

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

бакалавр

(ступінь вищої освіти)

на тему ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ПОШУКУ

ЗНАЙОМИХ ДЛЯ СПІЛЬНОЇ РИБОЛОВЛІ

SOFTWARE TO FIND MATES FOR JOINT FISHING

Виконав(ла): студент(ка) 4 курсу, групи КНТ-151

Спеціальності 121 Інженерія програмного

(код і найменування спеціальності)

забезпечення

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

ПІЧКО Г.Г.

(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

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

ПІЧКА Германа Геннадійовича
 (ПРІЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Програмне забезпечення пошуку знайомих для спільної риболовлі. Software to Find Mates for Joint Fishing

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)

(підпис) Герман ПІЧКО
(Ім'я ПРИЗВИЩЕ)

Керівник проєкту (роботи)

(підпис) Валерій ЛЬОВКІН
(Ім'я ПРИЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 100 с., 3 табл., 28 рис., 3 дод., 12 джерел.

РИБОЛОВЛЯ, РИБАЛКА, АКТИВНІСТЬ, ЗНАЙОМСТВО, ВЕБЗАСТОСУНОК.

Об'єкт роботи – процес пошуку знайомих для спільної риболовлі. Предмет роботи – програмне забезпечення пошуку знайомих для спільної риболовлі. Мета роботи – розробити програмне забезпечення пошуку знайомих для спільної риболовлі на основі інформування про власний досвід риболовлі.

Матеріали, методи та технічні засоби: для роботи з базою даних – система управління PostgreSQL, для створення вебзастосунку – мова Python, фреймворк Django.

Результати. Розроблена програма пошуку знайомих для спільної риболовлі враховує сучасні вимоги до такого типу вебзастосунків, використовує сучасні технології, сприяючи універсальності реалізації функціональних можливостей та враховуючи ці вимоги під час проєктування.

Висновки. Застосування розробленої програми для пошуку знайомих для спільної риболовлі базується не тільки на створенні відповідного пошуку за параметрами, але і на перегляді того досвіду, який потенційні знайомі раніше мали під час риболовлі, тих підходів до риболовлі, яким віддає переваги рибалка. Для цього програма надає ряд спрямованих інструментів, що полегшують ці процеси та сприяють прийняттю рішень, які можуть призвести до позитивного досвіду спільної риболовлі.

Галузь використання. Створення платформи для комунікації між рибалками та планування спільних активностей з забезпеченням розвитку спільноти рибалок, рибальських клубів.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 100 pages, 3 tables, 28 figures, 3 appendixes, 12 sources.

FISHING, FISHERMAN, ACTIVITY, ACQUAINTANCE, WEB APPLICATION.

The object of the work is the process of finding friends for joint fishing. The subject of the work is software for finding mates for joint fishing. The purpose of the work is to develop software for finding mates for joint fishing based on information about one's own fishing experience.

Materials, methods and tools. The tools include the PostgreSQL management system for working with the database, the Python language and the Django framework for creating a web application.

Results. The developed program for finding mates for joint fishing takes into account modern requirements for this type of web applications, uses modern technologies, contributing to the universal implementation of functional capabilities and taking these requirements into account during design.

Conclusions. The application of the developed program for finding acquaintances for joint fishing is based not only on creating an appropriate search by parameters, but also on reviewing the experience that potential mates had previously while fishing, those approaches to fishing that the fisherman prefers. To this end, the program provides a number of targeted tools that facilitate these processes and contribute to making decisions that can lead to a positive experience of joint fishing.

Field of use. Creating a platform for communication between fishermen and planning joint activities to ensure the development of a community of fishermen, fishing clubs.

ЗМІСТ

	С.
Перелік скорочень та умовних познач.....	8
Вступ.....	9
1 Аналіз предметної області.....	10
1.1 Огляд можливостей існуючих програм пошуку знайомих для спільної риболовлі	10
1.2 Виділення особливостей програми пошуку знайомих для спільної риболовлі, що розробляється, та порівняння з аналогами	15
1.3 Висновки за розділом 1	16
2 Проєктування програми.....	17
2.1 Визначення функціональних вимог користувачів до програми пошуку знайомих для спільної риболовлі.....	17
2.2 Визначення засобів для проєктування і розробки програми пошуку знайомих для спільної риболовлі.....	21
2.3 Структура бази даних програми пошуку знайомих для спільної риболовлі	23
2.4 Висновки за розділом 2	33
3 Розробка програми	34
3.1 Порядок зміни станів основної сутності програми	34
3.2 Реалізація основних класів програми пошуку знайомих для спільної риболовлі	35
3.3 Реалізація основних програмних функцій для пошуку знайомих для спільної риболовлі	44
3.4 Висновки за розділом 3	50
4 Експлуатація та тестування програми	51
4.1 Призначення програми пошуку знайомих для спільної риболовлі	51
4.2 Умови виконання програми пошуку знайомих для спільної риболовлі.....	51
4.3 Експлуатація програми пошуку знайомих для спільної риболовлі.....	51

4.4 Тестування програми пошуку знайомих для спільної риболовлі	56
4.5 Висновки за розділом 4	59
Висновки	60
Перелік джерел посилання	61
Додаток А Технічне завдання	63
Додаток Б Текст програми	68
Додаток В Слайди презентації	93

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

БД — база даних;

НЗ — не застосовується.

ВСТУП

Риболовля є одним з відомих активних хобі для багатьох людей. Непрямо на це вказує зокрема та статистика, що розмір ринку рибальських товарів у Сполучених Штатах Америки в 2023 році оцінювався в 8,78 мільярда доларів [1], що вказує на величезний обсяг і активність.

