

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проекту (роботи)

бакалавр

(ступінь вищої освіти)

на тему ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ОРГАНІЗАЦІЇ
ІНДИВІДУАЛЬНИХ ЗАНЯТЬ ТАНЦЯМИ
SOFTWARE FOR ORGANIZATION OF
PRIVATE DANCE CLASSES

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

Спеціальності 121 Інженерія програмного

(код і найменування спеціальності)

забезпечення

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

ЗІВЕР О.С.

(ПРИЗВИЩЕ та ініціали)

Керівник ЛЬОВКІН В.М.

(ПРИЗВИЩЕ та ініціали)

Рецензент ЗЕЛІК О.В.

(ПРИЗВИЩЕ та ініціали)

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

Факультет КНТ

Кафедра програмних засобів

Ступінь вищої освіти бакалавр

Спеціальність 121 Інженерія програмного забезпечення
(КОД І НАЙМЕНУВАННЯ)

Освітня програма (спеціалізація) Інженерія програмного забезпечення
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

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

ЗАВДАННЯ

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

ЗІВЕР Олександр Сергійович

(ПРІЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Програмне забезпечення організації індивідуальних занять танцями. Software for Organization of Private Dance Classes

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

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

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

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

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

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

Слайди презентації

6. Консультанти розділів проєкту (роботи)

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

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

КАЛЕНДАРНИЙ ПЛАН

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

Студент(ка)

_____ Олександра ЗІВЕР
(підпис) (Ім'я ПРИЗВИЩЕ)

Керівник проєкту (роботи)

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

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 118 с., 3 табл., 28 рис., 3 дод., 9 джерел.

ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, ТАНЦІ, ТАНЦІВНИК, ТРЕНЕР, ІНДИВІДУАЛЬНІ ЗАНЯТТЯ, БАЗА ДАНИХ.

Об'єкт роботи – процес організації індивідуальних занять танцями. Предмет роботи – програмне забезпечення організації індивідуальних занять танцями. Мета роботи – розробити програмне забезпечення організації індивідуальних занять танцями для підтримки супутнього процесу на його основних етапах.

Матеріали, методи та технічні засоби: Python і Django разом з HTML, CSS, JavaScript для веброзробки, PostgreSQL для підтримки у складі веброзробки роботи з базою даних.

Результати. Результати включають всі етапи виконання роботи, враховуючи визначення концепції програми організації індивідуальних занять танцями на основі виявлення відсутності готового рішення, встановлення вимог до програми, структури бази даних, розроблення всіх програмних одиниць та об'єднання їх у єдиний вебзастосунок, успішне тестування створеного вебзастосунку.

Висновки. Розроблена як підсумок програма призначена для забезпечення підтримки занять танцями на етапах пошуку партнерів для створення пари, підбору тренера для пари або окремого танцівника, підбору локацій для таких занять, комплексного вирішення таких проблем, планування самих індивідуальних занять танцями.

Галузь використання. Індивідуальні заняття парними та самотійними танцями стосовно створення і підтримки контактів між танцівниками, тренерами з танців на основі організації відповідного програмного майданчика.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 118 pages, 3 tables, 28 figures, 3 appendixes, 9 sources.

SOFTWARE, DANCE, DANCER, COACH, INDIVIDUAL LESSONS, DATABASE.

The object of the work is the process of organizing private dance classes. The subject of the work is the software for organizing private dance classes. The purpose of the work is to develop software for organizing private dance classes to support the accompanying process at its main stages.

Materials, methods and tools: Python and Django together with HTML, CSS, JavaScript for web development, PostgreSQL to support work with the database as part of web development.

Results. The results include all stages of the work, taking into account the definition of the concept of the program for organizing private dance classes based on the identification of the absence of a ready-made solution, setting requirements for the program, database structure, development of all program units and combining them into a single web application, successful testing of the created web application.

Conclusions. The program developed as a summary is intended to provide support for dance classes at the stages of searching for partners to create a pair, selecting a coach for a pair or an individual dancer, selecting locations for such classes, comprehensively solving such problems, planning the individual dance classes themselves.

Field of use. Individual pair and solo dance classes aimed at creating and maintaining contacts between dancers and dance coaches based on the organization of an appropriate program platform.

ЗМІСТ

	С.
Перелік скорочень та умовних познач.....	8
Вступ.....	9
1 Аналіз предметної області.....	10
1.1 Огляд можливостей існуючих програм організації індивідуальних занять танцями.....	10
1.2 Формування концепції розроблюваної програми організації індивідуальних занять танцями.....	16
1.3 Висновки за розділом 1	18
2 Проектування програми.....	19
2.1 Визначення функціональних вимог користувачів до програми організації індивідуальних занять танцями	19
2.2 Визначення засобів для розробки програми організації індивідуальних занять танцями	29
2.3 Структура бази даних програми організації індивідуальних занять танцями.....	29
2.4 Висновки за розділом 2	46
3 Розробка програми	47
3.1 Визначення порядку роботи з програмою.....	47
3.2 Реалізація основних класів програми організації індивідуальних занять танцями.....	48
3.3 Реалізація основних програмних функцій для організації індивідуальних занять танцями	62
3.4 Висновки за розділом 3	70
4 Експлуатація та тестування програми	71
4.1 Призначення та умови виконання програми організації індивідуальних занять танцями	71

4.2 Логіка експлуатації програми організації індивідуальних занять танцями	71
4.3 Тестування програми організації індивідуальних занять танцями.....	76
4.4 Висновки за розділом 4	81
Висновки	83
Перелік джерел посилання	84
Додаток А Технічне завдання	85
Додаток Б Текст програми	90
Додаток В Слайди презентації.....	111

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

CRUD – Create, Read, Update, Delete;

ПЗ – програмне забезпечення.

ВСТУП

Танці здавна привертають увагу людей, сформувавши цілу культуру навколо. Танці бувають індивідуальними, парними, можуть відбуватися у групах, а можуть організовуватися індивідуально для окремого танцівника, пари. При цьому танці потребують якісної підготовки, навчання для отримання задовільних результатів. Відповідно у всіх випадках окрім того, коли танцівник навчається повністю самостійно, він потребує взаємодії з іншими сторонами даного процесу, включаючи тренерів, інших танцівників. Також питанням є те, де саме такі заняття танцями будуть відбуватися, в який саме час. Це питання планування індивідуальних, групових занять танцями. І всі вони мають бути вирішені та вирішуються на практиці. Застосування програмного забезпечення (ПЗ), яке б повноцінно охопило всі складові цього процесу або принаймні основні з них, було би доцільним і актуальним. Попередній огляд існуючих програм, застосовуваних для організації індивідуальних занять танцями, показав, що такого повністю готового аналогу на цей момент не існує.

Відповідно мета роботи – розробити ПЗ організації індивідуальних занять танцями для підтримки супутнього процесу на його основних етапах.

Завдання роботи:

- аналіз існуючого стану проблеми розробки ПЗ організації індивідуальних занять танцями;
- визначення вимог до ПЗ організації індивідуальних занять танцями;
- проєктування ПЗ організації індивідуальних занять танцями;
- реалізація і тестування ПЗ організації індивідуальних занять танцями.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд можливостей існуючих програм організації індивідуальних занять танцями

Організація індивідуальних занять танцями доволі обмежено представлена в існуючих рішеннях, що пов'язано з фрагментацією цієї галузі на декілька окремих.

Перша з них пов'язана зі студіями танців. Тобто велика частина існуючих вебзастосунків представлена вебсайтами студій танців. Такі вебсайти містять дані виключно однієї студії танців, відповідно інформація подається консультативно, а заняття танцями представлені різними варіантами групових занять. Нерідко такі вебзастосунки є дуже обмеженими функціонально та містять фактично тільки загальний опис студії. Ширшу функціональність, наприклад, має вебзастосунок студії танців Mix Style [1] (рис. 1.1).

The screenshot displays the 'Розклад занять' (Class Schedule) page of the Mix Style dance studio website. At the top, there is a navigation bar with social media icons, a menu with 'ПРО НАС', 'РОЗКЛАД', 'СТИЛІ', a logo, and links for 'ТРЕНЕРИ', 'ФІЛІЇ', and 'КОНТАКТИ'. The main heading is 'Розклад занять'. Below it, a section titled 'Вибери вікову категорію' (Select age category) features a row of buttons for age groups: 'ДОРΟΣЛІ', '3-5 РОКІВ', '5-7 РОКІВ', '4-6 РОКІВ', '6-9 РОКІВ', '8-10 РОКІВ', '7-13 РОКІВ', '10-13 РОКІВ', and '13+ РОКІВ'. To the right of this row is a button labeled 'ВСІ ЗАНЯТТЯ'. Below the age selection is a dark blue banner for the 'Київ (Голосіїв)' (Kyiv (Holosiyiv)) location. It lists class times: 'Понеділок 18:00' and 'Середа 18:00'. A central image shows a group of dancers. To the right, the instructor is identified as 'Поліна Головня' (Polina Holovnya) for 'HIP-HOP' classes, suitable for '13+ років початковий' (13+ years, beginner). A green button says 'ПРОБНЕ ЗАНЯТТЯ' (Trial class) and a link below it says 'ЗА ЗАПИСОМ' (By appointment).

Рисунок 1.1 – Розклад занять студії танців Mix Style

У вебзастосунку Mix Style подано інформацію про розклад занять, при чому студія має декілька філіалів, про вартість різних варіантів абонементів (проте тільки такий варіант оплати доступний), про стилі, навчання яким здійснюється, про тренерів. Переважно функції організовані на сторінках таким чином, що представляється набір результатів та набір фільтрів, які так чи інакше впливають на відображення цих результатів, наявна можливість переглянути дані результатів детальніше на окремій сторінці.

Індивідуалізація у таких вебзастосунках відповідно не здійснюється. Другим напрямком є вебзастосунки, організовані для пошуку партнерів для танців. У такому випадку звісно заняття далі можуть здійснюватися у будь-яких форматах, у тому числі індивідуально. Однак, у такому випадку вже організація самих занять, тобто визначення місць зустрічі, можливо тренерів відбувається поза межами програми.

Прикладом другого напрямку є вебзастосунок DancePartner (рис. 1.2), у якому спочатку користувач під час реєстрації має задати свою анкету як танцівника, визначаючи власні параметри, визначаючи параметри пошуку майбутнього партнера для танців [2]. Відповідно вся подальша робота пов'язана з такими анкетами, представленням тих анкет потенційних партнерів для танців, які відповідають параметрам пошуку партнера. Фактично такі засоби є засобами сервісу знайомств, які застосовуються в сфері танців, тобто коли потенційні знайомства відбуваються між людьми, які займаються танцями і для занять танцями. Це дещо звужує застосування таких рішень, проте теж є доволі важливим інструментом.

DancePartner застосовується для пошуку партнерів для танців у всьому світі. Користувачі можуть обмінюватися повідомленнями між собою. Користувача, який шукає партнера для танців, можна або додати до улюблених, або заблокувати. Доступна також можливість прибрати сторінку з результатів пошуку.

dp Partner Search Criteria

This page is for *one-off* (unsaved) searches

Click [Search](#) (home page) to use your saved search criteria

Gender of my Partner

You're officially **Male** seeking **Female**, but you may **temporarily** change these to view "*your competition*" (other members like you). If you do, consider adjusting height, body styles, and anything else below as applicable. **Search criteria changes on this page will not be saved.**

I'm Male Female seeking Males Females Either

Search Area(s)

60 miles (97 km)

Lviv, Lviv Oblast, Ukraine

[EDIT](#)

Extended Search

You may **extend** your search **beyond your designated search area** to find dancers **willing-to-relocate** - willing to move to your city. **Extended** search is a popular choice for competitors and performers; less common for social dancers.

Normal Search within my search area only. Most common choice.

Extended Search my search area + **world-wide** for dancers tagged **willing-to-relocate**. Popular choice for competitors and performers.

Рисунок 1.2 – Критерії пошуку партнера для занять танцями у DancePartner

DancePartner за кожним користувачем здатний відобразити доволі велику кількість інформації, при цьому фактично за кожним виводячи тільки те, що ним було задано. Користувача можна привітати, відправити йому подарунок, написати лист (рис. 1.3). Це є базовими засобами комунікації між потенційними партнерами для занять танцями.

The screenshot shows the search results for a user named NYClatindancer98. The page includes a navigation bar with 'dp' logo, 'Search', 'Mailbox', 'Profile', 'Community', 'Help', 'Subscribe', and 'Logout'. The search results section shows 'Search Results' with filters for 'Photos' and 'Vaccinated', and a 'Sort Order' dropdown set to 'New, Proficiency, Distance, Last Login'. The user profile for NYClatindancer98 is displayed, including a 'New Member' badge, 'Female' gender, and '7 mi (11 km)' distance. The profile features a placeholder photo with 'Request Primary Photo' and 'Ask me for my Photo' buttons. Key information includes:

- Dances:** International Latin
- Goals:** Performance, Competition, Training/Practice
- Home Location:** Cliffside Park, New Jersey, United States of America, with a 'Willing to relocate' badge.
- Personal Details:** Proficiency: 6 (with 'Show', 'Pre-Champ', 'A2', 'Senior Staff' options), Height: 5'6" (168 cm), Body Type: Average, Covid Vaccination: Fully Vaccinated.

 A bio snippet at the bottom reads: 'I danced Latin competitively for 5 active years. I dance open gold level, and have been teaching dance for over 4 years now. I lov...'

Рисунок 1.3 – Результати пошуку партнерів для занять танцями у DancePartner

В Україні присутні також власні вебзастосунки для пошуку партнерів для танців.

DanceInfo (рис. 1.4) фактично представляє собою стіну оголошень, де кожен користувач може написати оголошення щодо пошуку партнера для танців [3]. Проблемою даного вебзастосунку можна вважати його застарілий дизайн та відповідно застарілі підходи. Усі користувачі представлені стандартними текстовими оголошеннями, відповідно здійснити пошук за

критеріями неможливо. Доступна інформація про клуби для занять танцями, проте фактично інформація взагалі не заповнена.

DanceInfo.com.ua
СПОРТИВНИЙ ТАНЕЦЬ В УКРАЇНІ СПОРТИВНИЙ ТАНЕЦЬ В УКРАЇНІ

ШУКАЮ ПАРТНЕРКУ

[добавить объявление](#)

1..15 :: [16..30](#) :: [31..45](#) :: [46..60](#) :: [61..75](#) :: [»»](#)

13.05.2025, 10:31
Назар ::
Шукаємо партнерку!
Шукаємо партнерку для серйозних занять танцями!
Про себе: м. Київ, зріст 174 см. Юніори 2, 10 танців, клуб StarDance.
Тел.: 067 793 88 64.

06.05.2025, 17:06
Партнер :: mirkap@ukr.net
Харків!!!! Терміново шукаю партнерку
Терміново шукаю партнерку на одну програму (латина).
Партнер: 2008 р.н (молодь 1)
Зріст 186 см, клас С (латина, стандарт)
т. 097-95-20-521

01.05.2025, 14:50
Ольга :: Okuz0410@gmail.com
Партнерка знайдись!
Шукаємо партнерку для Sgora.
2013 рік, Юніори 1.
N, E клас, зріст 150 см
Київ, метро Сирець.
Без переходу в інший клуб.

10.04.2025, 02:31
Анастасія ::
Київ, Боярка
Привіт!
Неспішно шукаю партнера, щоб повернутися до світу танцю. На першому місці задоволення, на другому - спортивні перемоги. Собі я вже давно все довела, але з радістю потішу себе гарними результатами і зараз)
Мені 21 рік, зріст 174 б/к. Кілька років тому завершила свою кар'єру на категорії C/Rt. Латина.
Швидко повернусь у форму, так як і зараз займаюсь спортом)
Про-Ам пропонувати по приколу можна, але не за всі гроші світу))
Писати на тг або вайбер сюди :
096-024-18-94
Всім успішних пошуків)

подписка на новости
Ваш e-mail

У ЯКІЙ ПОСЛІДОВНОСТІ Є
ДЛЯ ВАС ВИЗНАЧАЛЬНИМИ
ТРИ ФАКТОРА (Т., С., К.) ПРИ
ВИБОРІ УЧАСТІ У
ЗМАГАННЯХ:
Т. - ДУМКА ТРЕНЕРА
С. - НЕЗАЛЕЖНА
ПРЕДСТАВНИЦЬКА
СУДДІВСЬКА ПАНЕЛЬ
К. - КІЛЬКІСТЬ ТА
ПРЕДСТАВНИЦТВО
УЧАСНИКІВ

Т., С., К.
 Т., К., С.
 С., Т., К.
 С., К., Т.
 -

Рисунок 1.4 – Оголошення щодо пошуку партнерів для танців у DanceInfo

Стандартним інструментом пошуку партнерів для танців є мережа Facebook (рис. 1.5), де створюються відповідні групи, а в них самі користувачів

або адміністратори розміщують відповідні оголошення.

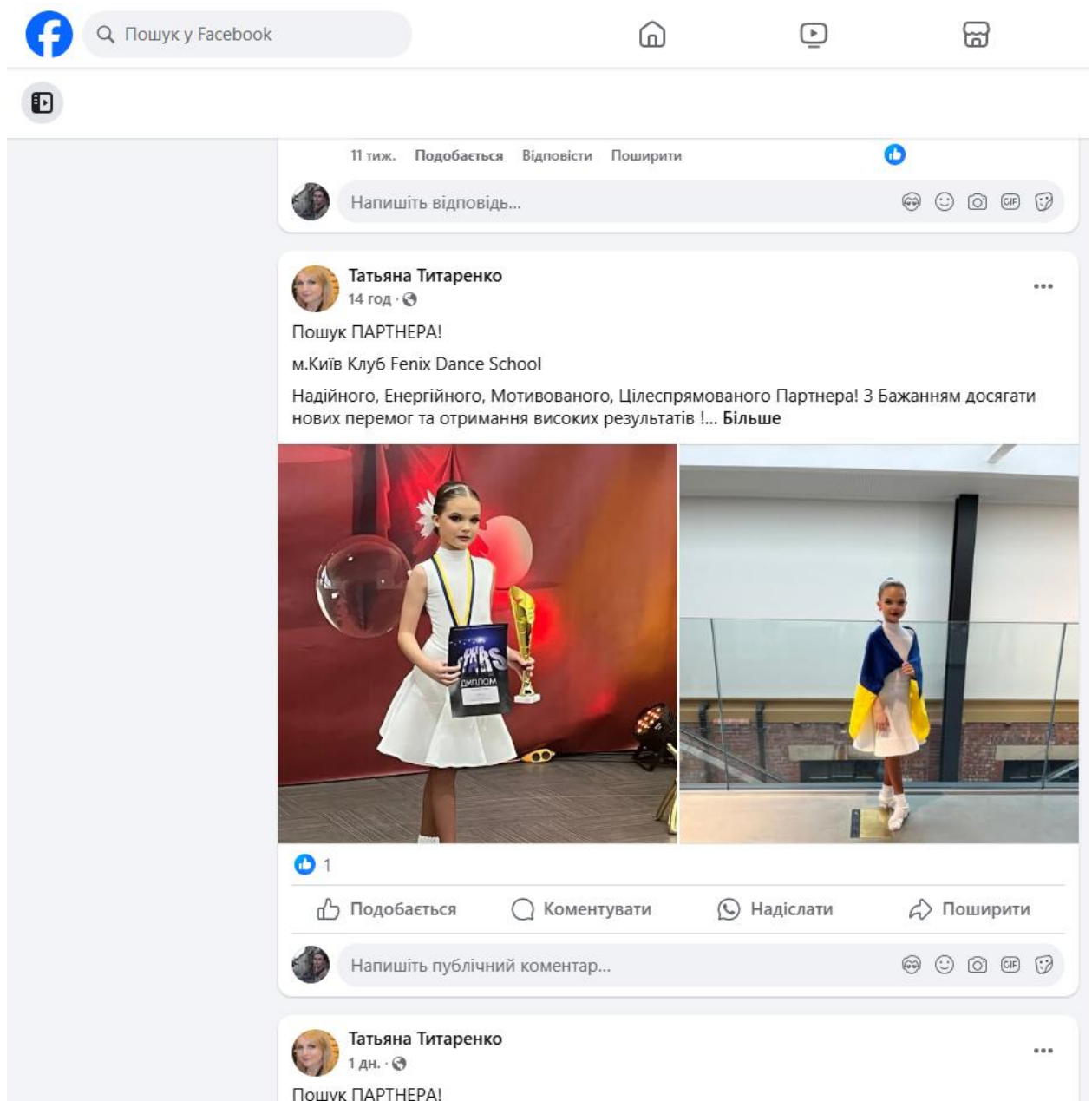


Рисунок 1.5 – Група для пошуку партнерів для танців у мережі Facebook

В Україні також представлений вебзастосунок Знакомка (рис. 1.6), який дозволяє виконувати пошук партнерів для тренажерних залів, для пробіжок, для велосипедних прогулянок, танців та гімнастики, футболу та хокею тощо [4]. Недоліком даного застосунку є те, що фільтрація відбувається тільки за містом. Ніяких інших засобів підбору варіантів під вимоги користувача немає.

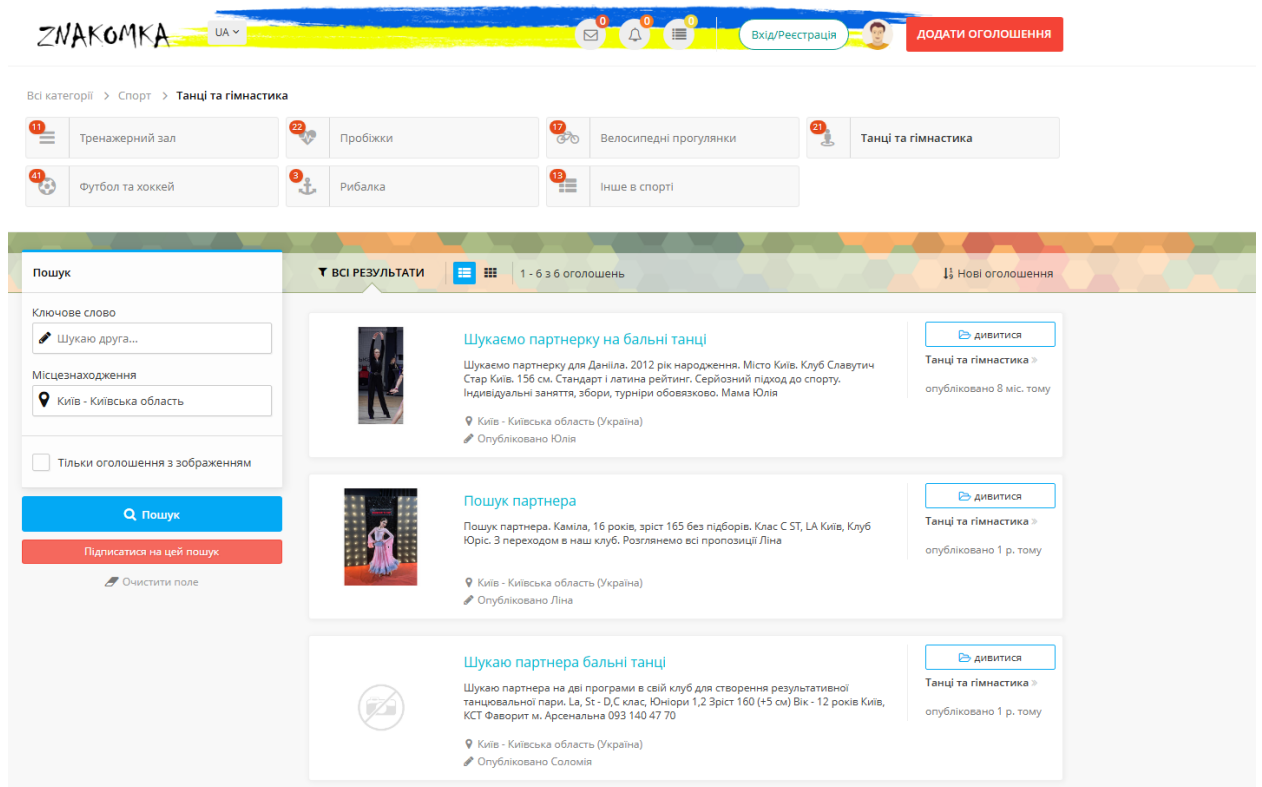


Рисунок 1.6 – Пошук партнерів для танців у вебзастосунку Znakomka

1.2 Формування концепції розроблюваної програми організації індивідуальних занять танцями

Як було зазначено в процесі огляду можливостей існуючих програм, які би могли застосовуватись під час організації індивідуальних занять танцями, вони зв'язані з організацією окремих етапів такої роботи та не охоплюють відповідний процес цілісно. Цілісне охоплення даного процесу могло б призвести до того, що більша кількість користувачів звертались би до програми, таким чином сприяючи наповненості даними вебзастосунку, що є однією важливою складовою успіху рішення, а з іншого боку – сприяючи задоволеності користувачів за рахунок пропонування широкого кола функцій, які б дозволили в підсумку саме організувати заняття стосовно місць проведення, часу, залучених учасників. Усе це має бути суміщено безпосередньо з самим знайомством, на встановлення якого направлення більша частина розглянутих аналогів. Тоді таке рішення буде витратити

менше додаткового часу людини, таким чином знову ж опосередковано сприяючи задоволеності від роботи.

