request, "wesports/newseason.html", data)

```

```

def new_sportequipment(request, sportequipmentcategory):
    try:
        host = Host.objects.get(login = request.user.login)
    except Host.DoesNotExist:
        host = None
    if host:
        if request.method == "POST":
            sportequipmentform = SportequipmentForm(request.POST)
            if sportequipmentform.is_valid():
                sportequip = sportequipmentform.save(commit=False)
                sportequip.host = host
                sportequip.save()
                sportequipmentform.save_m2m()
                redirect("ownsporteq")
            else:
                sportequipmentform = SportequipmentForm()
        else:
            sportequipmentform = None
    data = {}
    data["form"] = sportequipmentform
    return render(request, "wesports/newsporteq.html", data)

```

```

def new_helperbaseprice(request, helperid):
    try:

```

```

        host = Host.objects.get(login = request.user.login)
except Host.DoesNotExist:
    host = None
try:
    helper = Helper.objects.get(id = helperid)
except Helper.DoesNotExist:
    helper = None
if host and helper.host == host:
    if request.method == "POST":
        basepriceform = HelperBasePriceForm(request.POST)
        if basepriceform.is_valid():
            basepriceform.save()
            basepriceform.save_m2m()
            redirect("ownsporteq")
        else:
            basepriceform = HelperBasePriceForm()
    else:
        basepriceform = None
data = {}
data["form"] = basepriceform
return render(request, "wesports/prices.html", data)

def new_helper_coefficient(request):
    try:
        host = Host.objects.get(login = request.user.login)
    except Host.DoesNotExist:
        host = None
    if host:
        if request.method == "POST":
            coefform = HelperSeasonCoefficientForm(request.POST)

```

```

        if coefform.is_valid():
            coefform.save()
            redirect("ownsporteq")
        else:
            coefform = HelperSeasonCoefficientForm()
    else:
        coefform = None
    data = {}
    data["form"] = coefform
    return render(request, "wesports/newhelpcoeff.html", data)

def change_helper_coefficient(request, coefficientid):
    try:
        host = Host.objects.get(login = request.user.login)
    except Host.DoesNotExist:
        host = None
    try:
        coefficient =
HelperSeasonCoefficient.objects.select_related("sportequipmentcategory").select_r
elated("seasontype").get(id = coefficientid)
    except HelperSeasonCoefficient.DoesNotExist:
        coefficient = None
    if host and sportequipment:
        if request.method == "POST":
            coefform = HelperSeasonCoefficientForm(request.POST, instance
= coefficient)
            if coefform.is_valid():
                coefform.save()
                redirect("ownsporteq")
            else:

```

```

        coefform = HelperSeasonCoefficientForm(None, instance =
coefform)
    else:
        coefform = None
    data = {}
    data["form"] = coefform
    return render(request, "wesports/newhelpcoeff.html", data)

def sportequipment_cancellation(request, sportequipmentid,
cancellationid):
    try:
        host = Host.objects.get(login = request.user.login)
    except Host.DoesNotExist:
        host = None
    try:
        sportequipment = Sportequipment.objects.get(id = sportequipmentid)
    except Sportequipment.DoesNotExist:
        sportequipment = None
    try:
        cancellation = Cancellation.objects.get(id = cancellationid)
    except Cancellation.DoesNotExist:
        cancellation = None
    if host and cancellation and sportequipment and sportequipment.host ==
host:
        sport_cancel = SportequipmentCancellation(sportequipment =
sportequipment, cancellation = cancellation)
        sport_cancel.save()
        redirect("ownsporteq")

def change_sportequipment_cancellation(request, sportcancelid):

```

```

try:
    host = Host.objects.get(login = request.user.login)
except Host.DoesNotExist:
    host = None
try:
    sportcancel =
SportequipmentCancellation.objects.select_related("sportequipment").select_relate
d("sportequipment__host").filter(id = sportcancelid)
except SportequipmentCancellation.DoesNotExist:
    sportcancel = None
if host and sportcancel.sportequipment.host == host:
    sportcancel.delete()
    redirect("ownsporteq")

def change_sportequipment(request, sportequipmentid):
    try:
        host = Host.objects.get(login = request.user.login)
    except Host.DoesNotExist:
        host = None
    try:
        sportequipment = Sportequipment.objects.get(id = sportequipmentid)
    except Sportequipment.DoesNotExist:
        sportequipment = None
    if host and sportequipment.host == host:
        if request.method == "POST":
            sportequipmentform = SportequipmentForm(request.POST,
instance = sportequipment)
            if sportequipmentform.is_valid():
                sportequipmentform.save()
                sportequipmentform.save_m2m()

```