Спортивна риболовля привертає велику увагу зокрема і в Україні навіть під час війни [2]-[4]. Проводиться велика кількість заходів зі спортивної риболовлі. Створюється велика кількість клубів для риболовлі навколо окремих водойм або місць біля цих водойм. У той же час всі ці спільноти існують достатньо відокремлено. Комунікація пов'язана з персональними контактами, які мають налагоджуватись. Тому логічним є створення програмного засобу, який би підтримував налагодження таких контактів. Для цього розробка повинна надавати ряд спрямованих інструментів, що полегшують ці процеси та сприяють прийняттю рішень, які можуть призвести до позитивного досвіду спільної риболовлі.

Врахуванням цього попереднього огляду стану проблеми став вибір в якості мети роботи розробки програмного забезпечення пошуку знайомих для спільної риболовлі саме на основі інформування про власний досвід риболовлі. Дана мета має бути досягнута в роботі через виконання пов'язаних завдань:

- аналізу проблеми пошуку знайомих для спільної риболовлі;
- проєктування програми пошуку знайомих для спільної риболовлі;
- розробки програми пошуку знайомих для спільної риболовлі.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд можливостей існуючих програм пошуку знайомих для спільної риболовлі

Рибальська спільнота в Україні є доволі великою, однак підтримка на рівні відповідних програмних рішень потребує вивчення. Нерідко об'єднання рибалок здійснюється навколо магазинів риболовлі, відповідно вони є об'єднуючим фактором, адже їх власники нерідко займаються створенням різних активностей, просуваючи таким чином використання засобів, які є у них у продажі, для риболовлі. У такому випадку створюються сторінки у соціальній мережі Facebook для кожної такої активності (рис. 1.1), таким чином запрошуючи учасників [5]. Відповідно організатори мають сторінки у соціальній мережі Facebook, а на них рекламують проведення відповідних подій.

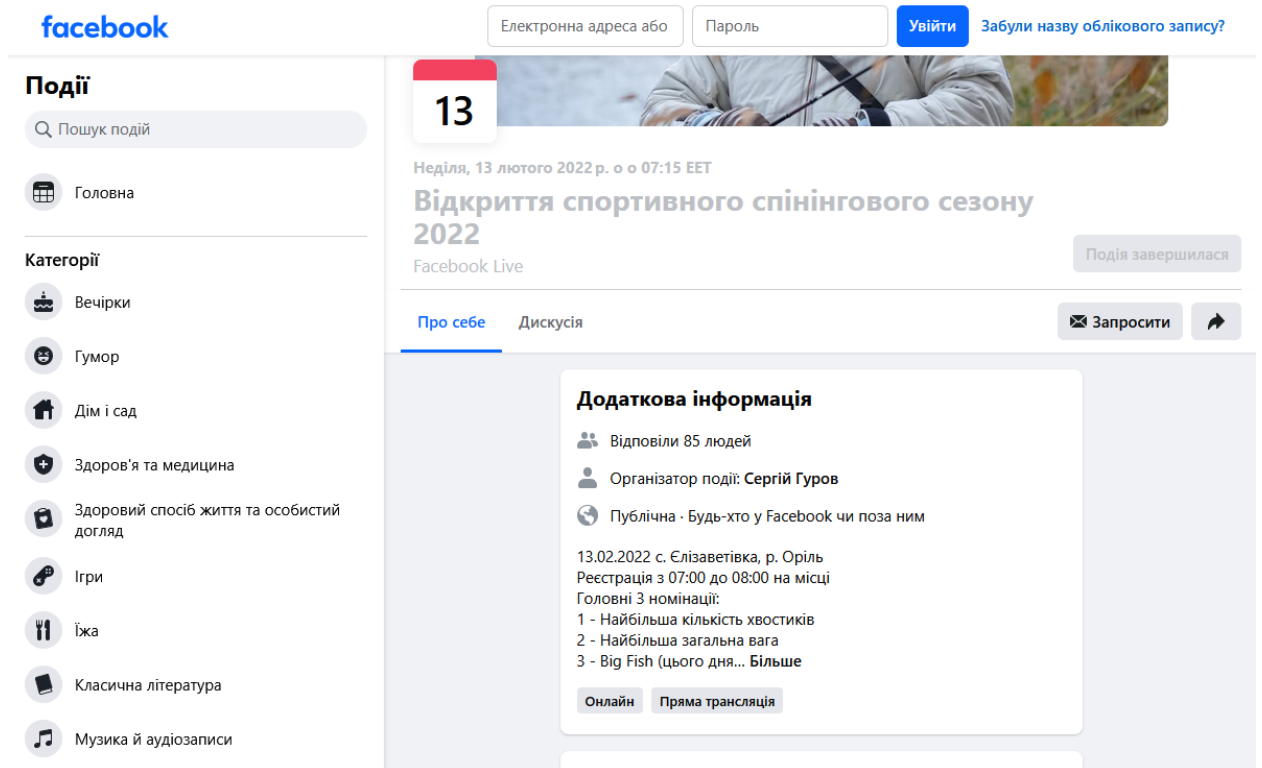


Рисунок 1.1 – Сторінка події зі спортивної риболовлі у мережі Facebook

Недоліком такого підходу є те, що взаємодія відбувається централізовано. Тобто є зазвичай загальний організатор, а він вже надає інформацію про ті події, які вважає за потрібне. Відповідно необхідно слідкувати за сторінкою організатора. Зазвичай же інформація про самі події з риболовлі неактивно заповнюється, відповідно отримання інформації для тих рибалок, які не взяли участь у події, непоширене.

Іншим напрямком об'єднання рибалок є створення місць для риболовлі, фактично кожне з яких представляє себе як певний клуб рибалок. Такі клуби рибалок зазвичай мають окремі вебсайти, де представлено інформацію про місця для риболовлі, а також про події, які безпосередньо в цих подіях беруть участь. Наприклад, це вебсайт WinFish (рис. 1.2) [6].