Відповідно до основних критеріїв порівняння аналогів між собою та розробки було винесено можливості:

– пошук партнерів для танців: пошук може здійснюватися як безпосередньо з визначенням критеріїв пошуку в момент його реалізації, так і з визначенням власних значень за цими критеріями, тобто стосовно заповнення анкети, яка надає відповідну інформацію іншій стороні, як це реалізовано в DancePartner і має бути впроваджено в розробці, проте в DanceInfo цей засіб присутній виключно як статичний перелік оголошень про пошук партнерів для танців, окрім того самі оголошення зрідка містять детальну інформацію, яка дозволяє сформувавши суттєве враження;

– надання матеріалів про власний рівень: окрім критеріїв та відповідних власних значень за цими критеріями важливим елементом надання інформації про свій рівень може бути розміщення посилань на відео або іншим подібним чином представлення себе, однак у аналогах це взагалі не розглянуто, хоча і доступне, наприклад, у мережі Facebook при створення оголошення;

– розгляд майданчиків для занять танцями: пошук партнерів для танців стосовно індивідуальних, групових занять є важливим, однак важливо розуміти і те, де такі заняття будуть відбуватися, а тому корисним є одразу під час пошуку це зазначати, при цьому в Mix Styleце лише представлення власних майданчиків студій, тобто ширше коло не представлено;

– вибір тренерів для індивідуальних занять танцями: заняття нерідко відбуваються з тренерами, тому пошук може здійснювати в такому випадку взагалі пара, а тренери відповідно таким чином можуть додатково рекламувати свої можливості, до того ж додатково надаючи інформацію і про доступні у них майданчики, що може додатково допомогти новоствореним у застосунку парам одразу шукати потрібні їм варіанти для цього (табл. 1.1).

Таблиця 1.1 – Результати порівняння аналогів та розробки стосовно організації індивідуальних занять танцями

Критерій	DancePartner	DanceInfo	Mix Style	Розробка
Пошук партнерів для танців	За критеріями	Статичні оголошення	Відсутній	За критеріями
Надання матеріалів про власний рівень	Відсутнє	Відсутнє	Відсутнє	Так
Вибір тренерів для індивідуальних занять танцями	Відсутній	Відсутній	3 тренерів студії для загальних занять	Так
Розгляд майданчиків для занять танцями	Відсутній	Розділ присутній, але ненаповнений	Студії мережі	Так

1.3 Висновки за розділом 1

Важливим результатом виконання даного розділу є визначення концепції розроблюваної програми організації індивідуальних занять танцями, яка відповідно повинна дозволити їй суттєво відрізнитися від наявних аналогів у першу чергу через організацію підтримки цілісно зазначеного процесу. Тобто програма не тільки виконувати пошук партнерів для танців, але і роблячи акцент на можливостях індивідуальних занять, інтегрувати пошук партнерів з пошуком тренерів, майданчиків.

2 ПРОЄКТУВАННЯ ПРОГРАМИ

2.1 Визначення функціональних вимог користувачів до програми організації індивідуальних занять танцями

Спочатку було визначено повний перелік вимог ПЗ організації індивідуальних занять танцями:

- визначення власного опису смаків та досвіду в танцях;
- пошук партнерів для занять танцями;
- розміщення доступного варіанту локації для занять танцями у танцівника, тренера, у тому числі приватної локації;
- відтворення анкети танцівника;
- пошук локацій для занять танцями;
- відтворення доступних варіантів локацій для занять танцями у танцівника, тренера;
- визначення тренерських характеристик;
- відтворення сторінки тренера для занять танцями;
- визначення варіантів індивідуальних занять танцями у тренера;
- відтворення доступних варіантів індивідуальних занять танцями у тренера;
- подання заявки та керування заявкою стосовно конкретного індивідуального заняття танцями у тренера;
- відтворення графіку запланованих індивідуальних занять танцями тренера;
- відтворення заявок за індивідуальними заняттями танцями тренера;
- пошук тренерів для занять танцями;
- комплексний пошук партнера, локації та тренера для занять танцями;
- розміщення матеріалів за танцювальним досвідом;
- створення пари для індивідуальних занять танцями;
- керування заявками стосовно створення пар для індивідуальних занять танцями;

- закріплення тренера для індивідуальних занять танцями пари або танцівника та його скасування;
- закріплення локації для індивідуальних занять танцями пари або танцівника та скасування;
- підтримування танцівника, тренера або скасування такої підтримки;
- відтворення даних підтриманих танцівників, тренерів користувачем;
- коментування анкети танцівника;
- відтворення даних прокоментованих танцівників користувачем;
- коментування сторінки тренера;
- відтворення даних прокоментованих тренерів користувачем;
- реєстрація танцівників та тренерів;
- відтворення коментарів за тренером або танцівником;
- визначення слідкування за танцівником, тренером або його скасування;
- відтворення даних танцівників, тренерів, за якими слідкує користувач;
- відтворення доступних варіантів індивідуальних занять танцями у власних тренерів;
- планування індивідуального заняття танцями пари;
- відтворення пар та танцівників, тренування яких закріплено за тренером;
- відтворення графіку індивідуальних занять танцями пари;
- авторизація танцівників та тренерів.

Ці вимоги включають ряд сценаріїв роботи користувачів, при цьому частина з них декілька сценаріїв одразу. Користувачі, які працюють з програмою організації індивідуальних занять танцями, є або неавторизованими користувачами, або танцівниками, або тренерами. Додатковою роллю є адміністратор, однак ця роль є стандартною щодо роботи зі всіма наявними у програмі даними стосовно внесення даних, перегляду, редагування та вилучення таких даних, тобто операцій Create, Read, Update і

Delete (CRUD), що має бути в подальшому враховано через стандартний інтерфейс.

Відповідно набір сценаріїв, промодельованих для неавторизованого користувача:

- пошук партнерів для занять танцями;
- відтворення анкети танцівника;
- пошук локацій для занять танцями;
- відтворення доступних варіантів локацій для занять танцями у танцівника;
- відтворення доступних варіантів локацій для занять танцями у тренера з танців;
- відтворення сторінки тренера для занять танцями;
- відтворення доступних варіантів індивідуальних занять танцями у тренера;
- пошук тренерів для занять танцями;
- відтворення коментарів за тренером з танців;
- відтворення коментарів за танцівником;
- авторизація (рис. 2.1).

Альтернативний сценарій у ПЗ організації індивідуальних занять танцями для неавторизованого користувача – реєстрація, що застосовується стосовно сценарію авторизації, коли користувач не має облікового запису танцівника або тренера, відповідно виконати саму авторизацію не можу, а тому йому надається можливість реєстрації, яка призведе далі до авторизації вже створеного облікового запису.

Танцівники є користувачами вже після авторизації, які були відповідним чином зареєстровані спочатку та яким надаються такі сценарії окрім тих, що належать неавторизованому користувачу:

- визначення власного опису смаків та досвіду в танцях;
- розміщення доступного власного варіанту локації для занять танцями;

- подання заявки стосовно конкретного індивідуального заняття танцями у тренера;
- комплексний пошук партнера, локації та тренера для занять танцями: винесення даного сценарію тільки для танцівників обумовлено тим, що дана функція потребує достатньо великої кількості ресурсів, тому це додатково може обтяжувати сервер і використовуватися зловмисно під час роботи, а тому було вирішено вимагати попередньої реєстрації і авторизації перед тим, як її використовувати.

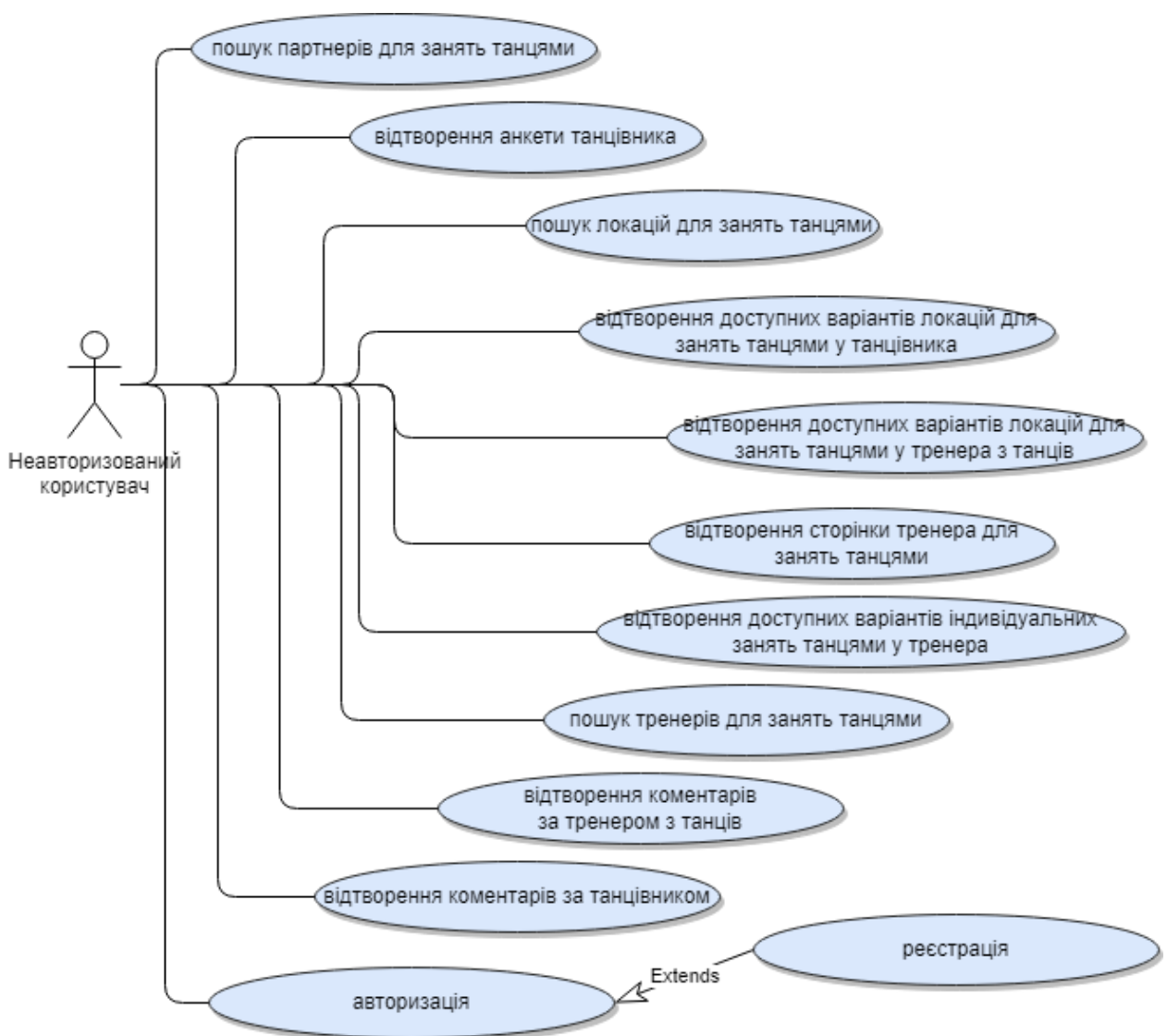


Рисунок 2.1 – Діаграма use case ПЗ організації індивідуальних занять танцями для неавторизованого користувача

Окрім того основні сценарії танцівника:

- розміщення матеріалів за танцювальним досвідом, включаючи створення записів про такі матеріали;
- створення пари для індивідуальних занять танцями, включаючи сценарії подання заявки на створення пари для індивідуальних занять танцями, відтворення поданих заявок, створених пар;
- керування заявками стосовно створення пар для індивідуальних занять танцями;
- закріплення тренера для індивідуальних занять танцями пари;
- закріплення тренера для індивідуальних занять танцями танцівника;
- закріплення локації для індивідуальних занять танцями танцівника;
- закріплення локації для індивідуальних занять танцями пари;
- підтримування танцівника;
- підтримування тренера з танців;
- відтворення даних підтриманих танцівників користувачем;
- відтворення даних підтриманих тренерів користувачем;
- коментування анкети танцівника;
- відтворення даних прокоментованих танцівників;
- коментування сторінки тренера з танців;
- відтворення даних прокоментованих тренерів;
- визначення слідування за танцівником;
- визначення слідування за тренером з танців;
- відтворення даних танцівників, за якими слідує користувач;
- відтворення даних тренерів, за якими слідує користувач;
- відтворення доступних варіантів індивідуальних занять танцями у власних тренерів;
- планування індивідуального заняття танцями пари;
- відтворення графіку індивідуальних занять танцями пари.

Серед сценаріїв танцівника є ті, які стосуються занять у парі, а є ті, які стосуються індивідуальних танців. Це пов'язано з тим, що можуть бути

користувачі, які займаються і парними, і індивідуальними танцями, окрім того вони можуть мати різні пари для різних видів танців, якщо в цьому є необхідність, тому над сценаріями фактично існують різні варіанти фільтрування, включаючи вибирання того, чи це індивідуальні танці, чи парні, а також до якої саме пари певна функціональність належить.

Альтернативні сценарії для танцівника:

– скасування доступного власного варіанту локації для занять танцями як альтернатива розміщенню доступного власного варіанту локації для занять танцями, коли цей варіант локації для занять танцями, який був визначений, більше не доступний;

– редагування матеріалу за танцювальним досвідом як альтернатива розміщенню матеріалу за танцювальним досвідом, коли танцівник цей матеріал за танцювальним досвідом вніс, але необхідно внести зміни;

– вилучення матеріалу за танцювальним досвідом як альтернатива розміщенню матеріалу за танцювальним досвідом, коли танцівник цей матеріал за танцювальним досвідом вніс, але більше він не потрібен;

– скасування власної заявки стосовно конкретного індивідуального заняття танцями у тренера як альтернатива поданню заявки стосовно конкретного індивідуального заняття танцями у тренера, коли створена заявка стосовно конкретного індивідуального заняття танцями у тренера більше не вважається актуальною;

– скасування пари для індивідуальних занять танцями як альтернатива створення пари для індивідуальних занять танцями, коли пара вже була створена, але більше не є такою;

– відхилення пари для індивідуальних занять танцями як альтернатива створення пари для індивідуальних занять танцями, коли пара ще не була створена, заявка на створення пари для індивідуальних занять танцями вже визначена, але інша сторона взаємодії не бажає позитивно відповідати на подану заявку;

– скасування закріплення тренера для індивідуальних занять танцями

пари як альтернатива закріплення тренера для індивідуальних занять танцями пари, коли таке закріплення тренера для індивідуальних занять танцями пари відбулось, але більше танцівники не підтримують таке закріплення;

– скасування закріплення тренера для індивідуальних занять танцями танцівника як альтернатива закріплення тренера для індивідуальних занять танцями танцівника, коли таке закріплення тренера для індивідуальних занять танцями танцівника відбулось, але більше танцівник не підтримує таке закріплення;

– скасування закріплення локації для індивідуальних занять танцями пари як альтернатива закріплення локації для індивідуальних занять танцями пари, коли закріплення локації для індивідуальних занять танцями пари відбулось, але більше пара не підтримує таке закріплення;

– скасування закріплення локації для індивідуальних занять танцями танцівника як альтернатива закріплення локації для індивідуальних занять танцями танцівника, коли закріплення локації для індивідуальних занять танцями танцівника відбулось, але більше танцівник не підтримує таке закріплення;

– скасування підтримування танцівника як альтернатива підтримуванню танцівника, коли підтримування одного танцівника іншим відбулось, але помилково;

– скасування підтримування тренера з танців як альтернатива підтримуванню тренера з танців, коли підтримування тренера танцівником відбулось, але помилково;

– скасування визначення слідкування за танцівником як альтернатива визначення слідкування за танцівником, коли цей танцівник за відповідним іншим танцівником вже слідкує, але більше це не потрібно;

– скасування визначення слідкування за тренером з танців як альтернатива визначення слідкування за тренером з танців, коли цей танцівник за цим тренером з танців вже слідкує, але більше це не потрібно (рис. 2.2).

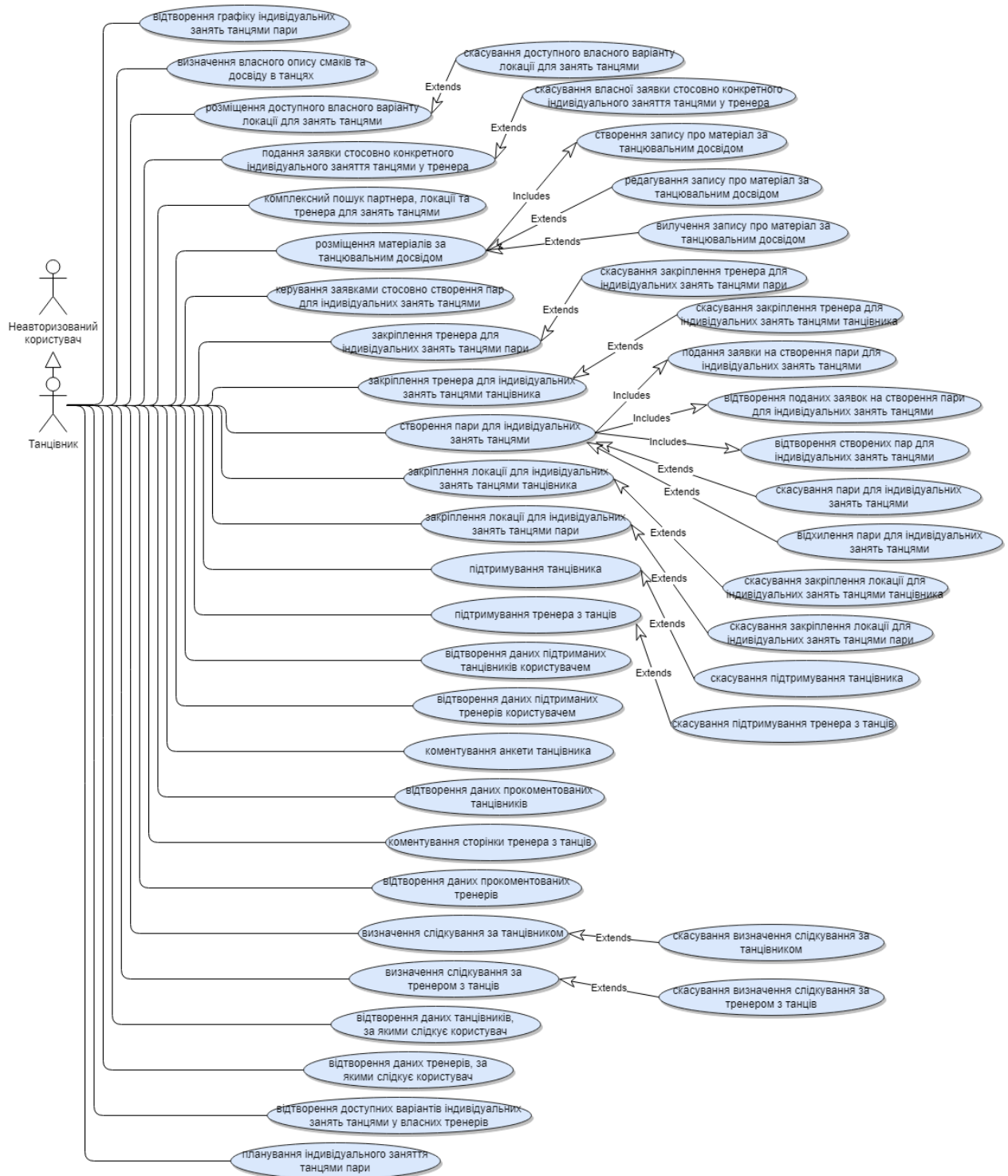


Рисунок 2.2 – Діаграма use case ПЗ організації індивідуальних занять танцями для танцівника

Тренери з танців є користувачами вже після авторизації, які були відповідним чином зареєстровані спочатку та яким надаються такі сценарії окрім тих, що належать неавторизованому користувачу:

– розміщення доступного варіанту локації для занять танцями у тренера

з танців;

- визначення тренерських характеристик;
- визначення варіантів індивідуальних занять танцями у тренера;
- керування поданою заявкою стосовно конкретного індивідуального заняття танцями цього тренера;
- відтворення графіку запланованих індивідуальних занять танцями цього тренера;
- відтворення заявок за індивідуальними заняттями танцями цього тренера;
- відтворення пар та танцівників, тренування яких закріплено за цим тренером;
- підтримування танцівника;
- відтворення даних підтриманих танцівників;
- підтримування тренера з танців;
- відтворення даних підтриманих тренерів з танців;
- коментування анкети танцівника;
- відтворення даних прокоментованих танцівників;
- коментування сторінки тренера;
- відтворення даних прокоментованих тренерів з танців.

Альтернативні сценарії для тренера з танців:

- скасування підтримування танцівника як альтернатива підтримуванню танцівника, коли підтримування танцівника тренером з танців відбулось, але помилково;
- скасування підтримування тренера з танців як альтернатива підтримуванню тренера з танців, коли підтримування тренера з танців іншим тренером відбулось, але помилково;
- вилучення варіанта індивідуальних занять танцями у тренера як альтернатива визначення варіантів індивідуальних занять танцями у тренера, коли такий варіант більше не доступний, хоча і був визначений;
- скасування закріплення тренера для індивідуальних занять танцями

пари як альтернатива відтворення пар та танцівників, тренування яких закріплено за цим тренером, коли таке закріплення тренера для індивідуальних занять танцями пари відбулось зі сторони пари, але більше тренер з танців не підтримує таке закріплення або він початково це не підтримує;

– скасування закріплення тренера для індивідуальних занять танцями танцівника як альтернатива відтворення пар та танцівників, тренування яких закріплено за цим тренером, коли таке закріплення тренера для індивідуальних занять танцями танцівника відбулось зі сторони танцівника, але більше тренер з танців не підтримує таке закріплення або він початково це не підтримує (рис. 2.3).

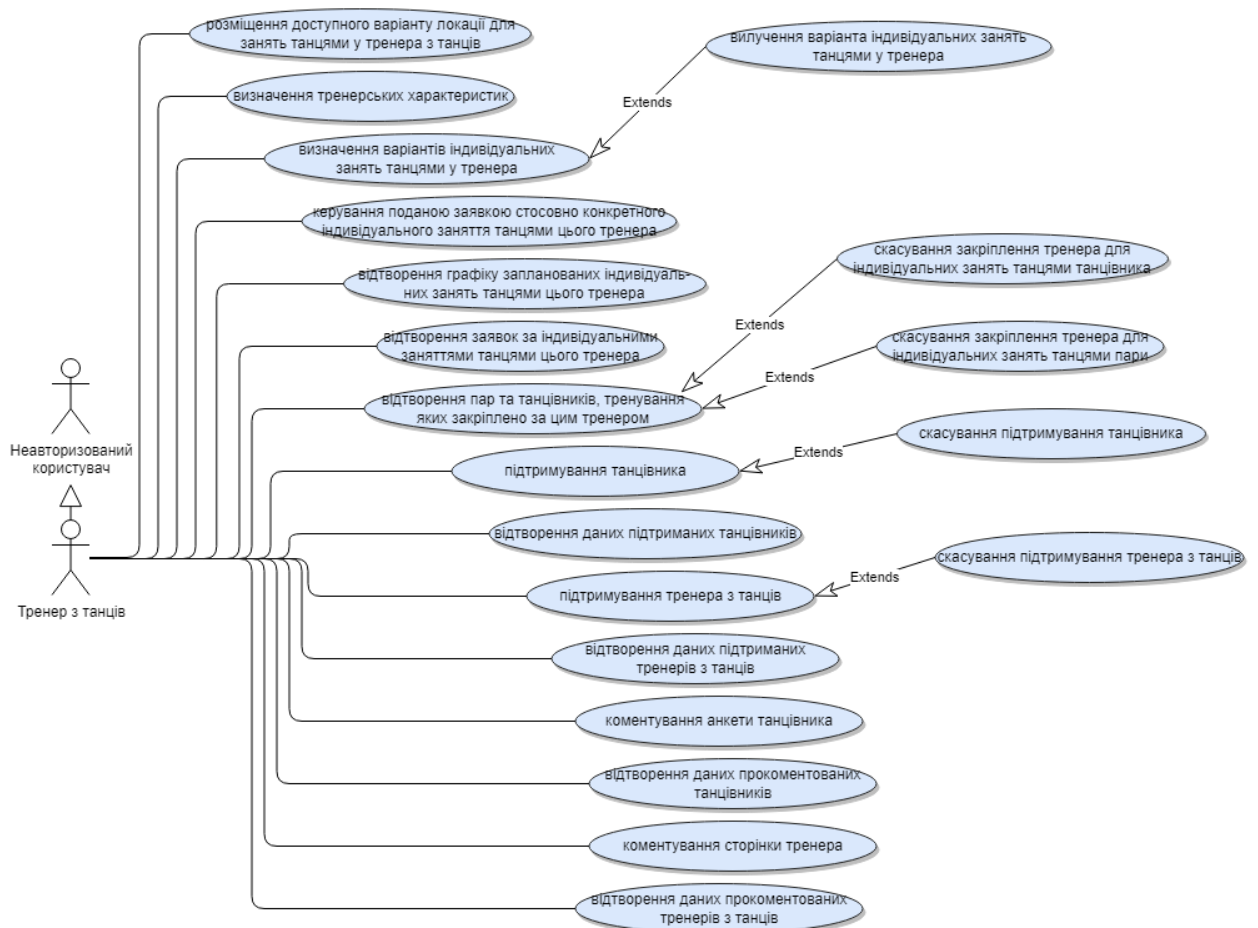


Рисунок 2.3 – Діаграма use case ПЗ організації індивідуальних занять танцями для тренера з танців

2.2 Визначення засобів для розробки програми організації індивідуальних занять танцями

Засоби для розробки програми організації індивідуальних занять танцями, серед яких здійснювалось остаточний вибір, включали фреймворки веброботки Django [5]-[6] і Laravel [7]-[8] (табл. 2.1).