```

        redirect("ownsporteq")
    else:
        sportequipmentform = SportequipmentForm(None, , instance =
sportequipment)
    else:
        sportequipmentform = None
    data = {}
    data["form"] = sportequipmentform
    return render(request, "wesports/newsporteq.html", data)

def sportequipment_helper(request, sportequipmentid, helperid):
    try:
        host = Host.objects.get(login = request.user.login)
    except Host.DoesNotExist:
        host = None
    try:
        sportequipment = Sportequipment.objects.get(id = sportequipmentid)
    except Sportequipment.DoesNotExist:
        sportequipment = None
    try:
        helper = Helper.objects.get(id = helperid)
    except Helper.DoesNotExist:
        helper = None
    if host and helper and sportequipment and sportequipment.host == host:
        sport_helper = SportequipmentHelper(sportequipment =
sportequipment, helper = helper)
        sport_helper.save()
        redirect("sportequipment", sportequipment.id)

def problem_booking(request, bookingid):

```

```
try:
    host = Host.objects.get(login = request.user.login)
except Host.DoesNotExist:
    host = None
try:
    booking =
Booking.objects.select_related("sportequipment").select_related("sportequipment_
_host").filter(id = bookingid).get()
except Booking.DoesNotExist:
    booking = None
if host and booking.sportequipment.host == host:
    booking.status = 2
    booking.save()
    redirect("bookings")
```

ДОДАТОК В
Слайди презентації



Рисунок В.1 – Слайд титульний

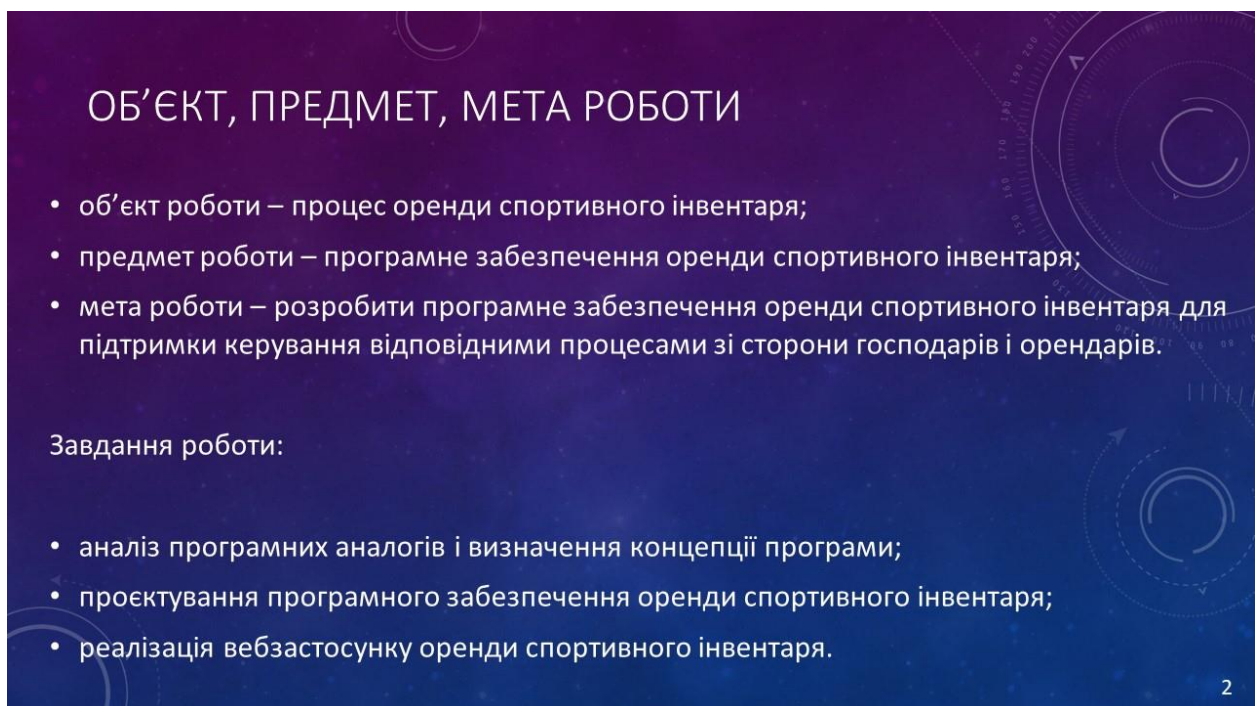


Рисунок В.2 – Об'єкт, предмет, мета роботи

РЕЗУЛЬТАТИ АНАЛІЗУ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ОРЕНДИ СПОРТИВНОГО ІНВЕНТАРЯ

Критерій	Checkyeti	FriendWithA	Моя програма
Джерело оренди спортивного інвентаря	Пункти прокату	Господарі	Господарі, можливо пункти прокату
Відгуки про господарів/пункти оренди спортивного інвентаря	–	+	+