The screenshot shows the WinFish website header with the logo and navigation links: Послуги, Про водойму, Правила, Ціни, Події, Блог, Контакти, Онлайн-магазин. A button 'Забронювати сектор' is visible. The main content area is titled 'Ціни' (Prices) and contains three boxes for different rental options:

Оренда сектору з будиночком	Оренда сектору на добу	Оренда сектору на день
Будиночок на 4 особи - відмінний варіант для риболовлі та відпочинку	Риболовля протягом усієї доби, з 12:00 до 12:00 наступного дня	Риболовля протягом світлої пори доби, з 12:00 до 22:00
1 500 грн.	800 грн.	600 грн.

Below the pricing section is a heading 'Додаткові послуги для вашого комфорту' (Additional services for your comfort) followed by a list of services and their prices:

🛖 Оренда альтанки (велика): 1 200 грн./доба	🛖 Оренда альтанки (мала): 700 грн./доба
🏕 Оренда намету (великий): 400 грн./доба	🏕 Оренда намету (малий): 300 грн./доба
🪑 Оренда крісла (велике): 200 грн./доба	🪑 Оренда крісла (мале): 100 грн./доба
🪑 Оренда розкладушки: 300 грн./доба	🪑 Оренда підсака: 200 грн./доба
🪑 Дрова для вогнища: 200 грн./мішок	🧼 Антисептик для риби: 200 грн.

Рисунок 1.2 – Сторінка вебсайту WinFish

Фактично WinFish представляє дані про водойму, що знаходиться у Тернопільській області. Вебсайт містить розділи з інформацією про водойму, правила поведінки під час риболовлі на водоймі, вартість наявних послуг, події, які проводяться на водоймі (рис. 1.3), блог з новинами, посилання на онлайн-магазин. За кожною новиною про риболовлю в блозі представлена фотографія та коротка новина. Коментарі при цьому не можна створити,

реакції надати також. Це призводить до звуження використання інформації. Її неможливо практично просувати, ділитися власним досвідом користувачі також не можуть. У підсумку це негативно відображається на взаємодії з вебсайтом. Підтвердженням цього твердження є те, що останні новини у блозі датовані осінню 2023 року. Тобто нерідко такі ініціативи створюються, але окрім того жорстко залежать від розвитку відповідного місця для риболовлі. Якщо ж одні й ті самі рибалки беруть участь в риболовлі в інших місцях, то отримати таку інформацію через вебсайт неможливо.



Рисунок 1.3 – Сторінка події з вебсайту WinFish

Додатковим прикладом вебсайту водойми є вебсайт Selfish Club (рис. 1.4), розташований на березі Дніпра [7]. Вебсайт має дуже обмежені функції, надаючи інформацію про варіанти проживання, меню в ресторані, вартість риболовлі, заходи, які можна організувати та окрему сторінку зі всіма тарифами. Отримати інформацію про якийсь досвід окремих рибалок через вебзастосунок неможливо. Відповідно неможливо побачити практику діяльності спільноти, а тому і знайти однодумців у підсумку також.

Хоча вносить інформацію про звіти фактично тільки адміністратор, тобто спільнота не підтримується, проте вебзастосунок має форму авторизації користувачів, а також реєстрації. Мала кількість користувачів відображається в тому, що деякі розділи застарілі (рис. 1.5), хоча інші свідчать про продовження підтримки застосунку і далі (рис. 1.6).



Рисунок 1.6 – Сторінка з інформацією про змагання вебсайту клубу рибалок «Відпочинок»

Проте клуб рибалок «Відпочинок» також має фізичне розташування, тому фактично представляє водойму для риболовлі. Вебсайт вміщує розділи з інформацією про вартість надаваних послуг, контакти, звіти про активності, новини, акції, партнерів.

1.2 Виділення особливостей програми пошуку знайомих для спільної риболовлі, що розробляється, та порівняння з аналогами

Базові можливості розглянутих у попередньому підрозділі програм були порівняні з розробкою (табл. 1.1), що дозволило встановити, що аналоги тільки опосередковано можуть застосовуватися для пошуку знайомих для спільної риболовлі.

Таблиця 1.1 – Результати порівняння програм, застосовуваних для пошуку знайомих для спільної риболовлі

Критерій	WinFish	Відпочинок	FishMateSoft
Звітування про активності	Так, без реакцій	Так, з коментарями	З наданням інформації про використанні засоби для риболовлі, з реакціями, коментарями
Установлення знайомств серед рибалок через сам вебзастосунок	Ні	Ні	Так
Пошук серед користувачів знайомих для спільної риболовлі	Ні	Ні	Так
Визначення запланованих активностей користувачами	Ні, тільки адміністратор	Ні, тільки адміністратор	Так

Виконувана в цій роботі розробка FishMateSoft повинна дозволити реалізувати і устанавлення знайомств серед рибалок через сам вебзастосунок, і пошук серед користувачів знайомих для спільної риболовлі, а також повинна сприяти утворенню цих знайомств та отриманню додаткової інформації про риболовлю на основі визначення запланованих активностей користувачами та звітування про активності з наданням інформації про використані засоби для риболовлі.

1.3 Висновки за розділом 1

Огляд можливостей існуючих програм комунікації всередині спільнот рибалок продемонстрував, що безпосередніх засобів пошуку знайомих для спільної риболовлі вони не мають, а встановлення контактів відбувається у загальних соціальних мережах. У той же час такі соціальні мережі не враховують особливостей риболовлі, а тому не сприяють додатково підтримці даного процесу, роблячи його доволі фрагментованим. Саме ці недоліки мають бути усунуті в розробці, яка має бути результатом виконання цієї роботи. Виконувана розробка повинна дозволити реалізувати і устанавлення знайомств серед рибалок через сам вебзастосунок, і пошук серед користувачів знайомих для спільної риболовлі, а також повинна сприяти утворенню цих знайомств та отриманню додаткової інформації про риболовлю на основі визначення запланованих активностей користувачами та звітування про активності.

2 ПРОЄКТУВАННЯ ПРОГРАМИ

2.1 Визначення функціональних вимог користувачів до програми пошуку знайомих для спільної риболовлі

Фактично всі користувачі програми пошуку знайомих для спільної риболовлі можуть бути авторизованими, або неавторизованими. Авторизовані користувачі представляються роллю «рибалка після авторизації».