Таблиця 2.1 – Порівняння засобів для розробки програми організації індивідуальних занять танцями

Критерій	Laravel	Django
Мова програмування	PHP	Python з простішим синтаксисом
Продуктивність	Стандартна	За релевантного використання засобів краща
Підтримка баз даних	Ефективна	Ефективна
Архітектурне рішення	Model-View-Controller	Model-View-Template
Масштабованість	Високий рівень	Вищий рівень за рахунок можливостей мови програмування
Безпека роботи і даних	Для відповідності потребує довгого налаштування	Краща за стандартних засобів
Засоби тестування	Зручні	Зручні

Вибір Django і Laravel обумовлений активністю застосування цих рішень для веброботки, а також тим, що вони представляють різні мови програмування, що дозволяє таким чином одразу порівнювати і мови

програмування. Перевага і вибір Django пов'язані з тим, що Django базується на мові з простішим синтаксисом, що дозволяє її ефективніше застосовувати в умовах обмежень, має вищу продуктивність, кращий рівень безпеки пропонує засобів та високу масштабованість у першу чергу за рахунок універсальності мови програмування, яка дозволяє легко додавати засоби з суміжних областей.

2.3 Структура бази даних програми організації індивідуальних занять танцями

База даних базувалась на тому, що для керування нею було обрано систему PostgreSQL, оскільки вона є більш підходящою порівняно, наприклад, з MySQL при роботі зі складними запитамі, запитамі на внесення даних, а не тільки читання, при цьому належить світу вільного ПЗ. Також база даних базувалась на структурі, що подана нижче.

Таблиця Dancecity визначає міста, де можуть знаходитись танцівники, локації для занять танцями, з такими полями:

- id – ідентифікатор міста;
- locationcountry – країна, де це місто є;
- dancecityname – назва міста.

Таблиця Danceuser визначає танцівників з обліковими записами користувачів з такими полями:

- id – ідентифікатор танцівника;
- useraccountindjango – обліковий запис у Django за тим користувачем, який є цим танцівником, маючи відношення до таблиці User (не виділена окремо як стандартна таблиця зі структури бази даних будь-якого Django-застосунку) один-до-одного;
- danceexperiencesince – рік, з якого почав займатися танцями цей танцівник;

– `dancelevel` – рівень танцювальних вмінь, на який себе оцінює цей танцівник;

– `dancername` – ім'я танцівника у програмі;

– `dancergender` – стать цього танцівника;

– `dancersearchinggender` – стать напарника для танців, якого шукає цей танцівник: якщо стать не зазначена, то це означає, що танцівник не шукає партнера у програмі;

– `dancecity` – місто, в якому проживає і безпосередньо шукає партнера для танців цей танцівник, визначений ідентифікатором з таблиці `Dancecity` через зв'язок «один-до-багатьох»;

– `dancesearchkm` – відстань у кілометрах, на якій готовий шукати партнера для танців цей танцівник;

– `dancerbirth` – дата народження цього танцівника;

– `dancerheight` – зріст цього танцівника;

– `dancerweight` – вага цього танцівника;

– `dancerbody` – структура тіла цього танцівника;

– `dancerhaircolor` – колір волосся цього танцівника;

– `dancerhaircut` – зачіска цього танцівника;

– `dancereyecolor` – колір очей цього танцівника;

– `dancerrelationship` – наявність відносин у цього танцівника;

– `dancersmoking` – звичка палити у цього танцівника;

– `dancerdrinking` – звичка вживати алкоголь у цього танцівника;

– `dancerethnicity` – етнічне походження цього танцівника;

– `danceropenness` – відкритість до стосунків цього танцівника з партнерами по танцям;

– `dancercharacterotype` – тип характеру цього танцівника;

– `dancereducation` – рівень освіти цього танцівника;

– `dancerincome` – загальний рівень доходу цього танцівника;

– `dancerreligion` – релігія цього танцівника;

– `danceroccupation` – професія в житті цього танцівника.

Таблиця Danceuserphoto визначає фотографії танцівника з такими полями:

- id – ідентифікатор фотографії танцівника;
- danceuser – танцівник, який є на фотографії, визначений ідентифікатором з таблиці Danceuser через зв'язок «один-до-багатьох»;
- danceuserphoto – фотографія, де є цей танцівник (вміст).

Таблиця Dancestylecategory визначає категорії стилів танців з такими полями:

- id – ідентифікатор категорії стилю танців;
- dancestylecategoryname – назва категорії стилю танців.

Таблиця Dancestyle визначає стилі танців з такими полями:

- id – ідентифікатор стилю танців;
- dancestylecategory – категорія, до якої належить цей стиль танцю, визначений ідентифікатором з таблиці Dancestylecategory через зв'язок «один-до-багатьох»;
- dancestylename – назва стилю танців.

Таблиця Danceuserstyle визначає стилів танців, які вміє танцювати або яким навчається танцівник, з такими полями:

- id – ідентифікатор стилю танців, який вміє танцювати або якому навчається танцівник;
- danceuser – танцівник, який вміє танцювати або навчається танцювати цей стиль танців, визначений ідентифікатором з таблиці Danceuser через зв'язок «один-до-багатьох»;
- dancestyle – стиль танцю, який вміє танцювати танцівник або який танцювати він навчається, визначений ідентифікатором з таблиці Dancestyle через зв'язок «один-до-багатьох»;
- dancelevel – рівень танцювальних вмінь, якими оцінює цей танцівник свої вміння стосовно цього стилю танців.

Таблиця Dancegoal визначає цілі навчання танцям або танцювання з такими полями:

- id – ідентифікатор цілі навчання танцям або танцювання;
- dancegoalname – назва цілі навчання танцям або танцювання.

Таблиця Danceusergoal визначає цілі навчання танцям або танцювання танцівника з такими полями:

- id – ідентифікатор цілі навчання танцям або танцювання танцівника;
- dancegoal – ціль навчання танцям або танцювання цього танцівника, визначена ідентифікатором з таблиці Dancegoal через зв'язок «один-до-багатьох»;

- danceuser – танцівник, який має цю ціль навчання танцям або танцювання, визначений ідентифікатором з таблиці Danceuser через зв'язок «один-до-багатьох».

Таблиця Dancepair визначає танцювальні пари та запити щодо формування таких пар для індивідуальних занять танцями з такими полями:

- id – ідентифікатор танцювальної пари або запиту щодо формування такої пари для індивідуальних занять танцями;

- danceuser1 – танцівник, який створив запит щодо формування пари для індивідуальних занять танцями, визначений ідентифікатором з таблиці Danceuser через зв'язок «один-до-багатьох»;

- danceuser2 – танцівник, до якого було адресовано запит щодо формування пари для індивідуальних занять танцями, визначений ідентифікатором з таблиці Danceuser через зв'язок «один-до-багатьох»;

- dancepairstatus – стан запиту щодо формування такої пари для індивідуальних занять танцями;

- dancepairdatecreation – час подання запиту щодо формування пари для індивідуальних занять танцями;

- dancepairdateapplication – час відповіді на запит щодо формування пари для індивідуальних занять танцями.

Таблиця Danceplace визначає локації для навчання танцям з такими полями:

- id – ідентифікатор локації для навчання танцям;

- `danceplacetype` – тип локації для навчання танцям;
- `danceplacename` – назва локації для навчання танцям;
- `dancecity` – місто, в якому ця локація для навчання танцям є, визначена ідентифікатором з таблиці `Dancecity` через зв'язок «один-до-багатьох»;
- `danceplaceaddress` – адреса локації для навчання танцям;
- `danceplacephoto` – фотографія локації для навчання танцям;
- `danceplaceprivate` – визначення того, чи є ця локація для навчання танцям приватною, чи публічною.

Таблиця `Danceplace` визначає локації, які доступні для занять танцями у танцівника, з такими полями:

- `id` – ідентифікатор локації, яка доступна для занять танцями у танцівника;
- `danceuser` – танцівник, якому доступна для занять танцями ця локація, визначений ідентифікатором з таблиці `Danceuser` через зв'язок «один-до-багатьох»;
- `danceplace` – локація для занять танцями, доступна цьому танцівнику, визначена ідентифікатором з таблиці `Danceplace` через зв'язок «один-до-багатьох»;
- `danceplacedatecreation` – дата визначення цієї локації як доступної для занять танцями у цього танцівника.

Таблиця `Danceteacher` визначає тренерів з танців з обліковими записами користувачів з такими полями:

- `id` – ідентифікатор цього тренера з танців;
- `useraccountindjango` – обліковий запис у Django за тим користувачем, який є цим тренером з танців, маючи відношення до таблиці `User` (не виділена окремо як стандартна таблиця зі структури бази даних будь-якого Django-застосування) один-до-одного;
- `danceexperiencesince` – рік, з якого почав займатися танцями цей тренер з танців;

– danceteachingsince – рік, з якого почав працювати тренером цей тренер з танців;

– dancelevel – рівень танцювальних вмінь, на який себе оцінює цей тренер з танців;

– dancetname – ім'я цього тренера з танців;

– dancetgender – стать цього тренера з танців;

– dancetcity – місто, в якому працює тренером з танців цей користувач, визначений ідентифікатором з таблиці Dancetcity через зв'язок «один-до-багатьох»;

– dancetbirth – дата народження цього тренера з танців;

– dancetethnicity – етнічне походження цього тренера з танців;

– dancetheractertype – тип характеру цього тренера з танців;

– danceteducation – рівень освіти цього тренера з танців;

– dancetreligion – релігія цього тренера з танців.

Таблиця Danceteacherphoto визначає фотографії тренерів з танців з такими полями:

– id – ідентифікатор фотографії тренера з танців;

– danceteacher – тренер з танців, який є на фотографії, визначений ідентифікатором з таблиці Danceteacher через зв'язок «один-до-багатьох»;

– dancetphoto – фотографія, де є цей тренер з танців (вміст).

Таблиця Danceteacherstyle визначає стилі танців, яким може навчати тренер з танців, з такими полями:

– id – ідентифікатор стилю танців, якому може навчати тренер з танців;

– danceteacher – тренер з танців, який може навчати цьому стилю з танців, визначений ідентифікатором з таблиці Danceteacher через зв'язок «один-до-багатьох»;

– dancestyle – стиль танців, якому може навчати цей тренер з танців, визначений ідентифікатором з таблиці Dancestyle через зв'язок «один-до-багатьох»;

– `danceexperiencesince` – рік, з якого повноцінно вміє танцювати цей танець цей тренер з танців;

– `dancelevel` – танцювальний рівень цього тренера з танців за цим стилем танців за його власним враженням.

Таблиця `Danceteacherplace` визначає локації, де можуть навчати танцям тренери з танців, з такими полями:

– `id` – ідентифікатор локації, доступної для навчання танцям цьому тренеру з танців;

– `danceteacher` – тренер з танців, якому доступна для навчання танцям ця локація, визначений ідентифікатором з таблиці `Danceteacher` через зв'язок «один-до-багатьох»;

– `danceplace` – локація, доступна для навчання танцям цьому тренеру з танців, визначена ідентифікатором з таблиці `Danceplace` через зв'язок «один-до-багатьох»;

– `danceplacedatecreation` – дата, з якої ця локація доступна для навчання танцям цьому тренеру з танців.

Таблиця `Danceteacherlesson` визначає індивідуальні заняття танцями у тренера з танців з такими полями:

– `id` – ідентифікатор індивідуального заняття танцями у тренера з танців;

– `danceteacher` – тренер з танців, у якого доступне це індивідуальне заняття танцями, визначений ідентифікатором з таблиці `Danceteacher` через зв'язок «один-до-багатьох»;

– `dancelessondate` – дата, коли це індивідуальне заняття танцями має відбуватися;

– `dancelessonduration` – тривалість цього індивідуального заняття танцями;

– `dancelessonstatus` – стан цього індивідуального заняття танцями: повністю зайняте, частково зайняте або вільне;

– `dancelessonindividualness` – визначення того, чи це індивідуальне заняття танцями, чи воно доступне для різних учасників;

– `dancelessonprice` – вартість відвідування цього індивідуального заняття танцями;

– `danceplace` – локація, де має відбуватися це індивідуальне заняття танцями, визначена ідентифікатором з таблиці `Danceplace` через зв'язок «один-до-багатьох».

Таблиця `Danceteachercomment` визначає коментарі тренерів з танців про інших тренерів з танців з такими полями:

– `id` – ідентифікатор коментаря тренера з танців про іншого тренера з танців;

– `danceteacher1` – тренер з танців, про якого написано коментар іншим тренером з танців, визначений ідентифікатором з таблиці `Danceteacher` через зв'язок «один-до-багатьох»;

– `danceteacher2` – тренер з танців, який написав коментар про іншого тренера з танців, визначений ідентифікатором з таблиці `Danceteacher` через зв'язок «один-до-багатьох»;

– `dancetcommenttext` – текст коментаря тренера з танців про іншого тренера з танців;

– `dancetcommentdate` – час створення коментаря тренера з танців про іншого тренера з танців.

Таблиця `Danceteacherusercomment` визначає коментарі танцівників про тренерів з танців з такими полями:

– `id` – ідентифікатор коментаря танцівника про тренера з танців;

– `danceteacher` – тренер з танців, про якого написано коментар танцівником, визначений ідентифікатором з таблиці `Danceteacher` через зв'язок «один-до-багатьох»;

– `danceuser` – танцівник, який написав коментар про тренера з танців, визначений ідентифікатором з таблиці `Danceuser` через зв'язок «один-до-багатьох»;

- `dancetcommenttext` – текст коментаря танцівника про тренера з танців;
- `dancetcommentdate` – час створення коментаря танцівника про тренера з танців.

Таблиця `Danceusercomment` визначає коментарі танцівників про інших танцівників з такими полями:

- `id` – ідентифікатор коментаря танцівника про іншого танцівника;
- `danceuser1` – танцівник, про якого інший танцівник створив коментар, визначений ідентифікатором з таблиці `Danceuser` через зв'язок «один-до-багатьох»;
- `danceuser2` – танцівник, який створив коментар про іншого танцівника, визначений ідентифікатором з таблиці `Danceuser` через зв'язок «один-до-багатьох»;
- `danceucommenttext` – текст коментаря танцівника про іншого танцівника;
- `danceucommentdate` – час створення коментаря танцівника про іншого танцівника.

Таблиця `Danceuserteachercomment` визначає коментарі тренерів з танців про танцівників з такими полями:

- `id` – ідентифікатор коментаря тренера з танців про танцівника;
- `danceuser` – танцівник, про якого тренер з танців створив коментар, визначений ідентифікатором з таблиці `Danceuser` через зв'язок «один-до-багатьох»;
- `danceteacher` – тренер з танців, який створив коментар про танцівника, визначений ідентифікатором з таблиці `Danceteacher` через зв'язок «один-до-багатьох»;
- `danceucommenttext` – текст коментаря тренера з танців про танцівника;
- `danceucommentdate` – час створення коментаря тренера з танців про танцівника.

Таблиця Danceteacherlessonpair визначає відвідування або заявки на відвідування індивідуальних занять з танців тренера танцювальною парою з такими полями:

- id – ідентифікатор відвідування або заявки на відвідування індивідуальних занять з танців тренера танцювальною парою;

- danceteacherlesson – індивідуальне заняття з танців тренера, яке має відвідати танцювальна пара, визначене ідентифікатором з таблиці Danceteacherlesson через зв'язок «один-до-багатьох»;

- danceplace – локація, де має відбуватися це індивідуальне заняття з танців тренера (сам варіант заняття може не містити конкретну локацію, в такому випадку танцювальна пара може запропонувати свою локацію, тому це поле присутнє в цій таблиці), визначена ідентифікатором з таблиці Danceplace через зв'язок «один-до-багатьох»;

- dancerpair – танцювальна пара, яка взялась відвідати це індивідуальне заняття з танців тренера, визначена ідентифікатором з таблиці Dancerpair через зв'язок «один-до-багатьох»;

- danceattendstatus – стан заявки на відвідування індивідуального заняття з танців тренера танцювальною парою.

Таблиця Danceteacherlessondancer визначає відвідування або заявки на відвідування індивідуальних занять з танців тренера танцівником з такими полями:

- id – ідентифікатор відвідування або заявки на відвідування індивідуальних занять з танців тренера танцівником;

- danceteacherlesson – індивідуальне заняття з танців тренера, яке має відвідати танцівник, визначене ідентифікатором з таблиці Danceteacherlesson через зв'язок «один-до-багатьох»;

- danceplace – локація, де має відбуватися це індивідуальне заняття з танців тренера (сам варіант заняття може не містити конкретну локацію, в такому випадку танцівник може запропонувати свою локацію, тому це поле

присутнє в цій таблиці), визначена ідентифікатором з таблиці Danceplace через зв'язок «один-до-багатьох»;

– danceuser – танцівник, який взявся відвідати це індивідуальне заняття з танців тренера, визначений ідентифікатором з таблиці Danceuser через зв'язок «один-до-багатьох»;

– danceattendstatus – стан заявки на відвідування індивідуального заняття з танців тренера танцівником.

Таблиця Danceuserfollow визначає слідкування за танцівником з такими полями:

– id – ідентифікатор слідкування за танцівником;

– danceuser1 – танцівник, слідкування за яким вирішив визначити інший танцівник, визначений ідентифікатором з таблиці Danceuser через зв'язок «один-до-багатьох»;

– danceuser2 – танцівник, який вирішив слідкувати за іншим танцівником, визначений ідентифікатором з таблиці Danceuser через зв'язок «один-до-багатьох».

Таблиця Danceteacherfollow визначає слідкування за тренером з танців танцівником з такими полями:

– id – ідентифікатор слідкування за тренером з танців танцівником;

– danceteacher – тренер з танців, слідкування за яким вирішив визначити танцівник, визначений ідентифікатором з таблиці Danceteacher через зв'язок «один-до-багатьох»;

– danceuser – танцівник, який вирішив слідкувати за цим тренером з танців, визначений ідентифікатором з таблиці Danceuser через зв'язок «один-до-багатьох».

Таблиця Danceusersupport визначає підтримування танцівниками інших танцівників з такими полями:

– id – ідентифікатор підтримування танцівником іншого танцівника;

– danceuser1 – танцівник, якого підтримав інший танцівник, визначений ідентифікатором з таблиці Danceuser через зв'язок «один-до-багатьох»;

– `danceuser2` – танцівник, який підтримав іншого танцівника, визначений ідентифікатором з таблиці `Danceuser` через зв'язок «один-до-багатьох».

Таблиця `Danceuserteachersupport` визначає підтримування тренерами з танців танцівників з такими полями:

- `id` – ідентифікатор підтримування тренером з танців танцівника;
- `danceuser` – танцівник, якого підтримав тренер з танців, визначений ідентифікатором з таблиці `Danceuser` через зв'язок «один-до-багатьох»;
- `danceteacher` – тренер з танців, який підтримав цього танцівника, визначений ідентифікатором з таблиці `Danceteacher` через зв'язок «один-до-багатьох».

Таблиця `Danceteachersupport` визначає підтримування тренерів з танців танцівниками з такими полями:

- `id` – ідентифікатор підтримування тренером з танців танцівника;
- `danceteacher` – тренер з танців, якого підтримав танцівник, визначений ідентифікатором з таблиці `Danceteacher` через зв'язок «один-до-багатьох»;
- `danceuser` – танцівник, який підтримав цього тренера з танців, визначений ідентифікатором з таблиці `Danceuser` через зв'язок «один-до-багатьох».

Таблиця `Danceteacherbyteachersupport` визначає підтримування тренерів з танців іншими тренерами з танців з такими полями:

- `id` – ідентифікатор підтримування тренера з танців іншим тренером з танців;
- `danceteacher1` – тренер з танців, якого підтримав інший тренер з танців, визначений ідентифікатором з таблиці `Danceteacher` через зв'язок «один-до-багатьох»;
- `danceteacher2` – тренер з танців, який підтримав іншого тренера з танців, визначений ідентифікатором з таблиці `Danceteacher` через зв'язок «один-до-багатьох».

Таблиця `Danceuserstylematerial` визначає матеріали за танцювальним досвідом танцівників з такими полями:

- `id` – ідентифікатор матеріалу за танцювальним досвідом танцівника;
- `danceuserstyle` – стиль танців танцівника, відтворений в цьому матеріалі за танцювальним досвідом танцівника, визначений ідентифікатором з таблиці `Danceuserstyle` через зв'язок «один-до-багатьох»;
- `dancematerialurl` – вебадреса матеріалу за танцювальним досвідом танцівника;
- `dancematerialname` – назва матеріалу за танцювальним досвідом танцівника;
- `dancematerialdescription` – опис матеріалу за танцювальним досвідом танцівника.

Таблиця `Dancepairteacher` визначає закріплених за танцювальними парами тренерів з танців з такими полями:

- `id` – ідентифікатор закріпленого за танцювальною парою тренера з танців;
- `dancepair` – танцювальна пара, за якою закріплено тренера з танців, визначена ідентифікатором з таблиці `Dancepair` через зв'язок «один-до-багатьох»;
- `danceteacher` – тренер з танців, який закріплений за цією танцювальною парою, визначений ідентифікатором з таблиці `Danceteacher` через зв'язок «один-до-багатьох»;
- `dancepairtstatus` – стан заявки на закріплення тренера з танців за парою для індивідуальних занять танцями;
- `dancepairteachdatecreation` – дата створення заявки на закріплення тренера з танців за парою для індивідуальних занять танцями;
- `dancepairteachdateapplication` – дата остаточного опрацювання заявки на закріплення тренера з танців за парою для індивідуальних занять танцями.

Таблиця `Danceuserteacher` визначає закріплених за танцівниками тренерів з танців з такими полями:

- id – ідентифікатор закріпленого за танцівником тренера з танців;
- danceuser – танцівник, за яким закріплено цього тренера з танців, визначений ідентифікатором з таблиці Danceuser через зв'язок «один-до-багатьох»;
- danceteacher – тренер з танців, закріплений за цим танцівником, визначений ідентифікатором з таблиці Danceteacher через зв'язок «один-до-багатьох»;
- danceairtstatus – стан заявки на закріплення тренера з танців за танцівником;
- danceairteachdatecreation – дата створення заявки на закріплення тренера з танців за танцівником;
- danceairteachdateapplication – дата остаточного опрацювання заявки на закріплення тренера з танців за танцівником.

Таблиця Danceairplace визначає локації, закріплені за парами для індивідуальних занять танцями, з такими полями:

- id – ідентифікатор закріплення локації за парою для індивідуальних занять танцями;
- danceair – пара для індивідуальних занять танцями, за якою закріплена ця локація, визначена ідентифікатором з таблиці Danceair через зв'язок «один-до-багатьох»;
- danceplace – локація, яка закріплена за цією парою для індивідуальних занять танцями, визначена ідентифікатором з таблиці Danceplace через зв'язок «один-до-багатьох»;
- dancemeetingtype – тип частоти застосування цієї локації для індивідуальних занять танцями цією парою.

Таблиця Danceuserplace визначає локації, закріплені за танцівниками, з такими полями:

- id – ідентифікатор закріплення локації за танцівником для індивідуальних занять танцями;

– `danceuser` – танцівник, за яким для індивідуальних занять танцями закріплено цю локацію, визначений ідентифікатором з таблиці `Danceuser` через зв'язок «один-до-багатьох»;

– `danceplace` – локація, яка закріплена за цим танцівником, визначена ідентифікатором з таблиці `Danceplace` через зв'язок «один-до-багатьох»;

– `dancemeetingtype` – тип частоти застосування цієї локації для індивідуальних занять танцями цим танцівником.

Таблиця `Dancepairmeeting` визначає заплановану зустріч для занять танцями танцювальної пари з такими полями:

– `id` – ідентифікатор запланованої зустрічі для занять танцями танцювальної пари;

– `dancepair` – танцювальна пара, для якої запланована ця зустріч, визначена ідентифікатором з таблиці `Dancepair` через зв'язок «один-до-багатьох»;

– `dancemeettype` – тип запланованої зустрічі для занять танцями танцювальної пари;

– `dancemeetdate` – дата, на яку запланована зустріч для занять танцями танцювальної пари;

– `dancemeetduration` – тривалість запланованої зустрічі для занять танцями танцювальної пари;

– `dancemeetstatus` – стан зустрічі для занять танцями танцювальної пари, яка початково є заявкою однієї зі сторін;

– `dancemeetindividualness` – визначення індивідуальної зустрічі або зустрічі, де будуть присутні і інші пари або танцівники, для занять танцями танцювальної пари;

– `danceplace` – локація, де відбувається зустріч для занять танцями танцювальної пари, визначена ідентифікатором з таблиці `Danceplace` через зв'язок «один-до-багатьох»;

– `dancemeetname` – назва зустрічі для занять танцями танцювальної пари (рис. 2.4).