Відгуки про сам спортивний інвентар	–	–	+
Бронювання оренди спортивного інвентаря	+	+	+
Створення закладок	–	+	+
Мінімальний строк оренди	День	День	Година

3

Рисунок В.3 – Результати аналізу програмного забезпечення оренди спортивного інвентаря

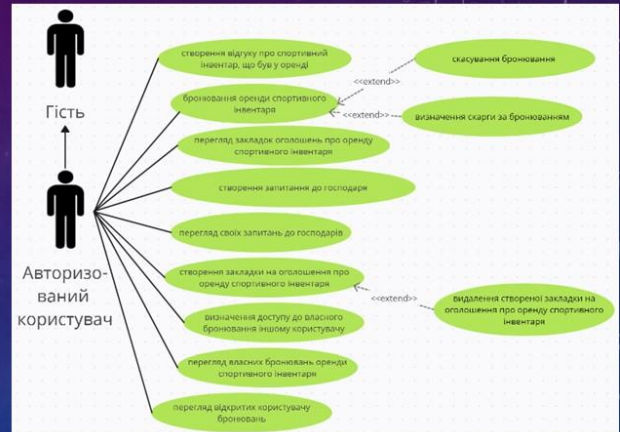
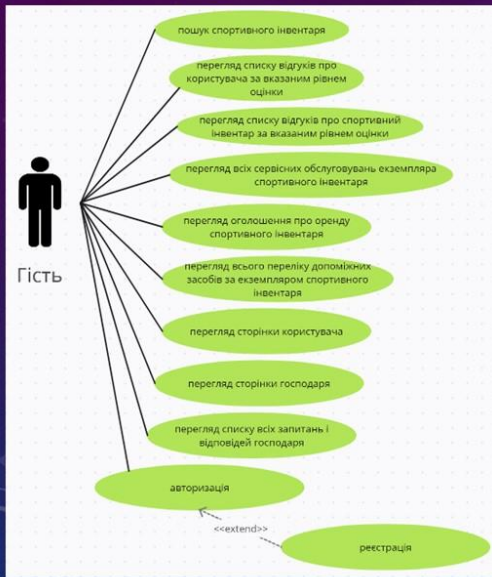
ПОРІВНЯННЯ МОВ ПРОГРАМУВАННЯ ДЛЯ РЕАЛІЗАЦІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Критерій	Python	Ruby
Тип мови програмування	Високорівнева	Високорівнева
Призначення мови програмування	Загальне	Загальне
Виконання веброзробки	Так	Так
Парадигми програмування	Мультипарадигмальна	Мультипарадигмальна
Доступність готових бібліотек	Широка	Порівняно менша
Засоби обробки даних	Високо поширені	Менша поширеність
Складність навчання	Низька	Вища

4

Рисунок В.4 – Порівняння мов програмування для реалізації програмного забезпечення

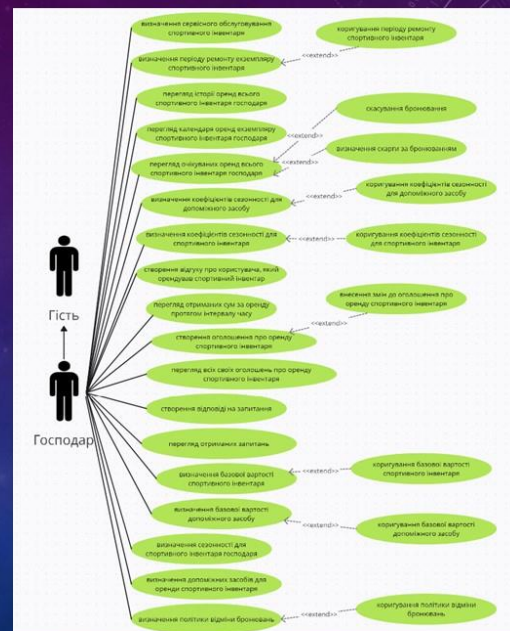
ДІАГРАМИ ПРЕЦЕДЕНТІВ



5