У програмі неавторизовані користувачі мають доступ до таких сценаріїв (рис. 2.1):

- перегляд активностей, пов'язаних з риболовлею, запланованих рибалкою;
- пошук серед активностей, пов'язаних з риболовлею;
- пошук знайомих для спільної риболовлі;
- перегляд інформації про активність, пов'язану з риболовлею;
- перегляд профільних даних рибалки;
- перегляд звітувань рибалки за досвідом активностей, пов'язаних з риболовлею;
- перегляд звітувань за досвідом активностей, пов'язаних з риболовлею, в яких застосовано певний засіб;
- авторизація рибалок;
- пошук рибальських засобів;
- перегляд інформації про рибальський засіб;
- перегляд інформації про звітування рибалки за активністю, пов'язаною з риболовлею.

Альтернативний сценарій для таких користувачів у програмі пошуку знайомих для спільної риболовлі передбачений тільки один – реєстрація рибалок, що є альтернативою для авторизації, коли користувач ще не отримав прав зареєстрованого рибалки у вебзастосунку.

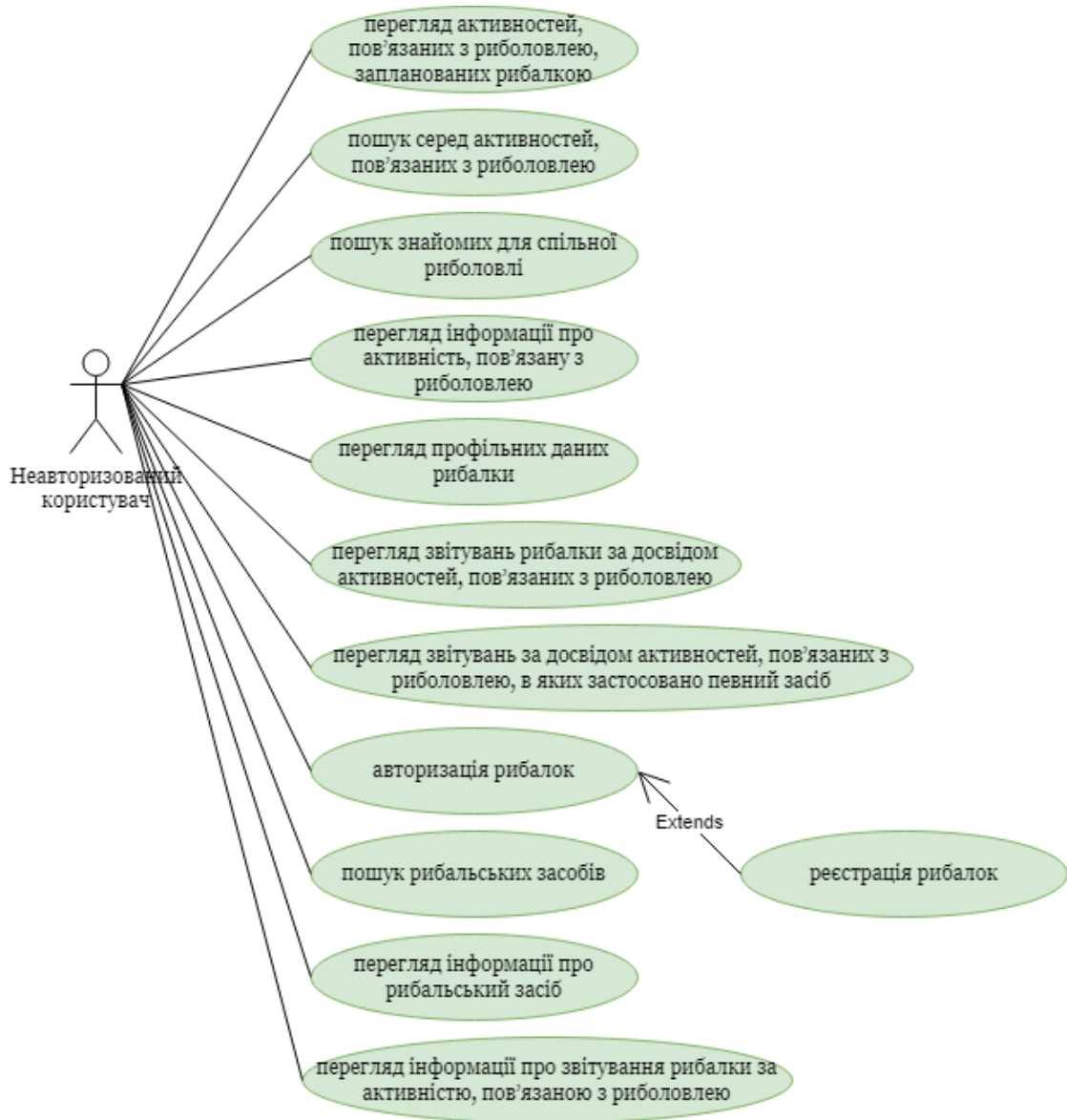


Рисунок 2.1 – Діаграма варіантів використання програми пошуку знайомих для спільної риболовлі неавторизованим користувачем

Рибалки після авторизації в програмі мають доступ до таких сценаріїв:

- планування активності, пов'язаної з риболовлю;
- додавання до активності, пов'язаної з риболовлю, в якості учасника;
- перегляд активностей, пов'язаних з риболовлю, в яких планує взяти участь рибалка;
- внесення профільних даних рибалки;
- виставлення власної сторінки рибалки для спільної рибалки;
- подання заявки на знайомство для спільної риболовлі;

- підтвердження заявки на знайомство для спільної риболовлі;
- перегляд знайомих для спільної риболовлі;
- перегляд заявок на знайомство для спільної риболовлі, що подані або отримані;
- звітування за власним досвідом активності, пов'язаної з риболовлею;
- оцінювання активності, пов'язаної з риболовлею;
- коментування звітування за досвідом активності, пов'язаної з риболовлею;
- перегляд запланованих активностей, пов'язаних з риболовлею, які влаштовуються знайомими;
- перегляд запланованих активностей, пов'язаних з риболовлею, в яких планують взяти участь знайомі;
- перегляд звітувань знайомих рибалок за досвідом активностей, пов'язаних з риболовлею;
- реагування на звітування за досвідом активності, пов'язаної з риболовлею;
- перегляд звітувань за досвідом активностей, пов'язаних з риболовлею, що отримали власні реакції (рис. 2.2).