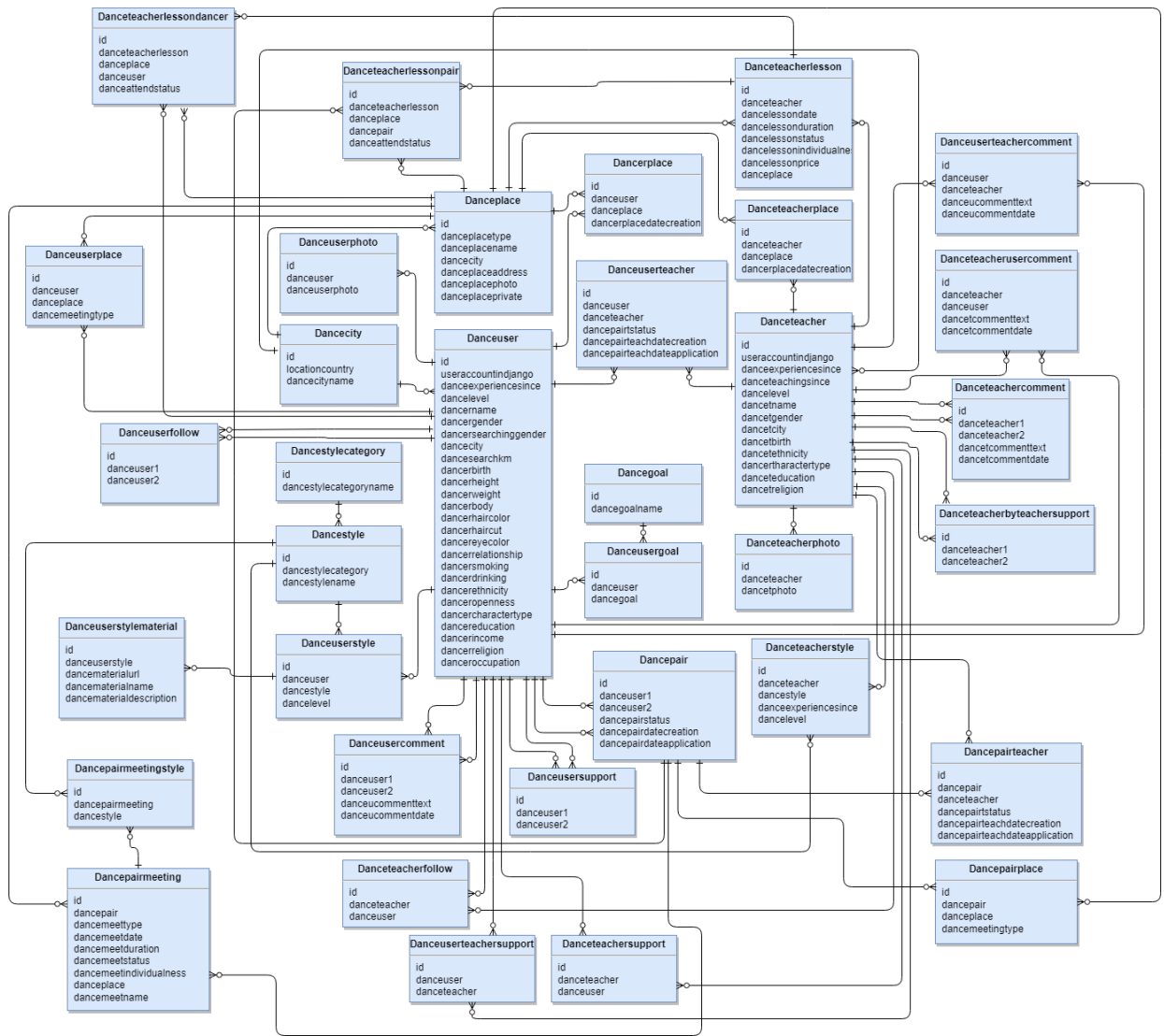


Рисунок 2.4 – Схема бази даних ПЗ організації індивідуальних занять танцями

Таблиця Dancepairmeetingstyle визначає стилі танців, які будуть використовуватися під час індивідуального заняття танцями пари, з такими полями:

– id – ідентифікатор стилю танців, який буде використовуватися під час індивідуального заняття танцями пари;

– dancepairmeeting – запланована зустріч для занять танцями танцювальної пари, на якій буде використовуватись цей стиль танців, визначена ідентифікатором з таблиці Dancepairmeeting через зв'язок «один-до-багатьох»;

– dancestyle – стиль танців, який буде використовуватися під час індивідуального заняття танцями пари, визначений ідентифікатором з таблиці Dancestyle через зв'язок «один-до-багатьох».

2.4 Висновки за розділом 2

Моделювання сценаріїв роботи з ПЗ організації індивідуальних занять танцями включало моделювання сценаріїв для танцівника, тренера з танців та неавторизованого користувача. Вибір засобів розробки виконувався між PHP і Python як мовами програмування, Laravel і Django як фреймворками з вибором Python і Django в підсумку. Структура бази даних ПЗ організації індивідуальних занять танцями враховує всю широту функціональних можливостей, які необхідно реалізувати в програмі.

3 РОЗРОБКА ПРОГРАМИ

3.1 Визначення порядку роботи з програмою

Заявки на створення пари для індивідуальних занять танцями, інші варіанти заявок передбачають певні стани, проте саме заявка на відвідування індивідуальних занять з танців тренера танцювальною парою передбачає найширшу сукупність станів (рис. 3.1), оскільки є взаємодією обох учасників танцювальної пари та тренера з танців.

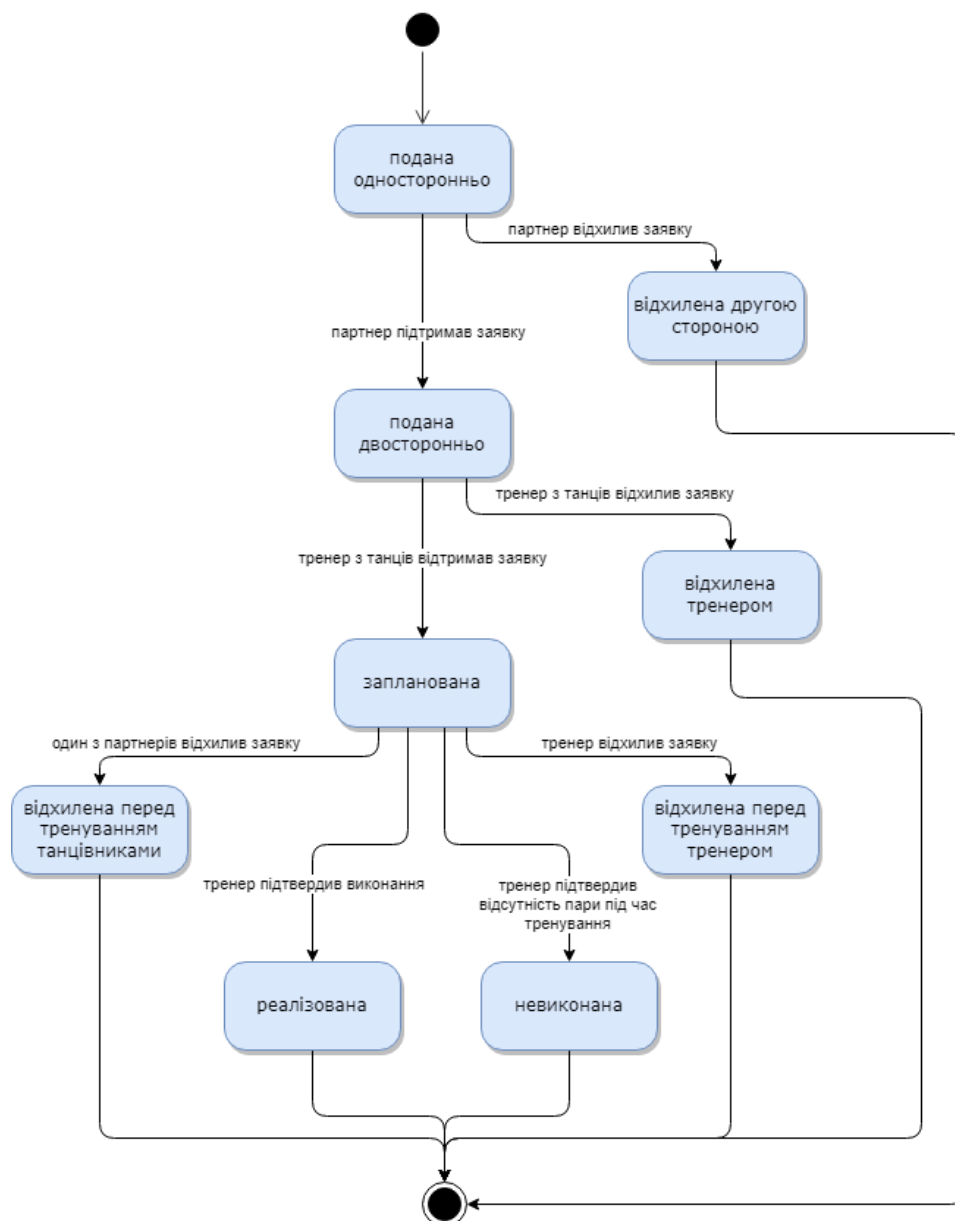


Рисунок 3.1 – Діаграма станів заявки на відвідування індивідуальних занять з танців тренера танцювальною парою

3.2 Реалізація основних класів програми організації індивідуальних занять танцями

У складі даної реалізації ПЗ організації індивідуальних занять танцями як Django-застосунку класи реалізують моделі застосунку.

Клас `Dancecity` визначає модель міста, де можуть знаходитись танцівники.

Поле `locationcountry` у складі класу `Dancecity` визначає країну, де це місто є, на основі типу `CharField`.

Поле `dancecityname` у складі класу `Dancecity` визначає назву міста на основі типу `CharField`.

Клас `Danceuser` визначає модель танцівників з обліковими записами користувачів.

Поле `useraccountindjango` у складі класу `Danceuser` визначає обліковий запис у Django за тим користувачем, який є цим танцівником, на основі типу `ForeignKey` з моделлю `User`.

Поле `danceexperiencesince` у складі класу `Danceuser` визначає рік, з якого почав займатися танцями цей танцівник, на основі типу `PositiveSmallIntegerField`.

Поле `dancelevel` у складі класу `Danceuser` визначає рівень танцювальних вмінь, на який себе оцінює цей танцівник, на основі типу `CharField`.

Поле `dancername` у складі класу `Danceuser` визначає ім'я танцівника у програмі на основі типу `CharField`.

Поле `dancergender` у складі класу `Danceuser` визначає стать цього танцівника на основі типу `CharField`.

Поле `dancersearchinggender` у складі класу `Danceuser` визначає стать напарника для танців, якого шукає цей танцівник, на основі типу `CharField`.

Поле `dancecity` у складі класу `Danceuser` визначає місто, в якому проживає і безпосередньо шукає партнера для танців цей танцівник, на основі типу `ForeignKey` з моделлю `Dancecity`.

Поле `dancesearchkm` у складі класу `Danceuser` визначає відстань у кілометрах, на якій готовий шукати партнера для танців цей танцівник, на основі типу `DecimalField`.

Поле `dancerbirth` у складі класу `Danceuser` визначає дату народження цього танцівника на основі типу `DateField`.

Поле `dancerheight` у складі класу `Danceuser` визначає зріст цього танцівника на основі типу `DecimalField`.

Поле `dancerweight` у складі класу `Danceuser` визначає вагу цього танцівника на основі типу `DecimalField`.

Поле `dancerbody` у складі класу `Danceuser` визначає структуру тіла цього танцівника на основі типу `CharField`.

Поле `dancerhaircolor` у складі класу `Danceuser` визначає колір волосся цього танцівника на основі типу `CharField`.

Поле `dancerhaircut` у складі класу `Danceuser` визначає зачіску цього танцівника на основі типу `CharField`.

Поле `dancereyecolor` у складі класу `Danceuser` визначає колір очей цього танцівника на основі типу `CharField`.

Поле `dancerrelationship` у складі класу `Danceuser` визначає наявність відносин у цього танцівника на основі типу `BooleanField`.

Поле `dancersmoking` у складі класу `Danceuser` визначає звичку палити у цього танцівника на основі типу `BooleanField`.

Поле `dancerdrinking` у складі класу `Danceuser` визначає звичку вживати алкоголь у цього танцівника на основі типу `BooleanField`.

Поле `dancerethnicity` у складі класу `Danceuser` визначає етнічне походження цього танцівника на основі типу `CharField`.

Поле `danceropenness` у складі класу `Danceuser` визначає відкритість до стосунків цього танцівника з партнерами по танцям на основі типу `BooleanField`.

Поле `dancercharacteretype` у складі класу `Danceuser` визначає тип характеру цього танцівника на основі типу `CharField`.

Поле `dancereducation` у складі класу `Danceuser` визначає рівень освіти цього танцівника на основі типу `CharField`.

Поле `dancerincome` у складі класу `Danceuser` визначає загальний рівень доходу цього танцівника на основі типу `CharField`.

Поле `dancerreligion` у складі класу `Danceuser` визначає релігію цього танцівника на основі типу `CharField`.

Поле `danceroccupation` у складі класу `Danceuser` визначає професію в житті цього танцівника на основі типу `CharField`.

Поле `danceusergoals` визначає цілі навчання танцям або танцювання танцівника на основі типу `ManyToManyField` з моделлю `Dancegoal`.

Поле `danceuserfollows` визначає слідкування за танцівниками цього танцівника на основі типу `ManyToManyField` з моделлю `Danceuser`.

Поле `danceteacherfollows` визначає слідкування за тренерами з танців танцівника на основі типу `ManyToManyField` з моделлю `Danceteacher`.

Поле `danceusersupports` визначає підтримування танцівників цим танцівником на основі типу `ManyToManyField` з моделлю `Danceuser`.

Поле `danceteachersupports` визначає підтримування тренерів з танців цим танцівником на основі типу `ManyToManyField` з моделлю `Danceteacher`.

Клас `Danceuserphoto` визначає модель фотографії танцівника.

Поле `danceuser` у складі класу `Danceuserphoto` визначає танцівника, який є на фотографії, на основі типу `ForeignKey` з моделлю `Danceuser`.

Поле `danceuserphoto` у складі класу `Danceuserphoto` визначає фотографію, де є цей танцівник, на основі типу `ImageField`.

Клас `Dancestylecategory` визначає модель категорії стилів танців.

Поле `dancestylecategoryname` у складі класу `Dancestylecategory` визначає назву категорії стилю танців на основі типу `CharField`.

Клас `Dancestyle` визначає модель стилів танців.

Поле `dancestylecategory` у складі класу `Dancestyle` визначає категорію, до якої належить цей стиль танцю, на основі типу `ForeignKey` з моделлю `Dancestylecategory`.

Поле `dancestylename` у складі класу `Dancestyle` визначає назву стилю танців на основі типу `CharField`.

Клас `Danceuserstyle` визначає модель стилів танців, які вмiє танцювати або яким навчається танцівник.

Поле `danceuser` у складі класу `Danceuserstyle` визначає танцівника, який вмiє танцювати або навчається танцювати цей стиль танців, на основі типу `ForeignKey` з моделлю `Danceuser`.

Поле `dancestyle` у складі класу `Danceuserstyle` визначає стиль танцю, який вмiє танцювати танцівник або який танцювати він навчається, на основі типу `ForeignKey` з моделлю `Dancestyle`.

Поле `dancelevel` у складі класу `Danceuserstyle` визначає рівень танцювальних вмiнь, якими оцінює цей танцівник свої вмiння стосовно цього стилю танців, на основі типу `CharField`.

Клас `Dancegoal` визначає модель цілей навчання танцям або танцювання.

Поле `dancegoalname` у складі класу `Dancegoal` визначає назву цілі навчання танцям або танцювання на основі типу `CharField`.

Клас `Dancepair` визначає модель танцювальних пар та запитів щодо формування таких пар для індивідуальних занять танцями.

Поле `danceuser1` у складі класу `Dancepair` визначає танцівника, який створив запит щодо формування пари для індивідуальних занять танцями, на основі типу `ForeignKey` з моделлю `Danceuser`.

Поле `danceuser2` у складі класу `Dancepair` визначає танцівника, до якого було адресовано запит щодо формування пари для індивідуальних занять танцями, на основі типу `ForeignKey` з моделлю `Danceuser`.

Поле `dancepairstatus` у складі класу `Dancepair` визначає стан запиту щодо формування такої пари для індивідуальних занять танцями на основі типу `CharField`.

Поле `dancepairdatecreation` у складі класу `Dancepair` визначає час подання запиту щодо формування пари для індивідуальних занять танцями на основі типу `DateTimeField`.

Поле `dancepairdateapplication` у складі класу `Dancepair` визначає час відповіді на запит щодо формування пари для індивідуальних занять танцями на основі типу `DateTimeField`.

Клас `Danceplace` визначає модель локації для навчання танцям.

Поле `danceplacetype` у складі класу `Danceplace` визначає тип локації для навчання танцям на основі типу `CharField`.

Поле `danceplacename` у складі класу `Danceplace` визначає назву локації для навчання танцям на основі типу `CharField`.

Поле `danceplaceaddress` у складі класу `Danceplace` визначає адресу локації для навчання танцям на основі типу `CharField`.

Поле `dancecity` у складі класу `Danceplace` визначає місто, в якому ця локація для навчання танцям є, на основі типу `ForeignKey` з моделлю `Dancecity`.

Поле `danceplacephoto` у складі класу `Danceplace` визначає фотографію локації для навчання танцям на основі типу `ImageField`.

Поле `danceplaceprivate` у складі класу `Danceplace` встановлює визначення того, чи є ця локація для навчання танцям приватною, чи публічною, на основі типу `BooleanField`.

Клас `Danceplace` визначає модель локацій, які доступні для занять танцями у танцівника.

Поле `danceuser` у складі класу `Danceplace` визначає танцівника, якому доступна для занять танцями ця локація, на основі типу `ForeignKey` з моделлю `Danceuser`.

Поле `danceplace` у складі класу `Danceplace` визначає локацію для занять танцями, доступна цьому танцівнику, на основі типу `ForeignKey` з моделлю `Danceplace`.

Поле `dancerplacedatecreation` у складі класу `Dancerplace` визначає дату визначення цієї локації як доступної для занять танцями у цього танцівника на основі типу `DateField`.

Клас `Danceteacher` визначає модель тренерів з танців з обліковими записами користувачів.

Поле `useraccountindjango` у складі класу `Danceteacher` визначає обліковий запис у Django за тим користувачем, який є цим тренером з танців, на основі типу `ForeignKey` з моделлю `User`.

Поле `danceexperiencesince` у складі класу `Danceteacher` визначає рік, з якого почав займатися танцями цей тренер з танців, на основі типу `PositiveSmallIntegerField`.

Поле `danceteachingsince` у складі класу `Danceteacher` визначає рік, з якого почав працювати тренером цей тренер з танців, на основі типу `PositiveSmallIntegerField`.

Поле `dancelevel` у складі класу `Danceteacher` визначає рівень танцювальних вмінь, на який себе оцінює цей тренер з танців, на основі типу `CharField`.

Поле `dancetname` у складі класу `Danceteacher` визначає ім'я цього тренера з танців на основі типу `CharField`.

Поле `dancetgender` у складі класу `Danceteacher` визначає стать цього тренера з танців на основі типу `BooleanField`.

Поле `dancetcity` у складі класу `Danceteacher` визначає місто, в якому працює тренером з танців цей користувач, на основі типу `ForeignKey` з моделлю `Dancecity`.

Поле `dancetbirth` у складі класу `Danceteacher` визначає дату народження цього тренера з танців на основі типу `DateField`.

Поле `dancetethnicity` у складі класу `Danceteacher` визначає етнічне походження цього тренера з танців на основі типу `CharField`.

Поле `dancertharactertype` у складі класу `Danceteacher` визначає тип характеру цього тренера з танців на основі типу `CharField`.

Поле `danceteducation` у складі класу `Danceteacher` визначає рівень освіти цього тренера з танців на основі типу `CharField`.

Поле `dancetreligion` у складі класу `Danceteacher` визначає релігію цього тренера з танців на основі типу `CharField`.

Поле `danceusersupports` визначає підтримування танцівників цим тренером з танців на основі типу `ManyToManyField` з моделлю `Danceuser`.

Поле `danceteachersupports` визначає підтримування тренерів з танців цим тренером з танців на основі типу `ManyToManyField` з моделлю `Danceteacher`.

Клас `Danceteacherphoto` визначає модель фотографії тренерів з танців.

Поле `danceteacher` у складі класу `Danceteacherphoto` визначає тренера з танців, який є на фотографії, на основі типу `ForeignKey` з моделлю `Danceteacher`.

Поле `dancetphoto` у складі класу `Danceteacherphoto` визначає фотографію, де є цей тренер з танців, на основі типу `ImageField`.

Клас `Danceteacherstyle` визначає модель стилів танців, яким може навчати тренер з танців.

Поле `danceteacher` у складі класу `Danceteacherstyle` визначає тренера з танців, який може навчати цьому стилю з танців, на основі типу `ForeignKey` з моделлю `Danceteacher`.

Поле `dancestyle` у складі класу `Danceteacherstyle` визначає стиль танців, якому може навчати цей тренер з танців, на основі типу `ForeignKey` з моделлю `Dancestyle`.

Поле `danceexperiencesince` у складі класу `Danceteacherstyle` визначає рік, з якого повноцінно вміє танцювати цей танець цей тренер з танців, на основі типу `PositiveSmallIntegerField`.

Поле `dancelevel` у складі класу `Danceteacherstyle` визначає танцювальний рівень цього тренера з танців за цим стилем танців за його власним враженням на основі типу `CharField`.

Клас `Danceteacherplace` визначає модель локацій, де можуть навчати танцям тренери з танців.

Поле `danceteacher` у складі класу `Danceteacherplace` визначає тренера з танців, якому доступна для навчання танцям ця локація, на основі типу `ForeignKey` з моделлю `Danceteacher`.

Поле `danceplace` у складі класу `Danceteacherplace` визначає локацію, доступну для навчання танцям цьому тренеру з танців, на основі типу `ForeignKey` з моделлю `Danceplace`.

Поле `danceplacedatecreation` у складі класу `Danceteacherplace` визначає дату, з якої ця локація доступна для навчання танцям цьому тренеру з танців, на основі типу `DateTimeField`.

Клас `Danceteacherlesson` визначає модель індивідуальних занять танцями у тренера з танців.

Поле `danceteacher` у складі класу `Danceteacherlesson` визначає тренера з танців, у якого доступне це індивідуальне заняття танцями, на основі типу `ForeignKey` з моделлю `Danceteacher`.

Поле `dancelessondate` у складі класу `Danceteacherlesson` визначає дату, коли це індивідуальне заняття танцями має відбуватися, на основі типу `DateField`.

Поле `dancelessonduration` у складі класу `Danceteacherlesson` визначає тривалість цього індивідуального заняття танцями на основі типу `TimeField`.

Поле `dancelessonstatus` у складі класу `Danceteacherlesson` визначає стан цього індивідуального заняття танцями на основі типу `CharField`.

Поле `dancelessonindividualness` у складі класу `Danceteacherlesson` встановлює визначення того, чи це індивідуальне заняття танцями, чи воно доступне для різних учасників, на основі типу `BooleanField`.

Поле `dancelessonprice` у складі класу `Danceteacherlesson` визначає вартість відвідування цього індивідуального заняття танцями на основі типу `DecimalField`.

Поле `danceplace` у складі класу `Danceteacherlesson` визначає локацію, де має відбуватися це індивідуальне заняття танцями, на основі типу `ForeignKey` з моделлю `Danceplace`.

Клас `Danceteachercomment` визначає модель коментарів тренерів з танців про інших тренерів з танців.

Поле `danceteacher1` у складі класу `Danceteachercomment` визначає тренера з танців, про якого написано коментар іншим тренером з танців, на основі типу `ForeignKey` з моделлю `Danceteacher`.

Поле `danceteacher2` у складі класу `Danceteachercomment` визначає тренера з танців, який написав коментар про іншого тренера з танців, на основі типу `ForeignKey` з моделлю `Danceteacher`.

Поле `dancetcommenttext` у складі класу `Danceteachercomment` визначає текст коментаря тренера з танців про іншого тренера з танців на основі типу `CharField`.

Поле `dancetcommentdate` у складі класу `Danceteachercomment` визначає час створення коментаря тренера з танців про іншого тренера з танців на основі типу `DateTimeField`.

Клас `Danceteacherusercomment` визначає модель коментарів танцівників про тренерів з танців.

Поле `danceteacher` у складі класу `Danceteacherusercomment` визначає тренера з танців, про якого написано коментар танцівником, на основі типу `ForeignKey` з моделлю `Danceteacher`.