Рисунок В.5 – Діаграми прецедентів

ДІАГРАМА ПРЕЦЕДЕНТІВ



6

Рисунок В.6 – Діаграма прецедентів

СХЕМА БАЗИ ДАНИХ

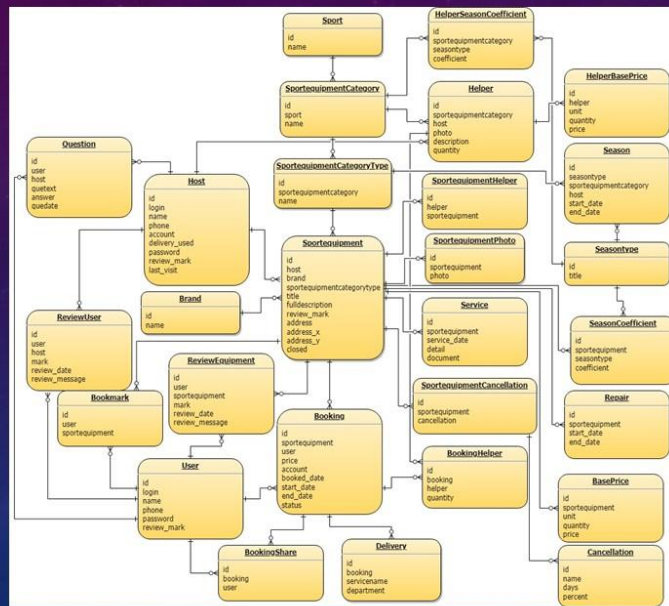


Рисунок В.7 – Схема бази даних

СХЕМА ФУНКЦІОНУВАННЯ ПРОГРАМИ

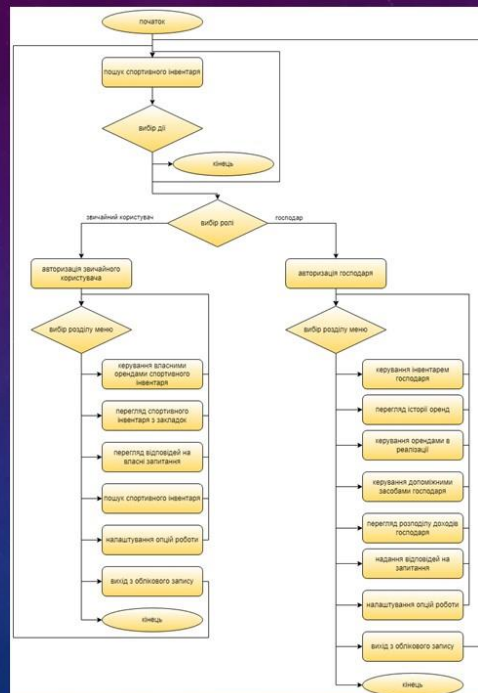


Рисунок В.8 – Схема функціонування програми

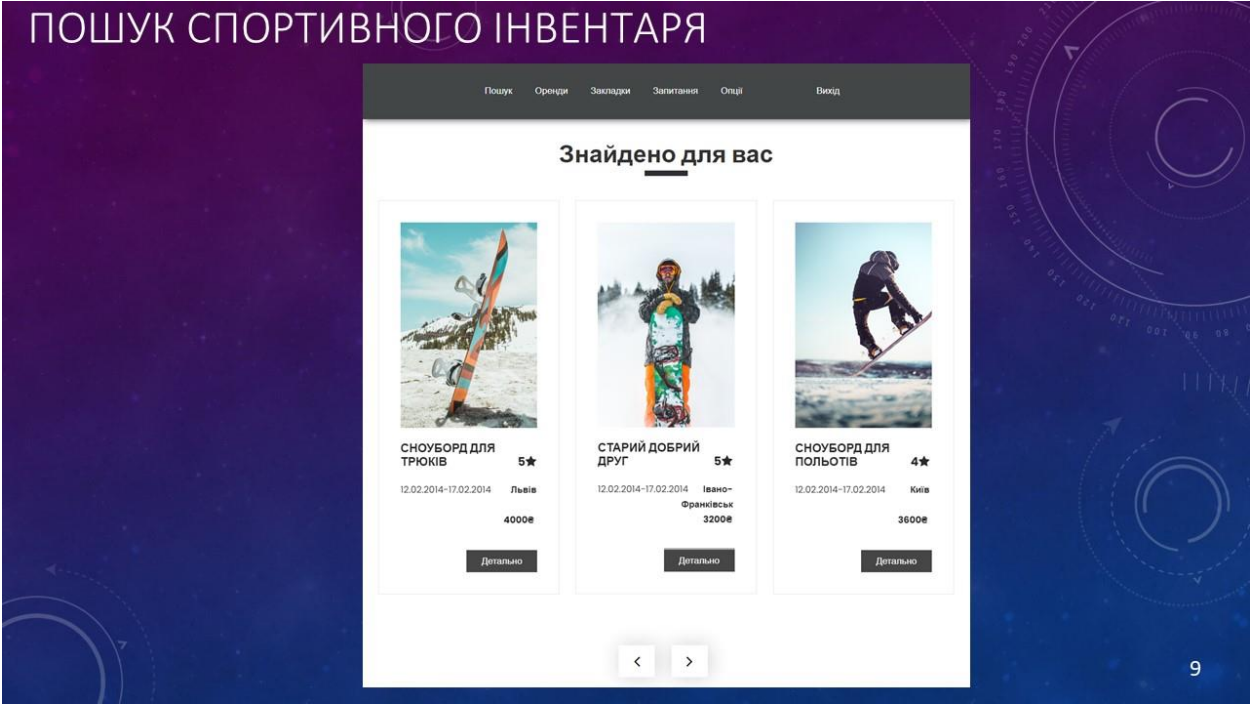


Рисунок В.9 – Пошук спортивного інвентаря

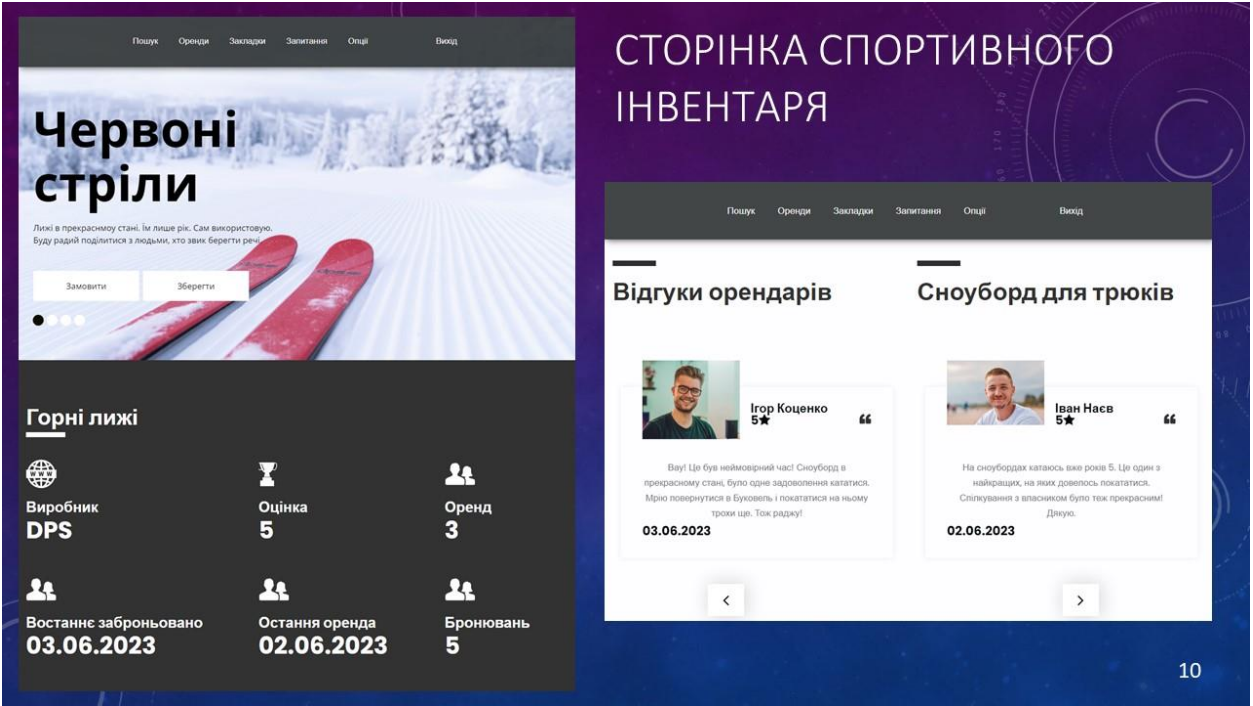
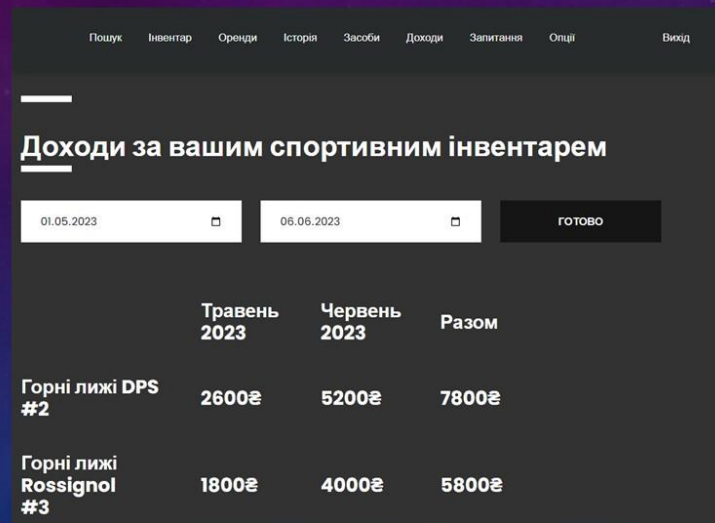


Рисунок В.10 – Сторінка спортивного інвентаря

ПЕРЕГЛЯД ВЛАСНИХ ДОХОДІВ ГОСПОДАРЕМ



	Травень 2023	Червень 2023	Разом
Горні лижі DPS #2	2600€	5200€	7800€
Горні лижі Rossignol #3	1800€	4000€	5800€

11

Рисунок В.11 – Перегляд власних доходів господарем

ВИСНОВКИ

- розроблено програмне забезпечення оренди спортивного інвентаря, що направлено на підтримку керування відповідними процесами зі сторони господарів і орендарів;
- проектування і розробку виконано на основі створення вебзастосунку: для реалізації цих етапів спочатку виконано аналіз стану проблему на поточний момент, базуючись на аналізі програмних аналогів, які головним чином є вебзастосунками підтримки роботи сервісів конкретної категорії спортивного інвентаря;
- виконано і описано рішення, які забезпечили проектування бази даних, моделювання сценаріїв роботи користувачів, вибір python як основної мови програмування, реалізацію основних програмних складових, створення інтерфейсу програми.
- розроблено програму, яка забезпечує підтримку господарів спортивного інвентаря щодо виставлення спортивного інвентаря у вигляді пропозицій оренди, керування самими цими пропозиціями і бронюваннями оренди спортивного інвентаря, а підтримку інших користувачів щодо пошуку тих пропозицій, які задовольняють їх вимогам, та бронювання оренди спортивного інвентаря за цими пропозиціями.
- усі завдання, винесені на дипломну кваліфікаційну роботу, виконано.

12

Рисунок В.12 – Висновки