Альтернативні сценарії для рибалки після авторизації:

- редагування активності, пов'язаної з риболовлею, є альтернативою для планування активності, пов'язаної з риболовлею, коли потрібно користувачу-рибалці, що створив цю активність, внести в неї зміни через зміну відповідних параметрів;
- скасування участі в активності, пов'язаній з риболовлею, є альтернативою для додавання до активності, пов'язаної з риболовлею, в якості учасника, коли участь у активності, пов'язаній з риболовлею, про яку було заявлено, довелось скасувати;
- редагування профільних даних рибалки є альтернативою для внесення профільних даних рибалки, коли профільні дані рибалки є застарілими.

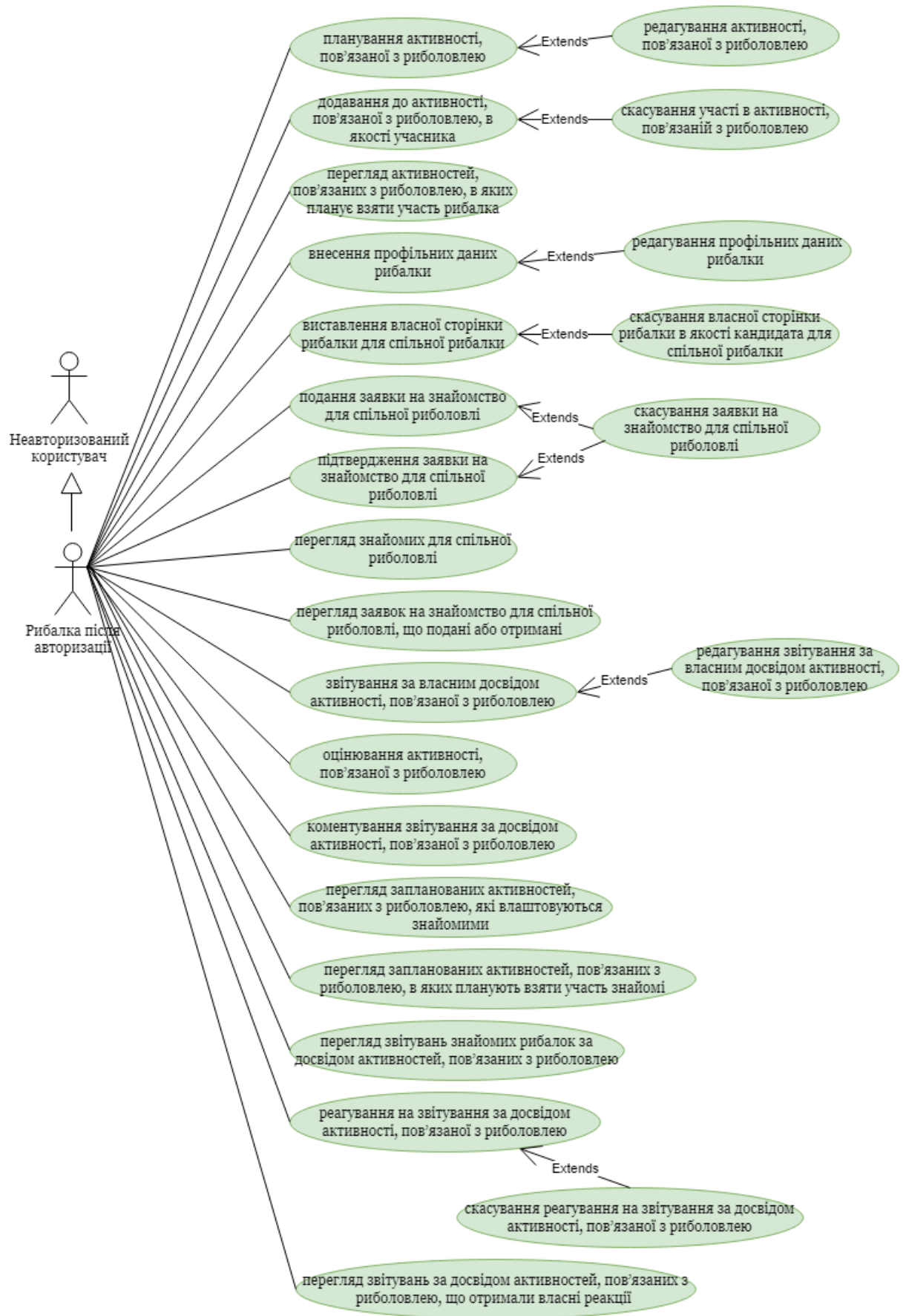


Рисунок 2.2 – Діаграма варіантів використання програми пошуку знайомих для спільної риболовлі рибалкою після авторизації

Також альтернативними сценаріями для рибалки після авторизації є наступні:

- скасування заявки на знайомство для спільної риболовлі є альтернативою для подання заявки на знайомство для спільної риболовлі, коли заявка була подана помилково або більше не відповідає бажанню рибалки, а також цей сценарій є альтернативою для сценарію підтвердження заявки на знайомство для спільної риболовлі, коли рибалка не бажає встановлювати знайомство з таким ініціатором;

- скасування власної сторінки рибалки в якості кандидата для спільної рибалки є альтернативою для виставлення власної сторінки рибалки для спільної рибалки, коли рибалка більше не має бажання встановлювати нові знайомства з іншими рибалками, відповідно необхідно скасувати відображення його сторінки в результатах пошуку;

- редагування звітування за власним досвідом активності, пов'язаної з риболовлею, є альтернативою для звітування за власним досвідом активності, пов'язаної з риболовлею, коли звітування рибалкою було реалізовано, проте містить неточності;

- скасування реагування на звітування за досвідом активності, пов'язаної з риболовлею, є альтернативою для реагування на звітування за досвідом активності, пов'язаної з риболовлею, коли рибалка після авторизації бажає змінити виставлену реакцію шляхом її прибирання, адже вона була виставлена помилково або більше не відповідає дійсності.