Поле `danceuser` у складі класу `Danceteacherusercomment` визначає танцівника, який написав коментар про тренера з танців, на основі типу `ForeignKey` з моделлю `Danceuser`.

Поле `dancetcommenttext` у складі класу `Danceteacherusercomment` визначає текст коментаря танцівника про тренера з танців на основі типу `CharField`.

Поле `dancetcommentdate` у складі класу `Danceteacherusercomment` визначає час створення коментаря танцівника про тренера з танців на основі типу `DateTimeField`.

Клас `Danceusercomment` визначає модель коментарів танцівників про інших танцівників.

Поле `danceuser1` у складі класу `Danceusercomment` визначає танцівника, про якого інший танцівник створив коментар, на основі типу `ForeignKey` з моделлю `Danceuser`.

Поле `danceuser2` у складі класу `Danceusercomment` визначає танцівника, який створив коментар про іншого танцівника, на основі типу `ForeignKey` з моделлю `Danceuser`.

Поле `danceucommenttext` у складі класу `Danceusercomment` визначає текст коментаря танцівника про іншого танцівника на основі типу `CharField`.

Поле `danceucommentdate` у складі класу `Danceusercomment` визначає час створення коментаря танцівника про іншого танцівника на основі типу `DateTimeField`.

Клас `Danceuserteachercomment` визначає модель коментарів тренерів з танців про танцівників.

Поле `danceuser` у складі класу `Danceuserteachercomment` визначає танцівника, про якого тренер з танців створив коментар, на основі типу `ForeignKey` з моделлю `Danceuser`.

Поле `danceteacher` у складі класу `Danceuserteachercomment` визначає тренера з танців, який створив коментар про танцівника, на основі типу `ForeignKey` з моделлю `Danceteacher`.

Поле `danceucommenttext` у складі класу `Danceuserteachercomment` визначає текст коментаря тренера з танців про танцівника на основі типу `CharField`.

Поле `danceucommentdate` у складі класу `Danceuserteachercomment` визначає час створення коментаря тренера з танців про танцівника на основі типу `DateTimeField`.

Клас `Danceteacherlessonpair` визначає модель відвідування або заявки на відвідування індивідуальних занять з танців тренера танцювальною парою.

Поле `danceteacherlesson` у складі класу `Danceteacherlessonpair` визначає індивідуальне заняття з танців тренера, яке має відвідати танцювальна пара, на основі типу `ForeignKey` з моделлю `Danceteacherlesson`.

Поле `danceplace` у складі класу `Danceteacherlessonpair` визначає локацію, де має відбуватися це індивідуальне заняття з танців тренера, на основі типу `ForeignKey` з моделлю `Danceplace`.

Поле `dancepair` у складі класу `Danceteacherlessonpair` визначає танцювальну пару, яка взялась відвідати це індивідуальне заняття з танців тренера, на основі типу `ForeignKey` з моделлю `Dancepair`.

Поле `danceattendstatus` у складі класу `Danceteacherlessonpair` визначає стан заявки на відвідування індивідуального заняття з танців тренера танцювальною парою на основі типу `CharField`.

Клас `Danceteacherlessondancer` визначає модель відвідування або заявки на відвідування індивідуальних занять з танців тренера танцівником.

Поле `danceteacherlesson` у складі класу `Danceteacherlessondancer` визначає індивідуальне заняття з танців тренера, яке має відвідати танцівник, на основі типу `ForeignKey` з моделлю `Danceteacherlesson`.

Поле `danceplace` у складі класу `Danceteacherlessondancer` визначає локацію, де має відбуватися це індивідуальне заняття з танців тренера, на основі типу `ForeignKey` з моделлю `Danceplace`.

Поле `danceuser` у складі класу `Danceteacherlessondancer` визначає танцівника, який взявся відвідати це індивідуальне заняття з танців тренера, на основі типу `ForeignKey` з моделлю `Danceuser`.

Поле `danceattendstatus` у складі класу `Danceteacherlessondancer` визначає стан заявки на відвідування індивідуального заняття з танців тренера танцівником на основі типу `CharField`.

Клас `Danceuserstylematerial` визначає модель матеріалів за танцювальним досвідом танцівників.

Поле `danceuserstyle` у складі класу `Danceuserstylematerial` визначає стиль танців танцівника, відтворений в цьому матеріалі за танцювальним досвідом танцівника, на основі типу `ForeignKey` з моделлю `Danceuserstyle`.

Поле `dancematerialurl` у складі класу `Danceuserstylematerial` визначає веб адресу матеріалу за танцювальним досвідом танцівника на основі типу `URLField`.

Поле `dancematerialname` у складі класу `Danceuserstylematerial` визначає назву матеріалу за танцювальним досвідом танцівника на основі типу `CharField`.

Поле `dancematerialdescription` у складі класу `Danceuserstylematerial` визначає опис матеріалу за танцювальним досвідом танцівника на основі типу `CharField`.

Клас `Dancepairteacher` визначає модель закріплених за танцювальними парами тренерів з танців.

Поле `dancepair` у складі класу `Dancepairteacher` визначає танцювальну пару, за якою закріплено тренера з танців, на основі типу `ForeignKey` з моделлю `Dancepair`.

Поле `danceteacher` у складі класу `Dancepairteacher` визначає тренера з танців, який закріплений за цією танцювальною парою, на основі типу `ForeignKey` з моделлю `Danceteacher`.

Поле `dancepairtstatus` у складі класу `Dancepairteacher` визначає стан заявки на закріплення тренера з танців за парою для індивідуальних занять танцями на основі типу `CharField`.

Поле `dancepairteachdatecreation` у складі класу `Dancepairteacher` визначає дату створення заявки на закріплення тренера з танців за парою для індивідуальних занять танцями на основі типу `DateTimeField`.

Поле `dancepairteachdateapplication` у складі класу `Dancepairteacher` визначає дату остаточного опрацювання заявки на закріплення тренера з танців за парою для індивідуальних занять танцями на основі типу `DateTimeField`.

Клас `Danceuserteacher` визначає модель закріплених за танцівниками тренерів з танців.

Поле `danceuser` у складі класу `Danceuserteacher` визначає танцівника, за яким закріплено цього тренера з танців, на основі типу `ForeignKey` з моделлю `Danceuser`.

Поле `danceteacher` у складі класу `Danceuserteacher` визначає тренера з танців, закріплений за цим танцівником, на основі типу `ForeignKey` з моделлю `Danceteacher`.

Поле `dancepairtstatus` у складі класу `Danceuserteacher` визначає стан заявки на закріплення тренера з танців за танцівником на основі типу `CharField`.

Поле `dancepairteachdatecreation` у складі класу `Danceuserteacher` визначає дату створення заявки на закріплення тренера з танців за танцівником на основі типу `DateTimeField`.

Поле `dancepairteachdateapplication` у складі класу `Danceuserteacher` визначає дату остаточного опрацювання заявки на закріплення тренера з танців за танцівником на основі типу `DateTimeField`.

Клас `Danceairplace` визначає модель локацій, закріплених за парами для індивідуальних занять танцями.

Поле `danceair` у складі класу `Danceairplace` визначає пару для індивідуальних занять танцями, за якою закріплена ця локація, на основі типу `ForeignKey` з моделлю `Danceair`.

Поле `danceplace` у складі класу `Danceairplace` визначає локацію, яка закріплена за цією парою для індивідуальних занять танцями, на основі типу `ForeignKey` з моделлю `Danceplace`.

Поле `dancemeetingtype` у складі класу `Danceairplace` визначає тип частоти застосування цієї локації для індивідуальних занять танцями цією парою на основі типу `CharField`.

Клас `Danceuserplace` визначає модель локацій, закріплених за танцівниками.

Поле `danceuser` у складі класу `Danceuserplace` визначає танцівника, за яким для індивідуальних занять танцями закріплено цю локацію, на основі типу `ForeignKey` з моделлю `Danceuser`.

Поле `danceplace` у складі класу `Danceuserplace` визначає локацію, яка закріплена за цим танцівником, на основі типу `ForeignKey` з моделлю `Danceplace`.

Поле `dancemeetingtype` у складі класу `Danceuserplace` визначає тип частоти застосування цієї локації для індивідуальних занять танцями цим танцівником на основі типу `CharField`.

Клас `Danceairmeeting` визначає модель запланованої зустрічі для занять танцями танцювальної пари.

Поле `danceair` у складі класу `Danceairmeeting` визначає танцювальну пару, для якої запланована ця зустріч, на основі типу `ForeignKey` з моделлю `Danceair`.

Поле `dancemeettype` у складі класу `Danceairmeeting` визначає тип запланованої зустрічі для занять танцями танцювальної пари на основі типу `CharField`.

Поле `dancemeetdate` у складі класу `Danceairmeeting` визначає дату, на яку запланована зустріч для занять танцями танцювальної пари, на основі типу `DateField`.

Поле `dancemeetduration` у складі класу `Danceairmeeting` визначає тривалість запланованої зустрічі для занять танцями танцювальної пари на основі типу `TimeField`.

Поле `dancemeetstatus` у складі класу `Danceairmeeting` визначає стан зустрічі для занять танцями танцювальної пари, яка початково є заявкою однієї зі сторін, на основі типу `CharField`.

Поле `dancemeetindividualness` у складі класу `Danceairmeeting` встановлює визначення індивідуальної зустрічі або зустрічі, де будуть присутні і інші пари або танцівники, для занять танцями танцювальної пари на основі типу `BooleanField`.

Поле `danceplace` у складі класу `Dancepairmeeting` визначає локацію, де відбувається зустріч для занять танцями танцювальної пари, на основі типу `ForeignKey` з моделлю `Danceplace`.

Поле `dancemeetname` у складі класу `Dancepairmeeting` визначає назву зустрічі для занять танцями танцювальної пари на основі типу `CharField`.

Поле `dancestyles` визначає стилі танців, які будуть використовуватися під час цього індивідуального заняття танцями пари, на основі типу `ManyToManyField` з моделлю `Dancestyle`.

3.3 Реалізація основних програмних функцій для організації індивідуальних занять танцями

Програмні функції у складі створеного Django-застосунку пов'язані з реалізацією представлень у файлі `views.py`. Відповідно вони за логікою Django-застосунку головним чином виділяють ті дані, які необхідні для інтерфейсів користувачів та передають їх, наповнюючи шаблони вебсторінок, таким чином саме ці складові пов'язані з безпосередньою реалізацією функцій програми. Шаблони створені на основі вільного шаблону [9]. Основна частина таких функцій описана нижче. Усі заплановані функціональні можливості програми були повністю реалізовані.

Відтворення доступних варіантів локацій для занять танцями у танцівника пов'язано з реалізацією у вигляді функції `view_danceuser_locations`, що має параметр `danceuser`, який встановлює логін самого танцівника, проте якщо цей логін не вказаний, то передбачається робота зі своїми власними даними, тому визначається те, як авторизувався користувач. Якщо він авторизувався некоректно, тобто він є тренером з танців, але намагається отримати доступ до цієї функції, то відбувається обробка такої ситуації, а користувачу виводиться повідомлення про ситуацію, що виникла. Обробка помилок реалізується у всіх випадках у подальшому. Якщо виникає помилка,

то виводиться повідомлення на відповідній сторінці. Шаблон використовуваної вебсторінки визначений як `danceplaces.html`.

Відтворення доступних варіантів локацій для занять танцями у тренера з танців пов'язано з реалізацією у вигляді функції `view_danceteacher_locations`, що має параметр `danceteacher`, який встановлює логін самого тренера з танців, проте якщо цей логін не вказаний, то передбачається робота зі своїми власними даними, тому визначається те, як авторизувався користувач. Шаблон використовуваної вебсторінки визначений як `danceteachplaces.html`.

Відтворення анкети танцівника пов'язано з реалізацією у вигляді функції `view_danceuser`, що має параметр `danceuser`, який встановлює логін самого танцівника, проте якщо цей логін не вказаний, то передбачається робота зі своїми власними даними, тому визначається те, як авторизувався користувач. Шаблон використовуваної вебсторінки визначений як `dancer.html`.

Відтворення коментарів за танцівником пов'язано з реалізацією у вигляді функції `view_danceuser_comments`, що має параметри `from`, що встановлює необхідність відтворення коментарів, які створив цей користувач (або навпаки відповідно значення `True/False`), `danceuser`, що встановлює логін танцівника. Шаблон використовуваної вебсторінки визначений як `dancercomms.html`.

Відтворення коментарів за тренером з танців пов'язано з реалізацією у вигляді функції `view_danceteacher_comments`, що має параметри `from`, що встановлює необхідність відтворення коментарів, які створив цей користувач (або навпаки відповідно значення `True/False`), `danceteacher`, що встановлює логін тренера з танців. Шаблон використовуваної вебсторінки визначений як `danceteachcomms.html`.

Відтворення сторінки тренера для занять танцями пов'язано з реалізацією у вигляді функції `view_danceteacher`, що має параметр `danceteacher`, який встановлює логін самого тренера з танців, проте якщо цей логін не вказаний, то передбачається робота зі своїми власними даними, тому

визначається те, як авторизувався користувач. Шаблон використовуваної вебсторінки визначений як `danceteacher.html`.

Відтворення графіку індивідуальних занять танцями пари пов'язано з реалізацією у вигляді функції `view_dancepair_schedule`, що має параметри `dancepair`, який встановлює сам ідентифікатор танцювальної пари, `meeting`, що встановлює варіант відображення індивідуальних занять (0 – для запланованих занять з танців самою парою, 1 – для занять, що були ініційовані тренером, 2 – для всіх занять з танців), `ordering`, що встановлює порядок сортування, `startdate` і `enddate`, що встановлюють часовий проміжок занять з танців. Шаблон використовуваної вебсторінки визначений як `pairschedule.html`.

Відтворення даних підтриманих танцівників користувачем пов'язано з реалізацією у вигляді функції `view_danceuser_supports` з шаблоном вебсторінки `supports.html`.

Відтворення даних підтриманих тренерів користувачем пов'язано з реалізацією у вигляді функції `view_danceteacher_supports` з шаблоном вебсторінки `teachsupports.html`.

Відтворення даних танцівників, за якими слідкує користувач, пов'язано з реалізацією у вигляді функції `view_danceuser_follows` з шаблоном вебсторінки `follows.html`.

Відтворення даних тренерів, за якими слідкує користувач, пов'язано з реалізацією у вигляді функції `view_danceuser_teachers_follows` з шаблоном вебсторінки `teachfollows.html`.

Відтворення даних прокоментованих танцівників пов'язано з реалізацією у вигляді функції `view_danceusers_commented` з шаблоном вебсторінки `ondancercomms.html`.

Відтворення даних прокоментованих тренерів пов'язано з реалізацією у вигляді функції `view_danceteachers_commented` з шаблоном вебсторінки `teachcommented.html`.

Відтворення поданих заявок на створення пари для індивідуальних занять танцями пов'язано з реалізацією у вигляді функції `view_dancepair_applications`, що має параметр `to`, який встановлює, чи це мають бути адресовані користувачу заявки на створення пари для індивідуальних занять танцями, чи навпаки створені ним. Шаблон використовуваної вебсторінки визначений як `pairsappl.html`.

Пошук партнерів для занять танцями пов'язаний з реалізацією у вигляді функції `view_dancers_search`, яка критерії пошуку отримує на основі ключів GET-запиту, визначаючи необхідний мінімальний рік початку занять танцями `danceexperiencesince`, необхідний мінімальний рівень танцювальних навичок `dancelevel`, стать танцівника `dancergender`, бажаний мінімальний та максимальний вік танцівника `danceragemin` та `dancermaxage`, мінімальна та максимальна висота танцівника `dancerminheight` та `dancermaxheight`, мінімальна та максимальна вага танцівника `dancerminweight` та `dancermaxweight`, бажана статура танцівника `dancerbody`, бажаний колір волосся танцівника `dancerhaircolor`, бажана зачіска танцівника `dancerhaircut`, бажаний колір очей танцівника `dancereyecolor`, стан відносин танцівника `dancerrelationship`, наявність звички палити `dancersmoking`, вживати алкоголь `dancerdrinking`, етнічна приналежність `dancerethnicity`, відкритість до побудови відносин `danceropenness`, тип характеру танцівника `dancercharacterstype`, рівень освіти танцівника `dancereducation`, бажаний рівень доходу танцівника `dancerincome`, бажані релігійні погляди танцівника `dancerreligion`, бажана професія танцівника `danceroccupation`, стиль танців, який повинен танцювати танцівник обов'язково `dancestyle`. Стать танцівника, що шукається, може задаватися в профілі танцівника, тоді використовується відповідне значення. Місто пошуку, відстань від міста задаються в профілі танцівника, визначаючи відповідні параметри пошуку. Результати пошуку партнерів для занять танцями надаються в шаблоні вебсторінки `dancersearch.html`.

Закріплення тренера для індивідуальних занять танцями пари та скасування закріплення тренера для індивідуальних занять танцями пари

пов'язано з реалізацією у вигляді функції `view_pair_danceteacher_set`, що має параметри `dancerpair`, що встановлює ідентифікатор танцювальної пари, та `danceteacher`, що задає логін тренера з танців. Відбувається перевірка, що користувач авторизувався або як такий тренер, або як партнер з цієї танцювальної пари. Також перевіряється, чи є таке закріплення зараз. Якщо так, то реалізується скасування, якщо ні – закріплення.

Закріплення тренера для індивідуальних занять танцями танцівника та скасування закріплення тренера для індивідуальних занять танцями танцівника пов'язано з реалізацією у вигляді функції `view_dancer_danceteacher_set`, що має параметри `dancer`, що встановлює логін танцівника, та `danceteacher`, що задає логін тренера з танців. Відбувається перевірка, що користувач авторизувався або як такий танцівник, або як такий тренер з танців. Також перевіряється, чи є таке закріплення зараз. Якщо так, то реалізується скасування, якщо ні – закріплення.

Підтримування танцівника та скасування підтримування танцівника пов'язано з реалізацією у вигляді функції `view_dancer_support_set`, що має параметр `dancer`, що встановлює логін танцівника.

Підтримування тренера з танців та скасування підтримування тренера з танців пов'язано з реалізацією у вигляді функції `view_danceteacher_support_set`, що має параметр `danceteacher`, що встановлює логін тренера з танців.

Визначення слідкування за танцівником та скасування визначення слідкування за танцівником пов'язано з реалізацією у вигляді функції `view_dancer_follow_set`, що має параметр `dancer`, що встановлює логін танцівника.

Визначення слідкування за тренером з танців та скасування визначення слідкування за тренером з танців пов'язано з реалізацією у вигляді функції `view_danceteacher_follow_set`, що має параметр `danceteacher`, що встановлює логін тренера з танців.

Закріплення локації для індивідуальних занять танцями танцівника та скасування закріплення локації для індивідуальних занять танцями танцівника пов'язано з реалізацією у вигляді функції `view_dancer_place_set`, що має параметр `danceplace`, що встановлює ідентифікатор локації, а якщо він не встановлений, то використовується клас `DanceplaceForm`, який підтримує роботу з формою для внесення нової локації.

Закріплення локації для індивідуальних занять танцями пари та скасування закріплення локації для індивідуальних занять танцями пари пов'язано з реалізацією у вигляді функції `view_dancepair_place_set`, що має параметр `danceplace`, що встановлює ідентифікатор локації, а якщо він не встановлений, то використовується клас `DanceplaceForm`, який підтримує роботу з формою для внесення нової локації. Другий параметр `dancepair` задає ідентифікатор танцювальної пари. До контексту даних шаблону вебсторінки також передається перелік танцювальних пар, у яких бере участь танцівник, що авторизувався.

Визначення власного опису смаків та досвіду в танцях пов'язано з реалізацією у вигляді функції `view_danceuser_profile_set`, що працює на основі підтримки за допомогою класу форми `DanceuserForm`.

Розміщення доступного власного варіанту локації для занять танцями пов'язано з реалізацією у вигляді функції `view_danceplace_add`, що працює на основі підтримки за допомогою класу форми `DanceplaceForm`.

Відтворення графіку запланованих індивідуальних занять танцями тренера пов'язано з реалізацією у вигляді функції `view_danceteacher_schedule_lessons` з шаблоном вебсторінки `prteacherless.html`.

Відтворення заявок за індивідуальними заняттями танцями тренера пов'язано з реалізацією у вигляді функції `view_danceteacher_lesson_applications` з шаблоном вебсторінки `prteacherapplications.html`.

Визначення тренерських характеристик пов'язано з реалізацією у вигляді функції `view_danceteacher_profile_set`, що працює на основі підтримки за допомогою класу форми `DanceteacherForm`.

Відтворення доступних варіантів індивідуальних занять танцями у тренера пов'язано з реалізацією у вигляді функції `view_danceteacher_lessons` з шаблоном вебсторінки `teacherless.html`.

Подання заявки на створення пари для індивідуальних занять танцями пов'язано з реалізацією у вигляді функції `view_dance_pair_create`, що має параметр `danceuser`, який задає логін танцівника, з яким створюється пара.

Скасування пари для індивідуальних занять танцями пов'язано з реалізацією у вигляді функції `view_dance_pair_cancel`, що має параметр `danceuser`, який задає логін танцівника, з яким скасовується пара.

Відхилення пари для індивідуальних занять танцями пов'язано з реалізацією у вигляді функції `view_dance_pair_delete`, що має параметр `danceuser`, який задає логін танцівника, з яким відхиляється пара.

Розміщення матеріалів за танцювальним досвідом пов'язано з реалізацією у вигляді функції `view_danceuser_material_add`, що працює на основі підтримки за допомогою класу форми `DanceuserstylematerialForm`.

Редагування запису про матеріал за танцювальним досвідом пов'язано з реалізацією у вигляді функції `view_danceuser_material_edit`, що працює на основі підтримки за допомогою класу форми `DanceuserstylematerialForm`. Параметр `dancematerial` задає ідентифікатор матеріалу за танцювальним досвідом.

Вилучення запису про матеріал за танцювальним досвідом пов'язано з реалізацією у вигляді функції `view_danceuser_material_delete`, що має параметр `dancematerial`, який задає ідентифікатор матеріалу за танцювальним досвідом.

Таким чином реалізується коло CRUD-операцій.

Пошук локацій для занять танцями пов'язаний з реалізацією у вигляді функції `view_dance_places_search`, яка критерії пошуку отримує на основі ключів GET-запиту, визначаючи тип локації для занять танцями

dancerplacetype, назву локації чи її частину dancerplacename, місто локації для занять танцями dancescity. Результати пошуку локацій для занять танцями надаються в шаблоні вебсторінки dancerplacesearch.html.

Пошук тренерів для занять танцями пов'язаний з реалізацією у вигляді функції view_danceteachers_search, яка критерії пошуку отримує на основі ключів GET-запиту, визначаючи мінімальний рік початку занять танцями тренера danceexperiencesince, мінімальний рік початку роботи як тренера з танців danceteachingsince, танцювальний рівень dancelevel, ім'я тренера з танців dancetname, стать тренера з танців dancetgender, ідентифікатор міста тренера з танців dancetcity, мінімальний і максимальний вік тренера з танців dancetminage та dancetmaxage, етнічну приналежність тренера з танців dancetethnicity, тип характеру тренера з танців dancetheractertype, рівень освіти тренера з танців danceteducation, релігійні уподобання тренера з танців dancetreligion, стиль танців, якому повинен навчати тренер обов'язково dancestyle. Результати пошуку тренерів для занять танцями надаються в шаблоні вебсторінки danceteachersearch.html.

Відтворення створених пар для індивідуальних занять танцями пов'язано з реалізацією у вигляді функції view_danceuser_dancepairs. Шаблон використовуваної вебсторінки визначений як pairs.html.

Комплексний пошук партнера, локації та тренера для занять танцями пов'язано з реалізацією у вигляді функції view_complex_search з шаблоном вебсторінки complexsearch.html з об'єднанням параметрів пошуку з попередніх представлених функцій пошуку.

Подання заявки стосовно конкретного індивідуального заняття танцями у тренера пов'язано з реалізацією у вигляді функції view_danceteacher_lesson_apply, що має параметр danceteacherlesson, який встановлює ідентифікатор індивідуального заняття танцями у тренера, та dancepair, що задає ідентифікатор танцювальної пари, а якщо значення пропущено, то передбачається внесення даних за самим танцівником.

Керування заявками стосовно створення пар для індивідуальних занять танцями пов'язано з реалізацією у вигляді функції `view_dance_pair_result`, що має параметр `dancepair`, що задає ідентифікатор заявки на створення танцювальної пари, `result`, що задає результат оброблення цієї заявки.

Скасування власної заявки стосовно конкретного індивідуального заняття танцями у тренера пов'язано з реалізацією у вигляді функції `view_danceteacher_lesson_apply_cancel`, що має параметр `lessonappl`, що задає ідентифікатор заявки, `individual`, що задає це індивідуальна заявка танцівника чи танцювальної пари.

Керування поданою заявкою стосовно конкретного індивідуального заняття танцями цього тренера пов'язано з реалізацією у вигляді функції `view_danceteacher_lesson_apply_result`, що має параметр `lessonappl`, що задає ідентифікатор заявки, `individual`, що задає це індивідуальна заявка танцівника чи танцювальної пари, `result`, що задає результат оброблення цієї заявки.