2.2 Визначення засобів для проєктування і розробки програми пошуку знайомих для спільної риболовлі

Визначення засобів для проєктування і розробки програми пошуку знайомих для спільної риболовлі виконувалось виходячи з того, що програма має бути реалізована вебзастосунком, адже за обмеження в часі саме таке рішення збільшує можливість потенційних звернень без додаткових вимог.

Оскільки мова програмування, що активно використовується для розробки зараз, маючи універсальні засоби для роботи в різних галузях, Python, має два основні фреймворки для веброзробки, Django [9]-[10] і Flask [11], то вибір відбувався саме серед них (табл. 2.1).

Таблиця 2.1 – Результати порівняння стосовно вибору засобів проєктування і розробки програми пошуку знайомих для спільної риболовлі

Критерій	Flask	Django
Призначення	Простіші вебзастосунки, для яких і час розробки зменшується	Складніші вебзастосунки, для яких і час розробки зменшується
Базова мова	Python	Python
Концепція	Мінімалістичний	Включені всі основні необхідні засоби
Безпека	Якісне налаштування протоколів безпеки потребує використання зовнішніх засобів	Одна з найкращих серед всіх фреймворків
Стандартне вирішення адміністративних завдань	Відсутнє	Адміністративна панель

Реалізовувати програму пошуку знайомих для спільної риболовлі вирішено за допомогою фреймворку Django, адже виходячи з наявних вимог до функцій, це складніше рішення, а Django проявляє себе у зменшенні часу розробки як раз для складніших вебзастосунків з активною взаємодією з базою даних. Також Django відзначається наявністю концепції включення всіх основних необхідних засобів розробки, не вимагаючи ці засоби обирати

окремо, підключати. Також робота з представленими функціями програми пошуку знайомих для спільної риболовлі вимагає часом адміністративного управління, проте додатково витрачати час на такі можливості недоцільно. Django надає стандартний засіб реалізації такої можливості.

2.3 Структура бази даних програми пошуку знайомих для спільної риболовлі

Структура бази даних (БД) програми пошуку знайомих для спільної риболовлі обрано безпосередньо на основі результатів аналізу предметної області визначених функціональних вимог до програми. Забезпечення роботи з БД реалізовано через систему управління БД PostgreSQL, враховуючи її надійність, продуктивність, безпеку, безкоштовний доступ.

Таблиця Fishinggoodcategory у БД визначає категорії засобів для риболовлі за стовпцями:

- id – ідентифікатор категорії засобів для риболовлі;
- fishinggood_category_name – назва категорії засобів для риболовлі.

Таблиця Fishinggoodsubcategory у БД визначає підкатегорії засобів для риболовлі за стовпцями:

- id – ідентифікатор підкатегорії засобів для риболовлі;
- fishinggoodcategory_id – категорія засобів для риболовлі, куди відноситься ця підкатегорія, визначена зовнішнім ключем за таблицею Fishinggoodcategory;
- fishinggood_subcategory_name – назва підкатегорії засобів для риболовлі.

Таблиця Fishinggoodmanufacturer у БД визначає виробників засобів для риболовлі за стовпцями:

- id – ідентифікатор виробника засобів для риболовлі;
- fishinggood_manufacturer_name – назва виробника засобів для риболовлі.

Таблиця Fishinggoodmodel у БД визначає моделі засобів для риболовлі за стовпцями:

- id – ідентифікатор моделі засобу для риболовлі;
- fishinggoodmanufacturer_id – виробник засобу для риболовлі цієї моделі, визначена зовнішнім ключем за таблицею Fishinggoodmanufacturer;
- fishinggoodsubcategory_id – підкатегорія засобів для риболовлі, куди слід віднести цю модель засобів для риболовлі, визначена зовнішнім ключем за таблицею Fishinggoodsubcategory;
- fishinggoodmodel_name – назва моделі засобу для риболовлі;
- fishinggood_basephoto – основна фотографія засобу для риболовлі в цій моделі;
- fishinggood_basedescription – опис засобу для риболовлі в цій моделі.

Таблиця Fishingactivitycategory у БД визначає категорії активностей, пов'язаних з риболовлею, за стовпцями:

- id – ідентифікатор категорії активностей, пов'язаних з риболовлею;
- fishing_activity_category_name – назва категорії активностей, пов'язаних з риболовлею.

Таблиця Fishingregion у БД визначає області країни, де може відбуватися риболовля, за стовпцями:

- id – ідентифікатор області країни, де може відбуватися риболовля;
- country – країна, в якій ця область, в якій може відбуватися риболовля, знаходиться;
- fishing_region_name – назва області країни, де може відбуватися риболовля.

Таблиця Fishingreservoir у БД визначає водойми для риболовлі за стовпцями:

- id – ідентифікатор водойми для риболовлі;
- fishing_reservoir_name – назва водойми для риболовлі;
- fishing_location_x – координата водойми для риболовлі за широтою;
- fishing_location_y – координата водойми для риболовлі за довготою;

– fishingregion_id – область країни, в якій знаходиться ця водойма для риболовлі, визначена зовнішнім ключем за таблицею Fishingregion;

– fishing_reservoir_type – тип водойми для риболовлі (озеро, річка тощо).