3.4 Висновки за розділом 3

Оскільки танці хоч і можуть реалізовуватися в рамках індивідуальних зустрічей, проте такі зустрічі можуть охоплювати і обох учасників танцювальної пари, і тренера з танців, то реалізація заявок щодо таких індивідуальних зустрічей для занять танцями пов'язана зі взаємодією всіх сторін, а тому була розглянута окремо у вигляді відповідної сутності заявки на відвідування індивідуальних занять з танців тренера танцювальною парою та її станів на діаграмі станів.

Стосовно реалізації ПЗ організації індивідуальних занять танцями було створено всі складові відповідного Django-застосунку, а класи моделей, функції представлень були описані в розділі.

4 ЕКСПЛУАТАЦІЯ ТА ТЕСТУВАННЯ ПРОГРАМИ

4.1 Призначення та умови виконання програми організації індивідуальних занять танцями

ПЗ організації індивідуальних занять танцями призначене для забезпечення підтримки занять танцями на етапах пошуку партнерів для створення пари, підбору тренера для пари або окремого танцівника, підбору локацій для таких занять, комплексного вирішення таких проблем, планування самих індивідуальних занять танцями.

Експлуатація програми організації індивідуальних занять танцями повинна виконуватись за забезпечення умов, які включають програмну частину сервера. Вимог до клієнтської частини не висувається, адже наявність встановленої глобальної мережі, встановленого браузера вже реалізують ті вимоги, які мають бути забезпечені на клієнтському пристрої. Вимоги ж до програмної частини сервера виходять з тих рішень, які прийняті в роботі щодо вибору засобів розробки: інтерпретатор мови Python, засоби розробки Django-проєкту, система PostgreSQL. Також необхідна наявність адміністратора програми, який би контролював роботу користувачів, вносив дані до програми, які знаходяться в базі даних.

4.2 Логіка експлуатації програми організації індивідуальних занять танцями

Для роботи з ПЗ організації індивідуальних занять танцями спершу потрібно звернутися за адресою вебсервера, після чого відкриється можливість пошуку партнерів для занять танцями. Під час пошуку партнерів для занять танцями можна задати:

- необхідний мінімальний рік початку занять танцями;
- необхідний мінімальний рівень танцювальних навичок;
- стать танцівника

- бажаний мінімальний та максимальний вік танцівника;
- мінімальний та максимальний зріст танцівника;
- мінімальну та максимальну вагу танцівника;
- бажану статуру танцівника;
- бажаний колір волосся танцівника;
- бажану зачіску танцівника;
- бажаний колір очей танцівника;
- стан відносин танцівника;
- наявність звички палити;
- наявність звички вживати алкоголь;
- етнічну приналежність;
- відкритість до побудови відносин;
- тип характеру танцівника;
- рівень освіти танцівника;
- бажаний рівень доходу танцівника;
- бажані релігійні погляди танцівника;
- бажану професію танцівника;
- стиль танців, який повинен танцювати танцівник обов'язково.

У результатах пошуку партнерів для занять танцями (рис. 4.1) відображається за кожним танцівником його фотографія, ім'я, стилі, в яких він танцює, рік дебюту в танцях, кількість користувачів, які сумарно його підтримали, кількість коментарів, написаних про нього. Танцівника можна підтримати, можна створити заявку на пару для індивідуальних занять танцями. Проте для виконання таких дій фактично потрібно авторизуватися. Для цього присутній відповідній крайній справа пункт меню.

Також через меню користувач до авторизації може перейти до пошуку локацій для занять танцями та до пошуку тренерів з танців. Усі відповідні пункти меню представлені окремою сторінкою, з якої можна перейти до перегляду детальних результатів відповідно за танцівником, тренером з танців, локацією для занять танцями.

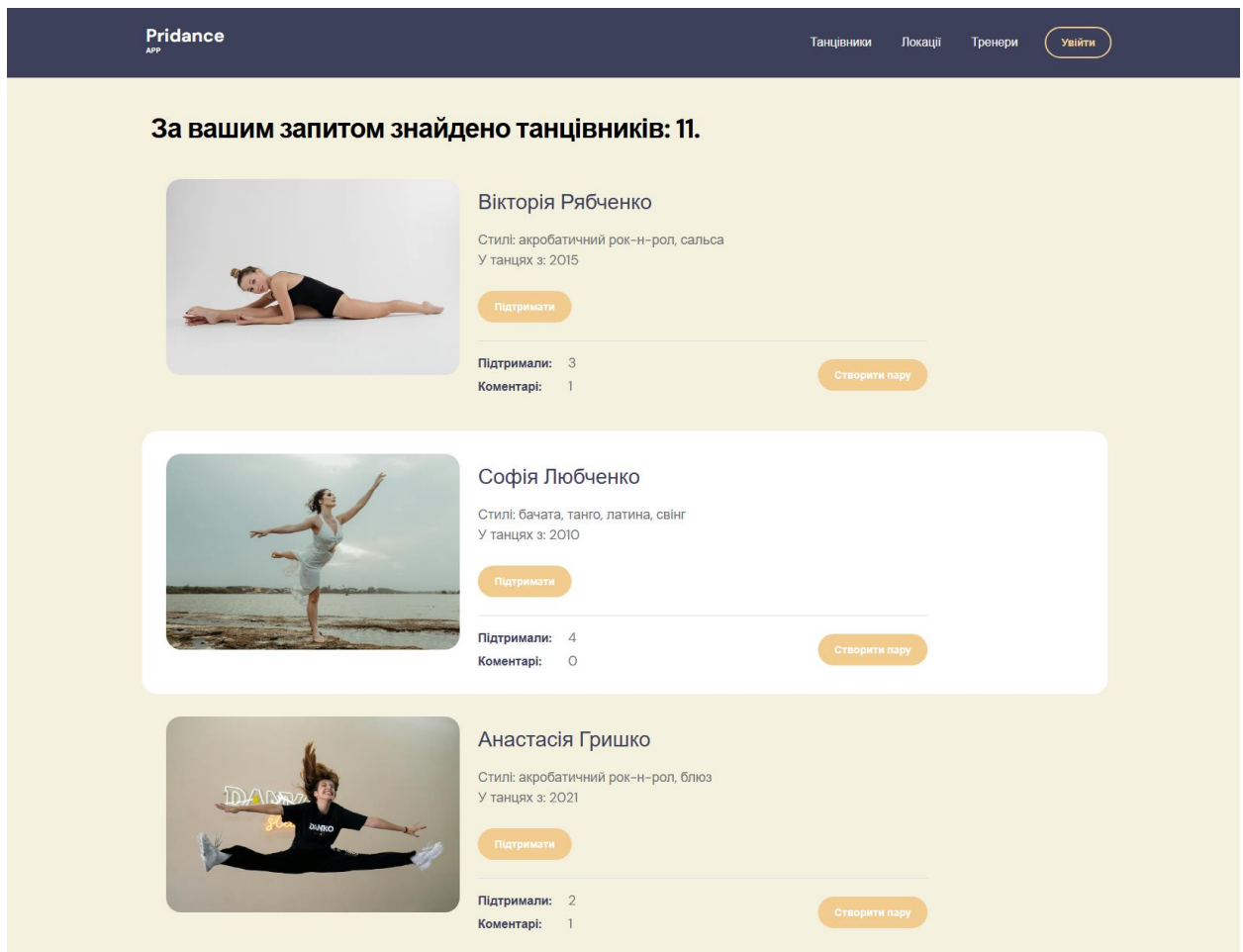


Рисунок 4.1 – Результати пошуку партнерів для занять танцями

Якщо ж користувач авторизувався як танцівник, то він має вже змінене меню, яке включає такі основні розділи:

- розділ «Пари» дозволяє з випадуючого переліку обирати можливості роботи з графіком індивідуальних занять танцями пари, планування нового заняття для танцювальної пар;

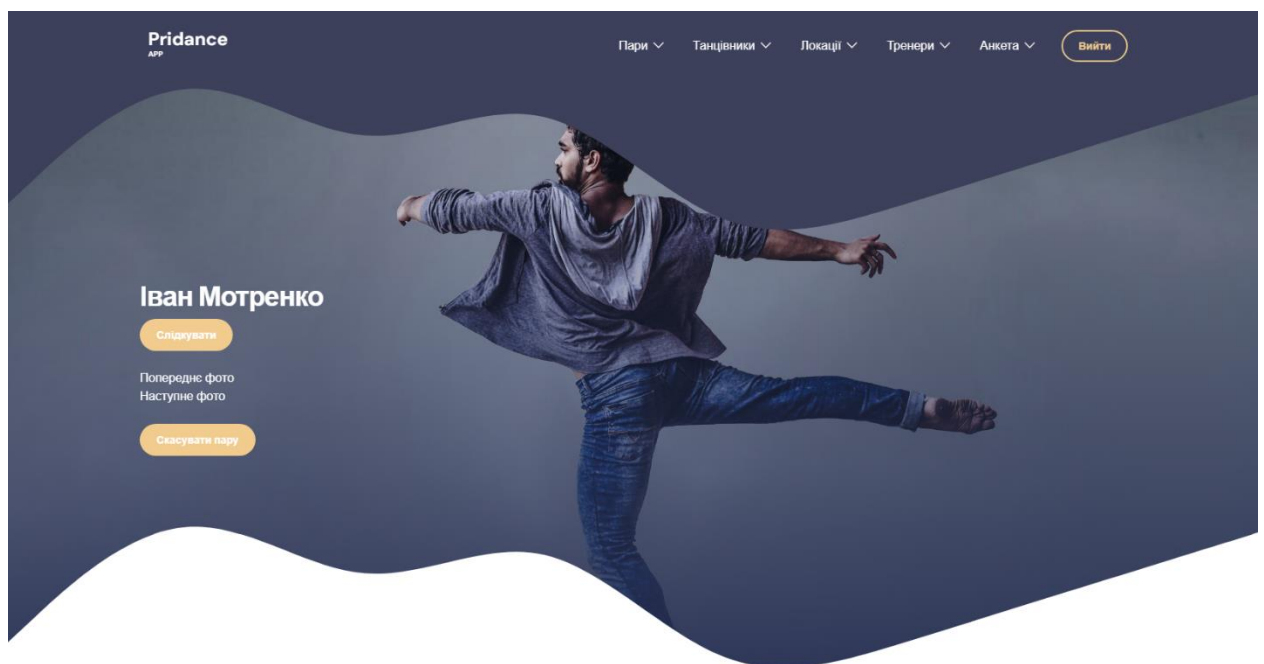
- розділ «Танцівники» дозволяє з випадуючого переліку обирати можливості пошуку, перегляду тих танцівників, за яким слідкує користувач, перегляду тих танцівників, яких він підтримав, перегляду тих танцівників, про яких він створив коментарі, виконання комплексного пошуку, коли підбирається танцівник, тренер та локація для занять танцями;

- розділ «Локації» дозволяє з випадуючого переліку обирати можливості пошуку, додавання власної локації, закріпленої за танцівником, відтворення закріплених локацій за собою;

– розділ «Тренери» дозволяє з випадуючого переліку обирати можливості пошуку, відтворення закріплених за танцівником або його парами тренерів з танців, підтриманих ним тренерів з танців, прокоментованих, подані заявки на закріплення тренера з танців, відтворення тих тренерів з танців, за якими слідує користувач, відтворення доступних для занять танцями у закріплених тренерів варіантів;

– розділ «Анкета» дозволяє з випадуючого переліку обирати можливості переходу до внесення даних про свої смаки та вподобання, створення власних матеріалів щодо танцювального досвіду.

Анкета танцівника для будь-якого іншого користувача-танцівника дозволяє отримати про нього інформацію (рис. 4.2).



Іван Мотренко

Підтримали: 1

Підтримати

Коментарів: 1

Про танцівника

Колір волосся: чорний

Колір очей: карі

Зріст: 175 см

Вага: 72 кг

Паління: ні

Стилі танців



Танго

просунутий

Стилі танців



Сальса

просунутий

Рисунок 4.2 – Анкета танцівника

Анкета танцівника в такому випадку включає можливості створення або керування створеною танцювальною парою між танцівниками, слідкування за танцівником або скасування такої можливості, підтримування або скасування підтримки танцівника, перегляду коментарів за танцівником від тренерів та інших танцівників, створення власного коментаря, перегляду закріплених за танцівником локацій для заняття танцями.

Тренер з танців після авторизації має доступне меню, що включає можливості у вигляді розділів:

- розділ «Танцівники» дозволяє з випадуючого переліку обирати можливості пошуку, відтворення підтриманих тренером з танців танцівників, танцівників, яких він прокоментував;

- розділ «Локації» дозволяє з випадуючого переліку обирати можливості пошуку, додавання власної локації, відтворення закріплених за собою локацій для занять танцями;

- розділ «Тренери» дозволяє з випадуючого переліку обирати можливості пошуку тренерів, відтворення запланованих занять з танців у тренера, графіку занять, тренерів, яких підтримав цей тренер або яких він прокоментував, подані заявки на заняття тренера (рис. 4.3), звідки можна переглянути або всі, або задати потрібну дату, закріплені пари або танцівники за тренером;

- пункт «Анкета» дозволяє вносити інформацію про свої тренерські характеристики.

Усі ці можливості сукупно дозволяють виконувати основні функції, доступні тренеру з танців, а інша частина з них доступна при переході з відповідних сторінок для подальшої роботи в залежності від виконаного вибору.

Під час керування поданими заявками на заняття тренера він може підтримати заявку, відхилити її.

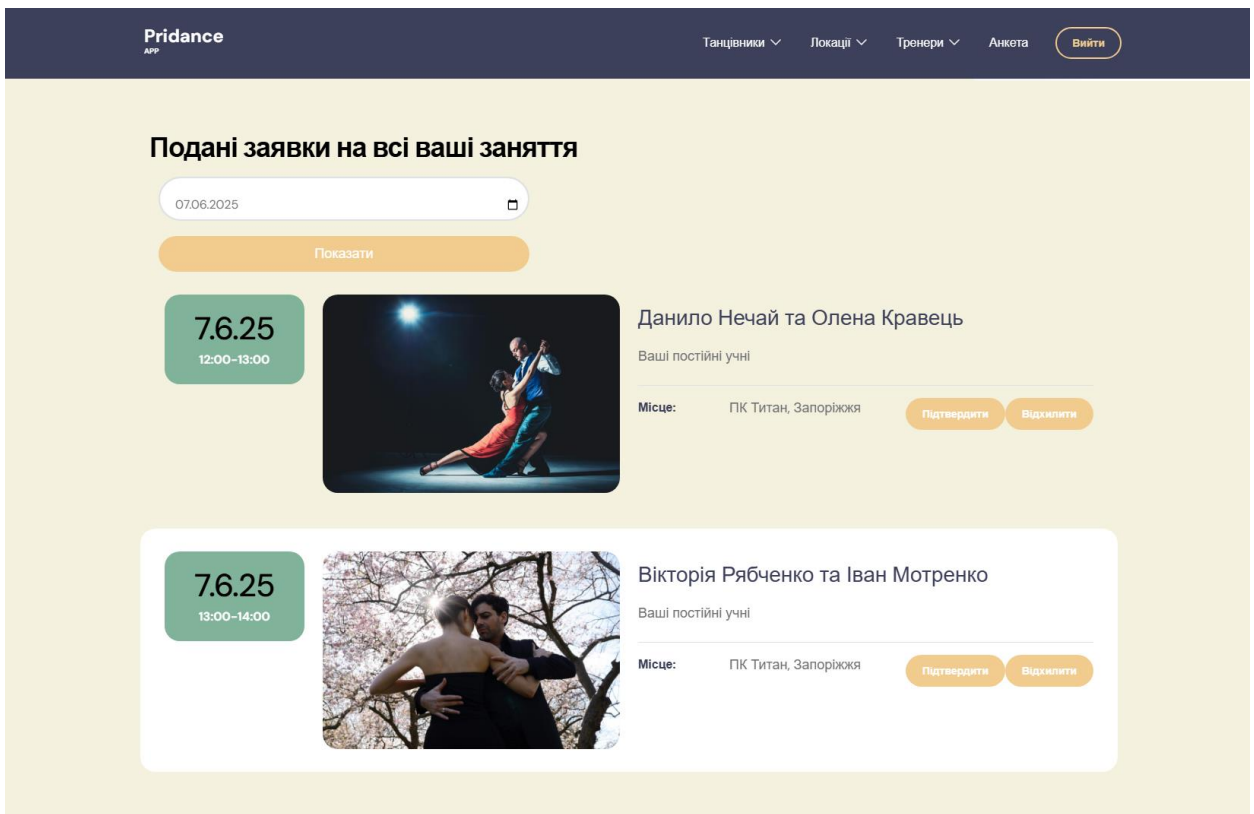


Рисунок 4.3 – Керування поданими заявками на заняття з танців тренера

4.3 Тестування програми організації індивідуальних занять танцями

Тестування програми організації індивідуальних занять танцями відбувалось за повним переліком функціональних можливостей, що були заплановані, зі створенням чек-листа та перевіркою кожної з них у варіанті для кожної ролі користувача (табл. 4.1). Якщо така функція у певній ролі не виконувалась, тобто не мала бути доступною, то позначка не ставилась. Якщо результат був коректним щодо виконання функції, то ставилась позначка плюс, у протилежному випадку – мінус. Однак некоректних результатів зафіксовано не було. Як у випадку доступу з персонального комп'ютеру та ноутбуку, так і у випадку мобільного пристрою усі функції відпрацювали коректно під час підсумкового тестування.

Таблиця 4.1 – Чек-лист тестування програми організації індивідуальних занять танцями

Функція	Неавторизований користувач	Танцівник	Тренер з танців
1	2	3	4
Пошук партнерів для занять танцями	+	+	+
Відтворення анкети танцівника	+	+	+
Пошук локацій для занять танцями	+	+	+
Відтворення доступних варіантів локацій для занять танцями у танцівника	+	+	+
Відтворення доступних варіантів локацій для занять танцями у тренера з танців	+	+	+
Відтворення сторінки тренера для занять танцями	+	+	+
Відтворення доступних варіантів індивідуальних занять танцями у тренера	+	+	+
Пошук тренерів для занять танцями	+	+	+
Відтворення коментарів за тренером з танців	+	+	+
Відтворення коментарів за танцівником	+	+	+
Авторизація	+	+	+
Реєстрація	+	+	+
Відтворення графіку індивідуальних занять танцями пари		+	

Продовження таблиці 4.1

1	2	3	4
Визначення власного опису смаків та досвіду в танцях		+	
Розміщення доступного власного варіанту локації для занять танцями		+	+
Подання заявки стосовно конкретного індивідуального заняття танцями у тренера		+	
Комплексний пошук партнера, локації та тренера для занять танцями		+	
Розміщення матеріалів за танцювальним досвідом		+	
Керування заявками стосовно створення пар для індивідуальних занять танцями		+	
Закріплення тренера для індивідуальних занять танцями пари		+	
Закріплення тренера для індивідуальних занять танцями танцівника		+	
Закріплення локації для індивідуальних занять танцями танцівника		+	
Закріплення локації для індивідуальних занять танцями пари		+	
Підтримування танцівника		+	+
Підтримування тренера з танців		+	+

Продовження таблиці 4.1

1	2	3	4
Відтворення даних підтриманих танцівників користувачем		+	+
Відтворення даних підтриманих тренерів користувачем		+	+
Коментування анкети танцівника		+	+
Відтворення даних прокоментованих танцівників		+	+
Коментування сторінки тренера з танців		+	+
Відтворення даних прокоментованих тренерів		+	+
Визначення слідкування за танцівником		+	
Визначення слідкування за тренером з танців		+	
Відтворення даних тренерів, за якими слідкує користувач		+	
Відтворення даних танцівників, за якими слідкує користувач		+	
Відтворення доступних варіантів індивідуальних занять танцями у власних тренерів		+	
Планування індивідуального заняття танцями пари		+	
Скасування визначення слідкування за тренером з танців		+	

Продовження таблиці 4.1

1	2	3	4
Скасування визначення слідкування за танцівником		+	
Скасування доступного власного варіанту локації для занять танцями		+	
Скасування власної заявки стосовно конкретного індивідуального заняття танцями у тренера		+	
Редагування запису про матеріал за танцювальним досвідом		+	
Вилучення запису про матеріал за танцювальним досвідом		+	
Скасування закріплення тренера для індивідуальних занять танцями пари		+	+
Скасування закріплення тренера для індивідуальних занять танцями танцівника		+	+
Подання заявки на створення пари для індивідуальних занять танцями		+	
Відтворення поданих заявок на створення пари для індивідуальних занять танцями		+	
Відтворення створених пар для індивідуальних занять танцями		+	
Скасування пари для індивідуальних занять танцями		+	
Відхилення пари для індивідуальних занять танцями		+	

Кінець таблиці 4.1

1	2	3	4
Скасування закріплення локації для індивідуальних занять танцями танцівника		+	
Скасування закріплення локації для індивідуальних занять танцями пари		+	
Скасування підтримування танцівника		+	+
Скасування підтримування тренера з танців		+	+
Визначення тренерських характеристик			+
Визначення варіантів індивідуальних занять танцями у тренера			+
Вилучення варіанта індивідуальних занять танцями у тренера			+
Керування поданою заявкою стосовно конкретного індивідуального заняття танцями цього тренера			+
Відтворення графіку запланованих індивідуальних занять танцями цього тренера			+
Відтворення заявок за індивідуальними заняттями танцями цього тренера			+
Відтворення пар та танцівників, тренування яких закріплено за цим тренером			+

4.4 Висновки за розділом 4

Усі функціональні можливості, які мали б надаватися програмою

організації індивідуальних занять танцями, були згруповані в залежності від ролі користувача у відповідні розділи меню з приведенням функції у такому розділі або з можливістю переходу до такої функції зі сторінки, що реалізує одну з функцій з розділу меню. Таким чином усі функції були реалізовані, а під час тестування перевірені з отриманням успішних результатів.

ВИСНОВКИ

Мета виконаної роботи була в тому, щоб розробити ПЗ організації індивідуальних занять танцями для підтримки супутнього процесу на його основних етапах. На основі виконання кожного окремого завдання було досягнуто мету роботи.

На основі огляду програм, які вже існують та могли би застосовуватися для підтримки частини етапів даного процесу, було визначено концепцію розроблюваної програми організації індивідуальних занять танцями, яка дозволила їй суттєво відрізнитися від наявних аналогів у першу чергу через організацію підтримки цілісно зазначеного процесу. Тобто програма не тільки виконує пошук партнерів для танців, але і роблячи акцент на можливостях індивідуальних занять, інтегрує пошук партнерів з пошуком тренерів, майданчиків.

У розробці програми використано мову Python і фреймворк Django разом з мовами HTML, CSS, JavaScript для веброзробки, PostgreSQL для підтримки у складі веброзробки роботи з базою даних.

Отримані результати включають всі етапи виконання роботи, враховуючи визначення концепції програми організації індивідуальних занять танцями на основі виявлення відсутності готового рішення, встановлення вимог до програми, структури бази даних, розроблення всіх програмних одиниць та об'єднання їх у єдиний вебзастосунок, успішне тестування створеного вебзастосунку.

Розроблена як підсумок програма призначена для забезпечення підтримки занять танцями на етапах пошуку партнерів для створення пари, підбору тренера для пари або окремого танцівника, підбору локацій для таких занять, комплексного вирішення таких проблем, планування самих індивідуальних занять танцями.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Студія танців в Києві для дорослих та дітей – Mixstyle [Електрон. ресурс]. – Режим доступу : <https://mixstyle.com.ua/>.
2. DancePartner.com – Find a dance partner who shares your passion for dance – World-wide Dance Partner Search [Electronic resource]. – Access mode : <https://www.dancepartner.com/>.
3. DanceSport [Електрон. ресурс] : Спортивні бальні танці. – Режим доступу : <http://danceinfo.com.ua/>.
4. ЗНАКОМКА – прості знайомства для дружби та спілкування [Електрон. ресурс]. – Режим доступу : <https://znakomka.in.ua/>.
5. Using Django. Django documentation. Django [Electronic resource]. – Access mode : <https://docs.djangoproject.com/en/5.2/topics/>.
6. Django introduction – Learn web development. MDN [Electronic resource]. – Access mode : https://developer.mozilla.org/en-US/docs/Learn_web_development/Extensions/Server-side/Django/Introduction.
7. Laravel – The PHP Framework For Web Artisans [Electronic resource]. – Access mode : <https://laravel.com/>.
8. GitHub – laravel/laravel [Electronic resource] : Laravel is a web application framework with expressive, elegant syntax. – Access mode : <https://github.com/laravel/laravel>.
9. Free Template 587 Tiya Golf Club [Electronic resource]. – Access mode : <https://templatemo.com/tm-587-tiya-golf-club>.

ДОДАТОК А
Технічне завдання

Вступ

Танці здавна привертають увагу людей, сформувавши цілу культуру навколо. Танці бувають індивідуальними, парними, можуть відбуватися у групах, а можуть організовуватися індивідуально для окремого танцівника, пари. При цьому танці потребують якісної підготовки, навчання для отримання задовільних результатів. Відповідно у всіх випадках окрім того, коли танцівник навчається повністю самостійно, він потребує взаємодії з іншими сторонами даного процесу, включаючи тренерів, інших танцівників. Також питанням є те, де саме такі заняття танцями будуть відбуватися, в який саме час. Це питання планування індивідуальних, групових занять танцями. І всі вони мають бути вирішені та вирішуються на практиці. Застосування програмного забезпечення, яке б повноцінно охопило всі складові цього процесу або принаймні основні з них, було би доцільним і актуальним.

A.1 Підстави для розробки

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

A.2 Призначення розробки

Розробка програми організації індивідуальних занять танцями призначена для забезпечення підтримки занять танцями на етапах пошуку партнерів для створення пари, підбору тренера для пари або окремого танцівника, підбору локацій для таких занять, комплексного вирішення таких проблем, планування самих індивідуальних занять танцями.

A.3 Основні вимоги до програми

A.3.1 Вимоги до функціональних характеристик

Вимоги до програми організації індивідуальних занять танцями за набором функцій складають:

- визначення власного опису смаків та досвіду в танцях;
- пошук партнерів для занять танцями;
- розміщення доступного варіанту локації для занять танцями у танцівника, тренера, у тому числі приватної локації;
- відтворення анкети танцівника;
- пошук локацій для занять танцями;
- відтворення доступних варіантів локацій для занять танцями у танцівника, тренера;
- визначення тренерських характеристик;
- відтворення сторінки тренера для занять танцями;
- визначення варіантів індивідуальних занять танцями у тренера;
- відтворення доступних варіантів індивідуальних занять танцями у тренера;
- подання заявки та керування заявкою стосовно конкретного індивідуального заняття танцями у тренера;
- відтворення графіку запланованих індивідуальних занять танцями тренера;
- відтворення заявок за індивідуальними заняттями танцями тренера;
- пошук тренерів для занять танцями;
- комплексний пошук партнера, локації та тренера для занять танцями;
- розміщення матеріалів за танцювальним досвідом;
- створення пари для індивідуальних занять танцями;
- керування заявками стосовно створення пар для індивідуальних занять танцями;
- закріплення тренера для індивідуальних занять танцями пари або

танцівника та його скасування;

- закріплення локації для індивідуальних занять танцями пари або танцівника та скасування;

- підтримування танцівника, тренера або скасування такої підтримки;

- відтворення даних підтриманих танцівників, тренерів користувачем;

- коментування анкети танцівника;

- відтворення даних прокоментованих танцівників користувачем;

- коментування сторінки тренера;

- відтворення даних прокоментованих тренерів користувачем;

- реєстрація танцівників та тренерів;

- відтворення коментарів за тренером або танцівником;

- визначення слідкування за танцівником, тренером або його скасування;

- відтворення даних танцівників, тренерів, за якими слідкує користувач;

- відтворення доступних варіантів індивідуальних занять танцями у власних тренерів;

- відтворення пар та танцівників, тренування яких закріплено за тренером;

- планування індивідуального заняття танцями пари;

- відтворення графіку індивідуальних занять танцями пари;

- авторизація танцівників та тренерів.

A.3.2 Умови експлуатації

Експлуатація програми організації індивідуальних занять танцями повинна виконуватись за забезпечення умов, які включають програмну частину сервера. Вимог до клієнтської частини не висувається, адже наявність встановленої глобальної мережі, встановленого браузера вже реалізують ті вимоги, які мають бути забезпечені на клієнтському пристрої. Вимоги ж до

програмної частини сервера виходять з тих рішень, які прийняті в роботі щодо вибору засобів розробки. Також необхідна наявність адміністратора програми, який би контролював роботу користувачів, вносив дані до програми, які знаходитимуться в базі даних.

А.3.3 Вимоги до складу та параметрів технічних засобів

Технічно сервер повинен відповідати мінімально рівню, який включає 4 Гб оперативної пам'яті для забезпечення одночасної підтримки звернень до програми, 2 ядра процесора, 50 Гб для організації сховища даних. Такі параметри є мінімальними, враховують необхідність багатокористувацької роботи, але далі можуть бути переоцінені в залежності від того, як має розширюватися коло користувачів програми та кількість запитів до неї.

А.4 Порядок контролю та приймання

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

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

ДОДАТОК Б
Текст програми

```

from .models import Dancecity
from .models import Danceuser
from .models import Danceuserphoto
from .models import Dancestylecategory
from .models import Dancestyle
from .models import Danceuserstyle
from .models import Dancegoal
from .models import Dancepair
from .models import Danceplace
from .models import Dancerplace
from .models import Danceteacher
from .models import Danceteacherphoto
from .models import Danceteacherstyle
from django.shortcuts import render
from .models import Danceteacherplace
from .models import Danceteacherlesson
from .models import Danceteachercomment
from .models import Danceteacherusercomment
from .models import Danceusercomment
from .models import Danceuserteachercomment
from .models import Danceteacherlessonpair
from .models import Danceteacherlessondancer
from .models import Danceuserstylematerial
from django.core.paginator import Paginator
from .models import Dancepairteacher
from .models import Danceuserteacher
from .models import Dancepairplace
from .models import Danceuserplace
from .models import Dancepairmeeting
from django.db.models import Q

#Відтворення доступних варіантів локацій для занять танцями
у танцівника
def view_danceuser_locations(request, danceuser = None):
    if not danceuser:
        if request.user.is_authenticated:
            try:
                danceuser =
Danceuser.objects.get(useraccountindjango__username =
request.user.username)
            except Danceuser.DoesNotExist:
                danceuser = None
        else:

```

```

        danceuser = None
    else:
        try:
            danceuser =
Danceuser.objects.get(useraccountindjango = request.user)
        except Danceuser.DoesNotExist:
            danceuser = None
    if not danceuser:
        return render(request, "pridanceapp/error.html",
{"info": "Не вдалось встановити такого танцівника. Вибачте за незручності"})
        dancer_places =
Dancerplace.objects.select_related("danceplace",
"danceplace__dancacity").filter(danceuser = danceuser)
        dancer_places = dancer_places.order_by("-
dancerplacedatecreation")
        dancer_places_number = dancer_places.count()
        dancer_places_paginator = Paginator(dancer_places, 10)
        dancer_places_part = request.GET.get('page')
        dancer_places =
dancer_places_paginator.get_page(dancer_places_part)
        return render(request, "pridanceapp/danceplaces.html",
{"places": dancer_places, "placesnumber": dancer_places_number})

#Відтворення доступних варіантів локацій для занять танцями
у тренера з танців
    def view_danceteacher_locations(request, danceteacher =
None):
        if not danceteacher:
            if request.user.is_authenticated:
                try:
                    danceteacher =
Danceteacher.objects.get(useraccountindjango__username =
request.user.username)
                except Danceteacher.DoesNotExist:
                    danceteacher = None
            else:
                danceteacher = None
        else:
            try:
                danceteacher =
Danceteacher.objects.get(useraccountindjango = request.user)
            except Danceuser.DoesNotExist:
                danceteacher = None
        if not danceteacher:

```

```

        return render(request, "pridanceapp/error.html",
{"info": "Не вдалось встановити такого тренера з танців. Вибачте за незручності"})
        dance_teacher_places =
Danceteacherplace.objects.select_related("danceplace",
"danceplace__dancecity").filter(danceteacher = danceteacher)
        dance_teacher_places = dance_teacher_places.order_by("-
dancerplacedatecreation")
        dance_teacher_places_number =
dance_teacher_places.count()
        dance_teacher_places_paginator =
Paginator(dance_teacher_places, 10)
        dance_teacher_places_part = request.GET.get('page')
        dance_teacher_places =
dance_teacher_places_paginator.get_page(dance_teacher_places_par
t)
        return render(request,
"pridanceapp/danceteachplaces.html", {"places":
dance_teacher_places, "placesnumber":
dance_teacher_places_number})

```

#Відтворення анкети танцівника

```

def view_danceuser(request, danceuser = None):
    if not danceuser:
        if request.user.is_authenticated:
            try:
                danceuser =
Danceuser.objects.select_related("useraccountindjango",
"dancecity").prefetch_related("danceusergoals",
"danceusergoals__dancegoal", "danceusersupports",
"danceusersupports__danceuser2",
"danceusersupports__danceuser2__useraccountindjango",
"danceteachersupports", "danceteachersupports__danceteacher",
"danceteachersupports__danceteacher__useraccountindjango").get(u
seraccountindjango__username = request.user.username)
            except Danceuser.DoesNotExist:
                danceuser = None
        else:
            danceuser = None
    else:
        try:
            danceuser =
Danceuser.objects.select_related("useraccountindjango",
"dancecity").prefetch_related("danceusergoals",
"danceusergoals__dancegoal", "danceusersupports",

```

```

"danceusersupports__dancer2",
"danceusersupports__dancer2__useraccountindjango",
"danceteachersupports", "danceteachersupports__danceteacher",
"danceteachersupports__danceteacher__useraccountindjango").get(u
seraccountindjango = request.user)
    except Danceuser.DoesNotExist:
        dancer = None
    if not dancer:
        return render(request, "pridanceapp/error.html",
{"info": "Не вдалось встановити такого танцівника. Вибачте за
незручності"})
    dancer_places =
Dancerplace.objects.select_related("danceplace",
"danceplace__dancecity").filter(dancer = dancer)
    dancer_places = dancer_places.order_by("-
dancerplacedatecreation")[:3]
    dancer_places_number = dancer_places.count()
    dancer_photos = Danceuserphoto.objects.filter(dancer =
dancer)
    dance_styles =
Danceuserstyle.objects.select_related("dancestyle").filter(dance
user = dancer)
    dancer_pairs =
Dancepair.objects.select_related("dancer1", "dancer2",
"dancer1__useraccountindjango",
"dancer2__useraccountindjango").filter(Q(dancer1 =
dancer)|Q(dancer2 = dancer))
    dancer_pairs = dancer_pairs.filter(dancepairstatus =
"approved")
    dancer_pairs_number = dancer_pairs.count()
    for dancer_pair in dancer_pairs:
        dancer_pair_places =
Danceuserplace.objects.select_related("danceplace",
"danceplace__dancecity").filter(dancepair = dancer_pair)
        dancer_pair_teachers =
Dancepairteacher.objects.select_related("danceteacher",
"danceteacher__useraccountindjango").filter(dancepair =
dancer_pair)
        dancer_teachers =
Danceuserteacher.objects.select_related("danceteacher",
"danceteacher__useraccountindjango").filter(dancer =
dancer)
        dancer_comments =
Danceusercomment.objects.select_related("dancer2",

```

```

"danceuser2__useraccountindjango").order_by("-
danceucommentdate")
        dancer_comments_number = dancer_comments.count()
        dancer_comments = dancer_comments[:5]
        danceteacher_comments =
Danceuserteachercomment.objects.select_related("danceteacher",
"danceteacher__useraccountindjango").order_by("-
danceucommentdate")
        danceteacher_comments = danceteacher_comments.count()
        danceteacher_comments = danceteacher_comments[:5]
        dancer_materials =
Danceuserstylematerial.objects.select_related("danceuserstyle",
"danceuserstyle__dancestyle").filter(danceuserstyle__danceuser =
danceuser)
        danceuser.teachesupports_number =
danceuser.danceteachersupports.count()
        danceuser.danceusersupports_number =
danceuser.danceusersupports.count()
        return render(request, "pridanceapp/dancer.html",
{"places": dancer_places, "placesnumber": dancer_places_number,
"photos": dancer_photos, "styles": dance_styles, "comments":
dancer_comments, "teachercomments": danceteacher_comments,
"commentnumber": dancer_comments_number, "teachcommentnumber":
danceteacher_comments, "materials": dancer_materials, "pairs":
dancer_pairs, "pairsnumber": dancer_pairs_number, "teachers":
dancer_teachers dancer_teachers })

#Відтворення коментарів за танцівником
def view_danceuser_comments(request, from = False,
danceuser = None):
    if not danceuser:
        if request.user.is_authenticated:
            try:
                danceuser =
Danceuser.objects.select_related("useraccountindjango",
"dancecity").prefetch_related("danceusergoals",
"danceusergoals__dancegoal").get(useraccountindjango__username =
request.user.username)
            except Danceuser.DoesNotExist:
                danceuser = None
        else:
            danceuser = None
    else:
        try:

```

```

        danceuser =
Danceuser.objects.select_related("useraccountindjango",
"dancecity").prefetch_related("danceusergoals",
"danceusergoals__dancegoal", "danceusersupports",
"danceusersupports__danceuser2",
"danceusersupports__danceuser2__useraccountindjango",
"danceteachersupports", "danceteachersupports__danceteacher",
"danceteachersupports__danceteacher__useraccountindjango").get(u
seraccountindjango = request.user)
        except Danceuser.DoesNotExist:
            danceuser = None
        if not danceuser:
            return render(request, "pridanceapp/error.html",
{"info": "Не вдалось встановити такого танцівника. Вибачте за
незручності"})
        if from:
            dancer_comments =
Danceusercomment.objects.select_related("danceuser1",
"danceuser1__useraccountindjango").filter(danceuser2 =
danceuser)
            dancer_comments = dancer_comments.order_by("-
danceucommentdate")
            dancer_comments_number = dancer_comments.count()
            dancer_comments_paginator =
Paginator(dancer_comments, 10)
            dancer_comments_part = request.GET.get('page')
            dancer_comments =
dancer_comments_paginator.get_page(dancer_comments_part)
            dancer_teacher_comments =
Danceteacherusercomment.objects.select_related("danceuser",
"danceuser__useraccountindjango").order_by("-danceucommentdate")
            dancer_teacher_comments_number =
dancer_teacher_comments.count()
            dancer_teacher_comments_paginator =
Paginator(dancer_teacher_comments, 10)
            dancer_teacher_comments_part =
request.GET.get('pageteach')
            dancer_teacher_comments =
dancer_teacher_comments_paginator.get_page(dancer_teacher_commen
ts_part)
        else:
            dancer_comments =
Danceusercomment.objects.select_related("danceuser2",
"danceuser2__useraccountindjango").filter(danceuser1 =
danceuser)

```

```

        dancer_comments = dancer_comments.order_by("-
danceucommentdate")
        dancer_comments_number = dancer_comments.count()
        dancer_comments_paginator =
Paginator(dancer_comments, 10)
        dancer_comments_part = request.GET.get('page')
        dancer_comments =
dancer_comments_paginator.get_page(dancer_comments_part)
        dancer_teacher_comments =
Danceuserteachercomment.objects.select_related("danceuser",
"danceuser__useraccountindjango").order_by("-danceucommentdate")
        dancer_teacher_number =
dancer_teacher_comments.count()
        dancer_teacher_comments_paginator =
Paginator(dancer_teacher_comments, 10)
        dancer_teacher_comments_part =
request.GET.get('pageteach')
        dancer_teacher_comments =
dancer_teacher_comments_paginator.get_page(dancer_teacher_commen
ts_part)
        return render(request, "pridanceapp/dancercomms.html",
{"comments": dancer_comments, "commentnumber":
dancer_comments_number, "teachcomments":
dancer_teacher_comments, "commentteachnumber":
dancer_teacher_number })

```

```

#Відтворення коментарів за тренером з танців
def view_danceteacher_comments(request, from = False,
danceteacher = None):
    if not danceteacher:
        if request.user.is_authenticated:
            try:
                danceteacher =
Danceteacher.objects.get(useraccountindjango__username =
request.user.username)
            except Danceteacher.DoesNotExist:
                danceteacher = None
        else:
            danceteacher = None
    else:
        try:
            danceteacher =
Danceteacher.objects.get(useraccountindjango = request.user)
        except Danceuser.DoesNotExist:
            danceteacher = None

```

```

        if not danceteacher:
            return render(request, "pridanceapp/error.html",
{"info": "Не вдалось встановити такого тренера з танців. Вибацте
за незручності"})
        if from:
            dance_teacher_comments =
Danceuserteachercomment.objects.select_related("danceuser",
"danceuser__useraccountindjango").filter(danceteacher =
danceteacher)
            dance_teacher_comments =
dance_teacher_comments.order_by("-danceucommentdate")
            dance_teacher_comments_number =
dance_teacher_comments.count()
            dance_teacher_comments_paginator =
Paginator(dance_teacher_comments, 10)
            dance_teacher_comments_part =
request.GET.get('page')
            dance_teacher_comments =
dance_teacher_comments_paginator.get_page(dance_teacher_comments
_part)
            dance_teacher_on_teachers_comments =
Danceteachercomment.objects.select_related("danceteacher2",
"danceteacher2__useraccountindjango").order_by("-
danceucommentdate")
            dance_teacher_on_teachers_comments_number =
dance_teacher_on_teachers_comments.count()
            dance_teacher_on_teachers_comments_paginator =
Paginator(dance_teacher_on_teachers_comments, 10)
            dance_teacher_on_teachers_comments_part =
request.GET.get('pageteach')
            dance_teacher_on_teachers_comments =
dance_teacher_on_teachers_comments_paginator.get_page(dance_teach
her_on_teachers_comments_part)
        else:
            dance_teacher_comments =
Danceteacherusercomment.objects.select_related("danceuser",
"danceuser__useraccountindjango").filter(danceteacher =
danceteacher)
            dance_teacher_comments =
dance_teacher_comments.order_by("-danceucommentdate")
            dance_teacher_comments_number =
dance_teacher_comments.count()
            dance_teacher_comments_paginator =
Paginator(dance_teacher_comments, 10)
            dance_teacher_comments_part =
request.GET.get('page')

```

```

        dance_teacher_comments =
dance_teacher_comments_paginator.get_page(dance_teacher_comments
_part)
        dance_teacher_on_teachers_comments =
Danceteachercomment.objects.select_related("danceteacher1",
"danceteacher1__useraccountindjango").order_by("-
danceucommentdate")
        dance_teacher_on_teachers_comments_number =
dance_teacher_on_teachers_comments.count()
        dance_teacher_on_teachers_comments_paginator =
Paginator(dance_teacher_on_teachers_comments, 10)
        dance_teacher_on_teachers_comments_part =
request.GET.get('pageteach')
        dance_teacher_on_teachers_comments =
dance_teacher_on_teachers_comments_paginator.get_page(dance_teach
er_on_teachers_comments_part)
        return render(request,
"pridanceapp/danceteachcomms.html", {"comments":
dance_teacher_comments, "commentnumber":
dance_teacher_comments_number, "teachercomments":
dance_teacher_on_teachers_comments, "teachercommentnumber":
dance_teacher_on_teachers_comments_number })

```

```

#Відтворення сторінки тренера для занять танцями
def view_danceteacher(request, danceteacher = None):
    if not danceteacher:
        if request.user.is_authenticated:
            try:
                danceteacher =
Danceteacher.objects.get(useraccountindjango__username =
request.user.username)
            except Danceteacher.DoesNotExist:
                danceteacher = None
            else:
                danceteacher = None
        else:
            try:
                danceteacher =
Danceteacher.objects.get(useraccountindjango = request.user)
            except Danceuser.DoesNotExist:
                danceteacher = None
            if not danceteacher:
                return render(request, "pridanceapp/error.html",
{"info": "Не вдалось встановити такого тренера з танців. Вибачте
за незручності"})

```

```

        dance_teacher_comments =
Danceteacherusercomment.objects.select_related("danceteacher",
"danceteacher__useraccountindjango").order_by("-
danceucommentdate")
        dance_teacher_comments_number =
dance_teacher_comments.count()
        dance_teacher_comments = dance_teacher_comments[:3]
        dance_teacher_on_teachers_comments =
Danceteachercomment.objects.select_related("danceteacher1",
"danceteacher1__useraccountindjango").order_by("-
danceucommentdate")
        dance_teacher_on_teachers_comments_number =
dance_teacher_on_teachers_comments.count()
        dance_teacher_on_teachers_comments =
dance_teacher_on_teachers_comments[:3]
        dance_teacher_places =
Danceteacherplace.objects.select_related("danceplace",
"danceplace__dancecity").filter(danceteacher = danceteacher)
        dance_teacher_places = dance_teacher_places.order_by("-
dancerplacedatecreation")
        dance_teacher_places_number =
dance_teacher_places.count()
        dance_teacher_places = dance_teacher_places[:3]
        return render(request, "pridanceapp/danceteacher.html",
{"comments": dance_teacher_comments, "commentnumber":
dance_teacher_comments_number, "teachercomments":
dance_teacher_on_teachers_comments, "teachercommentnumber":
dance_teacher_on_teachers_comments_number, "placesnumber":
dance_teacher_places_number, "places": dance_teacher_places })

#Відтворення графіку індивідуальних занять танцями пари
def view_dancepair_schedule(request, dancepair, meeting =
2, ordering = True, startdate = None, enddate = None):
    if request.user.is_authenticated:
        try:
            danceuser =
Danceuser.objects.select_related("useraccountindjango",
"dancecity").prefetch_related("danceusergoals",
"danceusergoals__dancegoal").get(useraccountindjango__username =
request.user.username)
            except Danceuser.DoesNotExist:
                danceuser = None
        else:
            danceuser = None
    if not danceuser:

```

```

        return render(request, "pridanceapp/error.html",
{"info": "Не вдалось встановити такого танцівника. Вибачте за незручності"})
    try:
        dancepair =
Dancepair.objects.select_related("danceuser1",
"danceuser1__useraccountindjango", "danceuser2",
"danceuser2__useraccountindjango").get(id = dancepair)
    except Dancepair.DoesNotExist:
        dancepair = None
    if not dancepair:
        return render(request, "pridanceapp/error.html",
{"info": "Не вдалось встановити таку пару для індивідуальних
занять танцями. Вибачте за незручності"})
    import datetime
    from datetime import datetime
    from dateutil.relativedelta import relativedelta
    if not startdate:
        startdate = datetime.datetime.now().date()
    else:
        startdate = datetime.strptime(startdate , '%Y-%m-
%d')

        startdate = startdate.date()
    if not enddate:
        enddate = startdate + relativedelta(months=1)
    else:
        enddate = datetime.strptime(enddate, '%Y-%m-%d')
        enddate = enddate.date()
    if meeting == 0:
        dance_pair_meetings =
Dancepairmeeting.objects.select_related("danceplace",
"danceplace__dancecity").filter(dancepair = dancepair)
        dance_pair_meetings =
dance_pair_meetings.filter(dancemeetstatus = "approved")
        dance_pair_meetings =
dance_pair_meetings.filter(dancemeetdate__gte = startdate)
        dance_pair_meetings =
dance_pair_meetings.filter(dancemeetdate__lte = enddate)
        if ordering:
            dance_pair_meetings =
dance_pair_meetings.order_by("-dancemeetdate")
        else:
            dance_pair_meetings =
dance_pair_meetings.order_by("dancemeetdate")

```

```

        dance_pair_meetings_count =
dance_pair_meetings.count()
        dance_pair_meetings_paginator =
Paginator(dance_pair_meetings, 10)
        dance_pair_meetings_part = request.GET.get('page')
        dance_pair_meetings =
dance_pair_meetings_paginator.get_page(dance_pair_meetings_part)
        return render(request,
"pridanceapp/pairschedule.html", {"meetings":
dance_pair_meetings, "number": dance_pair_meetings_count })
        elif meeting == 1:
            dance_pair_lessons =
Danceteacherlessonpair.objects.select_related("danceplace",
"danceplace__dancecity", "danceteacherlesson",
"danceteacherlesson__danceteacher",
"danceteacherlesson__danceteacher__useraccountindjango",
"danceteacherlesson__danceplace",
"danceteacherlesson__danceplace__dancecity").filter(dancepair =
dancepair)
            dance_pair_lessons =
dance_pair_lessons.filter(danceattendstatus = "approved")
            dance_pair_lessons =
dance_pair_lessons.filter(danceteacherlesson__dancelessondate__g
te = startdate)
            dance_pair_lessons =
dance_pair_lessons.filter(danceteacherlesson__dancelessondate__l
te = enddate)
            if ordering:
                dance_pair_lessons =
dance_pair_lessons.order_by("danceteacherlesson__dancelessondate
")
            else:
                dance_pair_lessons =
dance_pair_lessons.order_by("danceteacherlesson__dancelessondate
")
            dance_pair_lessons_count =
dance_pair_lessons.count()
            dance_pair_lessons_paginator =
Paginator(dance_pair_lessons, 10)
            dance_pair_lessons_part = request.GET.get('page')
            dance_pair_lessons =
dance_pair_lessons_paginator.get_page(dance_pair_lessons_part)
            return render(request,
"pridanceapp/pairschedule.html", {"meetings":
dance_pair_lessons, "number": dance_pair_lessons_count })
        else:

```

```

        dance_pair_meetings =
Dancepairmeeting.objects.select_related("danceplace",
"danceplace__dancecity").filter(dancepair = dancepair)
        dance_pair_meetings =
dance_pair_meetings.filter(dancemeetstatus = "approved")
        dance_pair_meetings =
dance_pair_meetings.filter(dancemeetdate__gte = startdate)
        dance_pair_meetings =
dance_pair_meetings.filter(dancemeetdate__lte = enddate)
        if ordering:
            dance_pair_meetings =
dance_pair_meetings.order_by("-dancemeetdate")
        else:
            dance_pair_meetings =
dance_pair_meetings.order_by("dancemeetdate")
            dance_pair_meetings_count =
dance_pair_meetings.count()
            dance_pair_lessons =
Danceteacherlessonpair.objects.select_related("danceplace",
"danceplace__dancecity", "danceteacherlesson",
"danceteacherlesson__danceteacher",
"danceteacherlesson__danceteacher__useraccountindjango",
"danceteacherlesson__danceplace",
"danceteacherlesson__danceplace__dancecity").filter(dancepair =
dancepair)
            dance_pair_lessons =
dance_pair_lessons.filter(danceattendstatus = "approved")
            dance_pair_lessons =
dance_pair_lessons.filter(danceteacherlesson__dancelessondate__g
te = startdate)
            dance_pair_lessons =
dance_pair_lessons.filter(danceteacherlesson__dancelessondate__l
te = enddate)
            if ordering:
                dance_pair_lessons =
dance_pair_lessons.order_by("danceteacherlesson__dancelessondate
")
            else:
                dance_pair_lessons =
dance_pair_lessons.order_by("danceteacherlesson__dancelessondate
")
                dance_pair_lessons_count =
dance_pair_lessons.count()
                pos = 0