Таблиця Fishingreservoirfishsort у БД визначає види риби, що водяться в цій водоймі, за стовпцями:

– id – ідентифікатор виду риби, що водиться в певній водоймі для риболовлі;

– fishsort_id – вид риби, який водиться в цій водоймі для риболовлі, визначений зовнішнім ключем за таблицею Fishsort;

– fishingreservoir_id – водойма, в якій водиться цей вид риби, визначена зовнішнім ключем за таблицею Fishingreservoir;

– appearance_type – тип частоти появи цього виду риби в цій водоймі.

Таблиця Fishingactivity у БД визначає активності, пов'язані з риболовлею, які проводяться рибалками, за стовпцями:

– id – ідентифікатор активності, пов'язаної з риболовлею, що проводиться рибалкою;

– fishingactivitycategory_id – категорія активностей, пов'язаних з риболовлею, до якої належить ця активність, що проводиться рибалкою, визначена зовнішнім ключем за таблицею Fishingactivitycategory;

– fishing_planned_start_time – запланований час початку активності, пов'язаної з риболовлею, що проводиться рибалкою;

– fishing_planned_end_time – запланований час закінчення активності, пов'язаної з риболовлею, що проводиться рибалкою;

– initiative_fisher_id – рибалка, який визначив проведення цієї активності, пов'язаної з риболовлею, у програмі, визначений зовнішнім ключем за таблицею Fisher;

– fishingreservoir_id – водойма, на якій має відбуватися активність, пов'язана з риболовлею, визначена зовнішнім ключем за таблицею Fishingreservoir;

– fishing_facebook_url – адреса Facebook-сторінки з представленням інформації про активність, пов'язану з риболовлю;

– fishing_instagram_url – адреса Instagram-сторінки з представленням інформації про активність, пов'язану з риболовлю;

– fishing_web_url – адреса вебсторінки з представленням інформації про активність, пов'язану з риболовлю;

– fishing_activity_name – назва активності, пов'язаної з риболовлю;

– fishing_activity_info – опис проведення активності, пов'язаної з риболовлю;

– fishing_score_average – середня оцінка активності, пов'язаної з риболовлю.

Таблиця Fishingactivityphoto у БД визначає фотографії активностей, пов'язаних з риболовлю, за стовпцями:

– id – ідентифікатор фотографії зображення активності, пов'язаної з риболовлю;

– fishingactivity_id – активність, пов'язана з риболовлю, яку проілюстровано цією фотографією, визначена зовнішнім ключем за таблицею Fishingactivity;

– fishing_photo – фотографія, на якій показана ця активність, пов'язана з риболовлю.

Таблиця Fisher у БД визначає користувачів, які є рибалками, за стовпцями:

– id – ідентифікатор користувача, який є рибалкою;

– fisher_user_name – логін користувача, який є рибалкою;

– fisher_first_name – ім'я користувача, який є рибалкою;

– fisher_last_name – прізвище користувача, який є рибалкою;

– fisher_email – електронна пошта користувача, який є рибалкою;

– fisher_phonenumber – номер телефону користувача, який є рибалкою;

– fisher_facebook – адреса Facebook-сторінки користувача, який є рибалкою;

- fisher_instagram – адреса Instagram-сторінки користувача, який є рибалкою;
- fisher_password – пароль користувача, який є рибалкою;
- fisher_is_staff – користувач, який є рибалкою, має доступ до адміністративної панелі або ні;
- fisher_is_active – користувач, який є рибалкою, має активний акаунт або ні;
- fisher_is_superuser – користувач, який є рибалкою, є користувачем з правами адміністратора;
- fisher_last_login – час авторизації користувача, який є рибалкою, яка була виконана востаннє;
- fisher_date_joined – дата, коли користувач, який є рибалкою, зареєструвався;
- fisher_presentationbio – опис свого захоплення риболовлю користувача, який є рибалкою;
- fisher_formating – готовність користувача, який є рибалкою, брати участь у пошуку знайомих для спільної риболовлі;
- fishing_type – тип риболовлі, якому віддає перевагу користувач, який є рибалкою;
- fishing_winter – користувач, який є рибалкою, бере участь в зимовій риболовлі або ні.

Таблиця Fisherreservoir у БД визначає типи водойми для риболовлі, на яких конкретні рибалки беруть участь, за стовпцями:

- id – ідентифікатор участі рибалки в риболовлі на певному типі водойми для риболовлі;
- fisher_id – рибалка, який бере участь в риболовлі на цьому типі водойми, визначений зовнішнім ключем за таблицею Fisher;
- fisher_reservoir_type – тип водойми для риболовлі, на якій звик брати участь цей рибалка.

Таблиця Fishsort у БД визначає види риби за стовпцями:

- id – ідентифікатор виду риби;
- fish_sort_name – назва виду риби.

Таблиця Fisherfishsort у БД визначає види риби, які звик ловити певний рибалка, за стовпцями:

- id – ідентифікатор видів риби, які звикли ловити певні рибалки;
- fisher_id – рибалка, який звик ловити цей вид риби, визначений зовнішнім ключем за таблицею Fisher;
- fishsort_id – вид риби, який звик ловити цей рибалка, визначений зовнішнім ключем за таблицею Fishsort.

Таблиця Fisherphoto у БД визначає фотографії самих рибалок за стовпцями:

- id – ідентифікатор фотографії рибалки;
- fisher_id – рибалка, який зображений на фотографії, визначений зовнішнім ключем за таблицею Fisher;
- fisher_photo – фотографія, на якій рибалка зображений.