```

```

        dance_pair_lessons = list(dance_pair_lessons)
        all_dance_meetings = []
        for dance_pair_meeting in dance_pair_meetings:
            pos2 = pos - 1
            while pos > pos2:
                pos2 = pos
                if ordering:
                    if
dance_pair_lessons[pos].danceteacherlesson.dancelessondate >
dance_pair_meeting.dancemeetdate:

all_dance_meetings.append(dance_pair_lessons[pos])
                pos = pos + 1
            else:

all_dance_meetings.append(dance_pair_meeting)
            else:
                if
dance_pair_lessons[pos].danceteacherlesson.dancelessondate <
dance_pair_meeting.dancemeetdate:

all_dance_meetings.append(dance_pair_lessons[pos])
                pos = pos + 1
            else:

all_dance_meetings.append(dance_pair_meeting)
            return render(request,
"pridanceapp/pairschedule.html", {"meetings":
all_dance_meetings, "number": dance_pair_lessons_count,
"meetnumber": dance_pair_meetings_count })

#Відтворення даних підтриманих танцівників користувачем
def view_danceuser_supports(request):
    danceteacher = None
    if request.user.is_authenticated:
        try:
            danceuser =
Danceuser.objects.select_related("useraccountindjango",
"dancecity").prefetch_related("danceusergoals",
"danceusergoals__dancegoal", "danceusersupports",
"danceusersupports__danceuser2",
"danceusersupports__danceuser2__useraccountindjango",
"danceteachersupports", "danceteachersupports__danceteacher",

```

```

"danceteachersupports__danceteacher__useraccountindjango").get(u
seraccountindjango__username = request.user.username)
    except Danceuser.DoesNotExist:
        danceuser = None
    else:
        danceuser = None
    if not danceuser and request.user.is_authenticated:
        try:
            danceteacher =
Danceteacher.objects.prefetch_related("danceusersupports",
"danceusersupports__danceuser",
"danceusersupports__danceuser__useraccountindjango").get(useracc
ountindjango__username = request.user.username)
            except Danceteacher.DoesNotExist:
                danceteacher = None
            if not danceuser and not danceteacher:
                return render(request, "pridanceapp/error.html",
{"info": "Не вдалось встановити такого користувача. Вибачте за
незручності"})
            if danceuser:
                dancer_supports_number =
danceuser.danceusersupports.count()
                dancer_supports_paginator =
Paginator(danceuser.danceusersupports, 10)
                dancer_supports_part = request.GET.get('page')
                dancer_supports =
dancer_supports_paginator.get_page(dancer_supports_part)
            else:
                dancer_supports_number =
danceteacher.danceusersupports.count()
                dancer_supports_paginator =
Paginator(danceteacher.danceusersupports, 10)
                dancer_supports_part = request.GET.get('page')
                dancer_supports =
dancer_supports_paginator.get_page(dancer_supports_part)
            return render(request, "pridanceapp/supports.html",
{"supports": dancer_supports, "number": dancer_supports_number
})

#Відтворення даних підтриманих тренерів користувачем
def view_danceteacher_supports(request):
    danceteacher = None
    if request.user.is_authenticated:
        try:

```

```

        danceuser =
Danceuser.objects.select_related("useraccountindjango",
"dancecity").prefetch_related("danceusergoals",
"danceusergoals__dancegoal", "danceusersupports",
"danceusersupports__danceuser2",
"danceusersupports__danceuser2__useraccountindjango",
"danceteachersupports", "danceteachersupports__danceteacher",
"danceteachersupports__danceteacher__useraccountindjango").get(u
seraccountindjango__username = request.user.username)
        except Danceuser.DoesNotExist:
            danceuser = None
    else:
        danceuser = None
    if not danceuser and request.user.is_authenticated:
        try:
            danceteacher =
Danceteacher.objects.prefetch_related("danceteachersupports",
"danceteachersupports__danceteacher1",
"danceteachersupports__danceteacher1__useraccountindjango").get(
useraccountindjango__username = request.user.username)
            except Danceteacher.DoesNotExist:
                danceteacher = None
        if not danceuser and not danceteacher:
            return render(request, "pridanceapp/error.html",
{"info": "Не вдалось встановити такого користувача. Вибачте за
незручності"})
        if danceuser:
            dancer_supports_number =
danceuser.danceteachersupports.count()
            dancer_supports_paginator =
Paginator(danceuser.danceteachersupports, 10)
            dancer_supports_part = request.GET.get('page')
            dancer_supports =
dancer_supports_paginator.get_page(dancer_supports_part)
        else:
            dancer_supports_number =
danceteacher.danceteachersupports.count()
            dancer_supports_paginator =
Paginator(danceteacher.danceteachersupports, 10)
            dancer_supports_part = request.GET.get('page')
            dancer_supports =
dancer_supports_paginator.get_page(dancer_supports_part)
        return render(request,
"pridanceapp/teachsupports.html", {"supports": dancer_supports,
"number": dancer_supports_number })

```

```

#Відтворення даних танцівників, за якими слідкує користувач
def view_danceuser_follows(request):
    if request.user.is_authenticated:
        try:
            danceuser =
Danceuser.objects.select_related("useraccountindjango",
"dancecity").prefetch_related("danceusergoals",
"danceusergoals__dancegoal", "danceuserfollows",
"danceuserfollows__danceuser1", "
danceuserfollows__danceuser1__useraccountindjango",
"danceteacherfollows", "danceteacherfollows__danceteacher",
"danceteacherfollows__danceteacher__useraccountindjango").get(us
eraccountindjango__username = request.user.username)
        except Danceuser.DoesNotExist:
            danceuser = None
    else:
        danceuser = None
    if not danceuser:
        return render(request, "pridanceapp/error.html",
{"info": "Не вдалось встановити такого користувача. Вибачте за
незручності"})
        dancer_follows_number =
danceuser.danceuserfollows.count()
        dancer_follows_paginator =
Paginator(danceuser.danceuserfollows, 10)
        dancer_follows_part = request.GET.get('page')
        dancer_follows =
dancer_follows_paginator.get_page(dancer_follows_part)
        return render(request, "pridanceapp/follows.html",
{"follows": dancer_follows, "number": dancer_follows_number })

#Відтворення даних тренерів, за якими слідкує користувач
def view_danceuser_teachers_follows(request):
    if request.user.is_authenticated:
        try:
            danceuser =
Danceuser.objects.select_related("useraccountindjango",
"dancecity").prefetch_related("danceusergoals",
"danceusergoals__dancegoal", "danceuserfollows",
"danceuserfollows__danceuser1", "
danceuserfollows__danceuser1__useraccountindjango",
"danceteacherfollows", "danceteacherfollows__danceteacher",
"danceteacherfollows__danceteacher__useraccountindjango").get(us
eraccountindjango__username = request.user.username)
        except Danceuser.DoesNotExist:

```

```

        danceuser = None
    else:
        danceuser = None
    if not danceuser:
        return render(request, "pridanceapp/error.html",
{"info": "Не вдалось встановити такого користувача. Вибачте за незручності"})
        dancer_follows_number =
danceuser.danceteacherfollows.count()
        dancer_follows_paginator =
Paginator(danceuser.danceteacherfollows, 10)
        dancer_follows_part = request.GET.get('page')
        dancer_follows =
dancer_follows_paginator.get_page(dancer_follows_part)
        return render(request, "pridanceapp/teachfollows.html",
{"follows": dancer_follows, "number": dancer_follows_number })

#Відтворення даних прокоментованих танцівників
def view_danceusers_commented(request):
    if request.user.is_authenticated:
        try:
            danceuser =
Danceuser.objects.select_related("useraccountindjango",
"dancecity").prefetch_related("danceusergoals",
"danceusergoals__dancegoal").get(useraccountindjango__username =
request.user.username)
            except Danceuser.DoesNotExist:
                danceuser = None
        else:
            danceuser = None
    if not danceuser:
        return render(request, "pridanceapp/error.html",
{"info": "Не вдалось встановити такого танцівника. Вибачте за незручності"})
        dancer_comments =
Danceusercomment.objects.select_related("danceuser1",
"danceuser1__useraccountindjango").filter(danceuser2 =
danceuser)
        comments_on_dancer =
dancer_comments.values('danceuser1').distinct()
        for comment_on_dancer in comments_on_dancer:
            comment_on_dancer.comments =
Danceusercomment.objects.select_related("danceuser1",
"danceuser1__useraccountindjango").filter(danceuser1 =
comment_on_dancer, danceuser2 = danceuser)

```

```

        return render(request,
"pridanceapp/ondancercomms.html", {"dancers": dancer_comments })

#Відтворення даних прокоментованих тренерів
def view_danceteachers_commented(request):
    if request.user.is_authenticated:
        try:
            danceuser =
Danceuser.objects.select_related("useraccountindjango",
"dancecity").prefetch_related("danceusergoals",
"danceusergoals__dancegoal").get(useraccountindjango__username =
request.user.username)
        except Danceuser.DoesNotExist:
            danceuser = None
        else:
            danceuser = None
        if not danceuser:
            return render(request, "pridanceapp/error.html",
{"info": "Не вдалось встановити такого тренера з танців. Вибачте
за незручності"})
            dance_teacher_comments =
Danceteacherusercomment.objects.select_related("danceteacher",
"danceteacher__useraccountindjango").filter(danceuser =
danceteacher)
            comments_on_teacher =
dance_teacher_comments.values("danceteacher").distinct()
            for comment_on_teacher in comments_on_teacher:
                comment_on_teacher.comments =
Danceteacherusercomment.objects.select_related("danceteacher",
"danceteacher__useraccountindjango").filter(danceteacher =
comments_on_teacher, danceuser = danceuser)
            return render(request,
"pridanceapp/teachcommented.html", {"teachers":
comments_on_teacher })

#Відтворення поданих заявок на створення пари для
індивідуальних занять танцями
def view_dancepair_applications(request, to = True):
    if request.user.is_authenticated:
        try:
            danceuser =
Danceuser.objects.select_related("useraccountindjango",
"dancecity").prefetch_related("danceusergoals",

```

```

"danceusergoals__dancegoal").get(useraccountindjango__username =
request.user.username)
    except Danceuser.DoesNotExist:
        danceuser = None
    else:
        danceuser = None
    if not danceuser:
        return render(request, "pridanceapp/error.html",
{"info": "Не вдалось встановити такого тренера з танців. Вибачте
за незручності"})
    if to:
        dance_pairs =
Dancepair.objects.select_related("danceuser2",
"danceuser2__useraccountindjango").filter(danceuser1 =
danceuser, dancepairstatus = "created")
        dance_pairs = dance_pairs.order_by("-
dancepairdatecreation")
        return render(request,
"pridanceapp/pairsappl.html", {"pairs": dance_pairs })
    else:
        dance_pairs =
Dancepair.objects.select_related("danceuser1",
"danceuser1__useraccountindjango").filter(danceuser2 =
danceuser, dancepairstatus = "created")
        dance_pairs = dance_pairs.order_by("-
dancepairdatecreation")
        return render(request,
"pridanceapp/pairsappl.html", {"pairs": dance_pairs })

```

ДОДАТОК В
Слайди презентації

Міністерство освіти і науки України
Національний університет «Запорізька політехніка»
кафедра програмних засобів

Тема дипломної кваліфікаційної роботи бакалавра
Програмне забезпечення організації
індивідуальних занять танцями
Software for Organization of Private Dance
Classes

Виконавець: Олександра ЗІВЕР

Керівник: к.т.н., доцент Валерій ЛЬОВКІН

2025

Рисунок В.1 – Початковий слайд

Об'єкт, предмет, мета роботи

Об'єкт роботи – процес організації індивідуальних занять танцями.

Предмет роботи – програмне забезпечення організації індивідуальних занять танцями.

Мета роботи – розробити програмне забезпечення організації індивідуальних занять танцями для підтримки супутнього процесу на його основних етапах.

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

Завдання роботи

- аналіз існуючого стану проблеми розробки програмного забезпечення організації індивідуальних занять танцями;
- визначення вимог до програмного забезпечення організації індивідуальних занять танцями;
- проєктування програмного забезпечення організації індивідуальних занять танцями;
- реалізація і тестування програмного забезпечення організації індивідуальних занять танцями.

3

Рисунок В.3 – Завдання роботи

Порівняння аналогів та розробки стосовно організації індивідуальних занять танцями

Критерій	DancePartner	DanceInfo	Mix Style	Розробка
Пошук партнерів для танців	За критеріями	Статичні оголошення	Відсутній	За критеріями
Надання матеріалів про власний рівень	Відсутнє	Відсутнє	Відсутнє	Так
Вибір тренерів для індивідуальних занять танцями	Відсутній	Відсутній	3 тренерів студії для загальних занять	Так
Розгляд майданчиків для занять танцями	Відсутній	Розділ присутній, але ненаповнений	Студії мережі	Так

4

Рисунок В.4 – Порівняння аналогів та розробки стосовно організації індивідуальних занять танцями

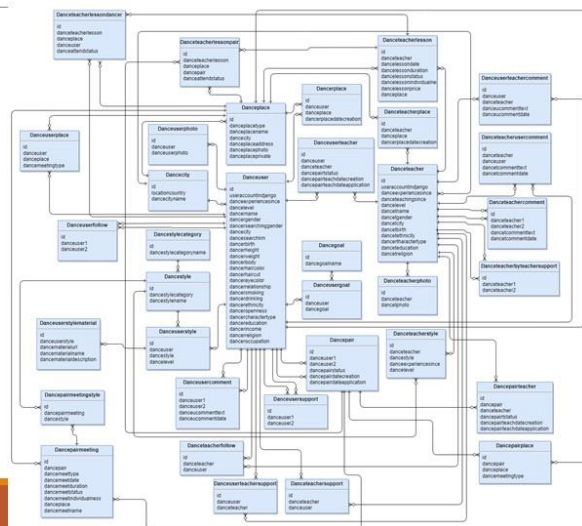
Порівняння засобів для розробки програми організації індивідуальних занять танцями

Критерій	Laravel	Django
Мова програмування	PHP	Python з простішим синтаксисом
Продуктивність	Стандартна	За релевантного використання засобів краща
Підтримка баз даних	Ефективна	Ефективна
Архітектурне рішення	Model-View-Controller	Model-View-Template
Масштабованість	Високий рівень	Вищий рівень за рахунок можливостей мови програмування
Безпека роботи і даних	Для відповідності потребує довгого налаштування	Краща за стандартних засобів
Засоби тестування	Зручні	Зручні

7

Рисунок В.7 – Порівняння засобів для розробки програми організації індивідуальних занять танцями

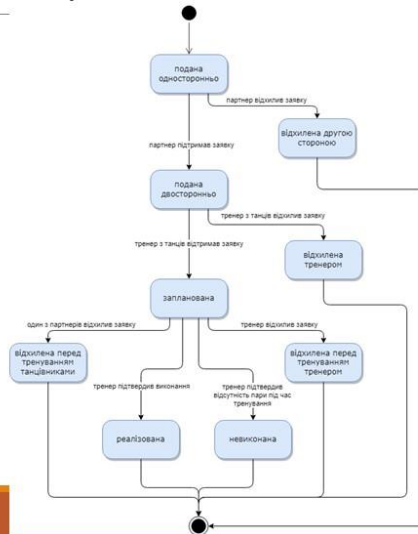
Схема бази даних програмного забезпечення організації індивідуальних занять танцями



8

Рисунок В.8 – Схема бази даних програмного забезпечення організації індивідуальних занять танцями

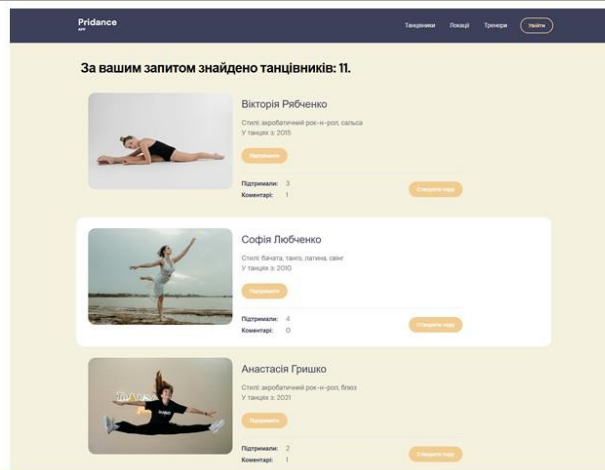
Діаграма станів заявки на відвідування індивідуальних занять з танців тренера танцювальною парою



9

Рисунок В.9 – Діаграма станів заявки на відвідування індивідуальних занять з танців тренера танцювальною парою

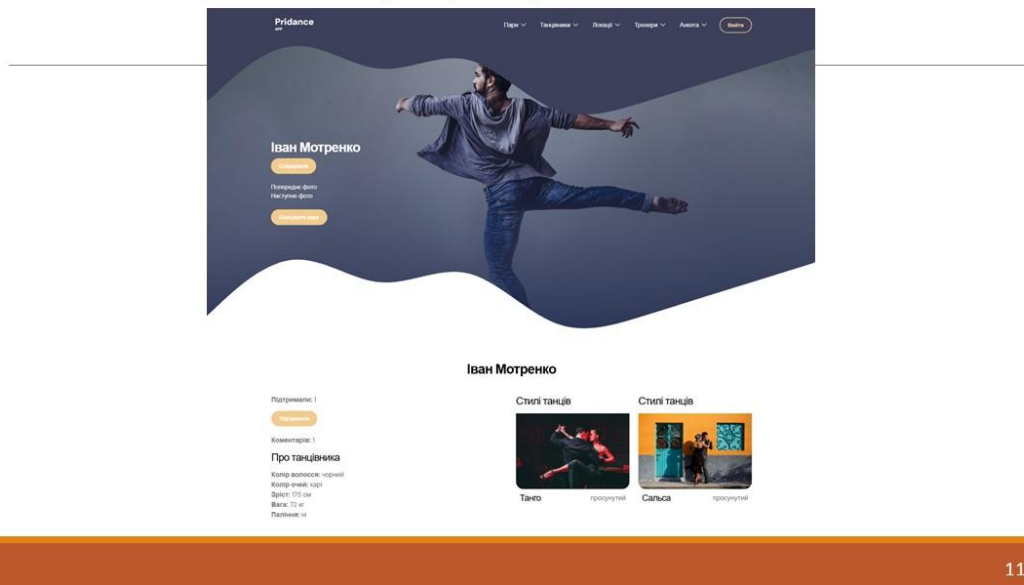
Результати пошуку партнерів для занять танцями у програмі



10

Рисунок В.10 – Результати пошуку партнерів для занять танцями у програмі

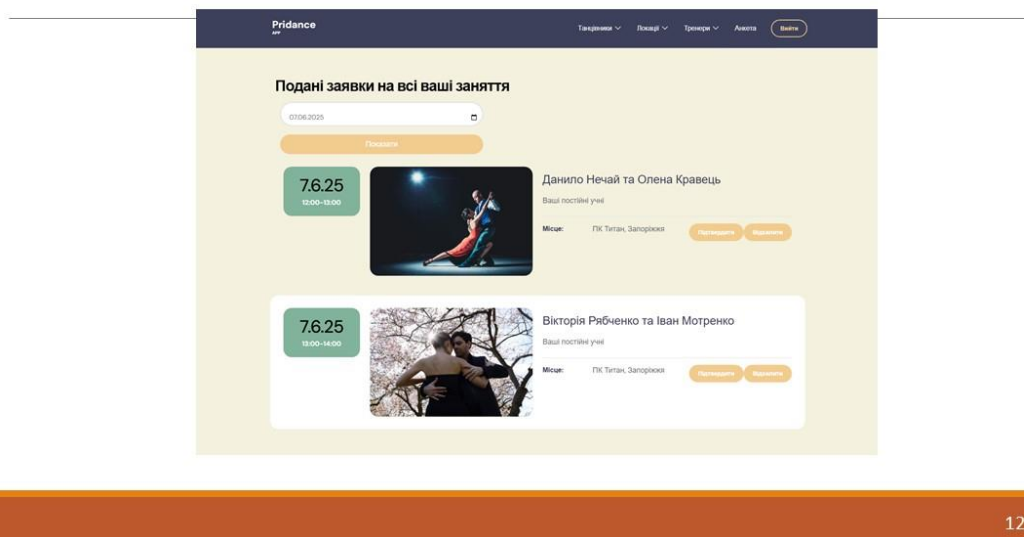
Анкета танцівника у програмі



11

Рисунок В.11 – Анкета танцівника у програмі

Керування поданими заявками на заняття з танців тренера



12

Рисунок В.12 – Керування поданими заявками на заняття з танців тренера

Тестування програми організації індивідуальних занять танцями

Функція	Неавторизований користувач	Танцівник	Тренер з танців	Функція	Танцівник	Тренер з танців
Визначення власного опису смаків та досвіду в танцях	+			Скасування визначення слідування за танцівником	+	
Розміщення доступного власного варіанту локації для занять танцями	+		+	Скасування доступного власного варіанту локації для занять танцями	+	
Пошук партнерів для занять танцями	+	+	+	Скасування власної заявки стосовно конкретного індивідуального заняття танцями у тренера	+	
Відтворення анкети танцівника	+	+	+	Редагування запису про матеріал за танцювальним досвідом	+	
Пошук локацій для занять танцями	+	+	+	Вилучення запису про матеріал за танцювальним досвідом	+	
Відтворення доступних варіантів локацій для занять танцями у танцівника	+	+	+	Скасування закріплення тренера для індивідуальних занять танцями пари	+	+
Відтворення доступних варіантів локацій для занять танцями у тренера з танців	+	+	+	Скасування закріплення тренера для індивідуальних занять танцями танцівника	+	+
Відтворення сторінки тренера для занять танцями	+	+	+	Подання заявки на створення пари для індивідуальних занять танцями	+	
Відтворення сторінки тренера для занять танцями у тренера	+	+	+	Відтворення поданих заявок на створення пари для індивідуальних занять танцями	+	
Пошук тренерів для занять танцями	+	+	+	Відтворення сторінок тренера для індивідуальних занять танцями пари	+	
Відтворення коментарів за тренером з танців	+	+	+	Закріплення тренера для індивідуальних занять танцями пари	+	
Відтворення коментарів за танцівником	+	+	+	Закріплення локації для індивідуальних занять танцями пари	+	
Авторизація	+	+	+	Закріплення локації для індивідуальних занять танцями танцівника	+	
Ресурси	+	+	+	Закріплення локації для індивідуальних занять танцями танцівника	+	
Відтворення графіку індивідуальних занять танцями пари		+		Підтримування тренера з танців	+	+
Відтворення даних підтриманих танцівників користувачем		+		Відтворення даних тренерів, за якими слідує користувач	+	
Відтворення даних підтриманих тренерів користувачем		+		Відтворення даних танцівників, за якими слідує користувач	+	
Коментування анкети танцівника		+	+	Відтворення доступних варіантів індивідуальних занять танцями у власних тренерів	+	
Відтворення даних прокоментованих танцівників		+	+	Планування індивідуального заняття танцями пари	+	
Коментування сторінки тренера з танців		+	+	Скасування визначення слідування за тренером з танців	+	
Відтворення даних прокоментованих тренерів		+	+	Керування поданою заявкою стосовно конкретного індивідуального заняття танцями цього тренера		+
Визначення слідування за танцівником		+		Відтворення графіку запланованих індивідуальних занять танцями цього тренера		+
Визначення слідування за тренером з танців		+		Відтворення заявок за індивідуальними заняттями танцями цього тренера		+
						13

Рисунок В.13 – Тестування програми організації індивідуальних занять танцями

Висновки

Мета виконаної роботи була в тому, щоб розробити програмне забезпечення організації індивідуальних занять танцями для підтримки супутнього процесу на його основних етапах. На основі виконання кожного окремого завдання було досягнуто мету роботи.

Отримані результати включають всі етапи виконання роботи, враховуючи визначення концепції програми організації індивідуальних занять танцями на основі виявлення відсутності готового рішення, встановлення вимог до програми, структури бази даних, розроблення всіх програмних одиниць та об'єднання їх у єдиний вебзастосунок, успішне тестування створеного вебзастосунку.

Розроблена як підсумок програма призначена для забезпечення підтримки занять танцями на етапах пошуку партнерів для створення пари, підбору тренера для пари або окремого танцівника, підбору локацій для таких занять, комплексного вирішення таких проблем, планування самих індивідуальних занять танцями.

Рисунок В.14 – Висновки