Таблиця Fishermate у БД визначає знайомства між рибалками для спільної риболовлі, які можуть бути тільки поданими заявками, відхиленими заявками, а можуть бути безпосередньо підтвердженими знайомствами, за стовпцями:

- id – ідентифікатор знайомства між рибалками для спільної риболовлі;
- first_fisher_id – рибалка, який є однією зі сторін знайомства для спільної риболовлі (ініціатор початкової заявки щодо знайомства), визначений зовнішнім ключем за таблицею Fisher;
- second_fisher_id – рибалка, який є другою зі сторін знайомства для спільної риболовлі, визначений зовнішнім ключем за таблицею Fisher;
- fisher_application_time – час, коли була створена заявка на знайомство для спільної рибалки;
- fisher_approve_time – час, коли заявка на знайомство для спільної рибалки була підтверджена;

– fisher_mate_state – стан заявки на знайомство для спільної рибалки: подана, підтверджена, скасована другою стороною, скасована першою стороною.

Таблиця Fishingactivityreport у БД визначає звітування рибалок за активностями, пов'язаними з риболовлюю, за стовпцями:

- id – ідентифікатор звітування рибалки за активністю, пов'язаної з риболовлюю;
- fishingactivity_id – активність, пов'язана з риболовлюю, за якою звітує рибалка, визначена зовнішнім ключем за таблицею Fishingactivity;
- fisher_id – рибалка, який звітує за цією активністю, пов'язаною з риболовлюю, визначений зовнішнім ключем за таблицею Fisher;
- fishing_report_creation_time – час, коли відбулось звітування за цією активністю, пов'язаною з риболовлюю, цього рибалки;
- fishing_activity_report_name – назва звіту за активністю, пов'язаною з риболовлюю;
- fishing_activity_report_info – текст звіту за активністю, пов'язаною з риболовлюю.

Таблиця Fishingactivityreportphoto у БД визначає фотографії, долучені до звіту за активністю, пов'язаною з риболовлюю, за стовпцями:

- id – ідентифікатор фотографії, долученої до звіту за активністю, пов'язаною з риболовлюю;
- fishingactivityreport_id – звіт рибалки за активністю, пов'язаною з риболовлюю, до якого прикріплено цю фотографію, визначений зовнішнім ключем за таблицею Fishingactivityreport;
- fishing_photo – фотографія, яка прикріплена до звіту рибалки за активністю, пов'язаною з риболовлюю;
- fishing_photo_comment – коментар самого рибалки до фотографії, яка прикріплена до звіту рибалки за активністю, пов'язаною з риболовлюю.

Таблиця Fishingactivityreportgood у БД визначає засоби для риболовлі, які були використані під час активності, пов'язаної з риболовлею, рибалки, який сформував звіт за результатами, за стовпцями:

- id – ідентифікатор запису про засоби для риболовлі, які були використані під час активності, пов'язаної з риболовлею, рибалки;
- fishingactivityreport_id – звіт рибалки за активністю, пов'язаною з риболовлею, до якого прикріплено цю фотографію, визначений зовнішнім ключем за таблицею Fishingactivityreport;
- fishinggoodmodel_id – модель засобу для риболовлі, яка була застосована рибалкою під час участі в активності, пов'язаної з риболовлею, за якою він сформував звіт, визначена зовнішнім ключем за таблицею Fishinggoodmodel;
- fishing_good_comment – коментар рибалки до моделі засобу для риболовлі, яка була застосована ним під час участі в активності, пов'язаної з риболовлею.

Таблиця Fishingactivityreportgoodphoto у БД визначає фотографії засобів для риболовлі, використаних рибалками під час їхньої участі в активностях, пов'язаних з риболовлею, за стовпцями:

- id – ідентифікатор фотографії засобів для риболовлі, використаних рибалками під час їхньої участі в активностях, пов'язаних з риболовлею;
- fishingactivityreportgood_id – модель засобу для риболовлі, що був використаний рибалкою під час проведення активності, пов'язаної з риболовлею, про участь в якій відбулось звітування, визначена зовнішнім ключем за таблицею Fishinggoodmodel;
- fishing_good_photo – фотографія, на якій продемонстровано використання рибалкою засобу для риболовлі під час проведення активності, пов'язаної з риболовлею, щодо участі в якій він звітував.

Таблиця Fishingactivityparticipant у БД визначає учасників активності, пов'язаної з риболовлею, за стовпцями:

– id – ідентифікатор участі рибалки в активності, пов'язаній з риболовлею;

– fishingactivity_id – активність, пов'язана з риболовлею, в якій взяв участь рибалка, визначена зовнішнім ключем за таблицею Fishingactivity;

– fished_id – рибалка, який взяв участь у цій активності, пов'язаній з риболовлею, визначений зовнішнім ключем за таблицею Fisher.

Таблиця Fishingactivityscore у БД визначає оцінювання активностей, пов'язаних з риболовлею, за стовпцями:

– id – ідентифікатор оцінювання активності, пов'язаної з риболовлею;

– fishingactivity_id – активність, пов'язана з риболовлею, оцінювання якої було виконано рибалкою, визначена зовнішнім ключем за таблицею Fishingactivity;

– fisher_id – рибалка, який оцінив цю активність, пов'язану з риболовлею, визначений зовнішнім ключем за таблицею Fisher;

– fishing_score – оцінка, яку рибалка встановив цій активності, пов'язаній з риболовлею;

– fishing_score_time – час, коли рибалка встановив оцінку цій активності, пов'язаній з риболовлею.

Таблиця Fishingactivityreportcomment у БД визначає коментарі до звітів рибалок про активності, пов'язані з риболовлею, за стовпцями:

– id – ідентифікатор коментаря рибалки до звіту за активністю, пов'язаною з риболовлею;

– fishingactivityreport_id – звіт за активністю, пов'язаною з риболовлею, за яким цей коментар створено, визначений зовнішнім ключем за таблицею Fishingactivityreport;

– fisher_id – рибалка, який створив коментар до звіту за активністю, пов'язаною з риболовлею, визначений зовнішнім ключем за таблицею Fisher;

– fishing_comment – коментар, який було створено рибалкою до звіту за активністю, пов'язаною з риболовлею;

– fishing_comment_time – час, коли було створено цей коментар
рибалки до звіту за активністю, пов’язаною з риболовлю (рис. 2.3).

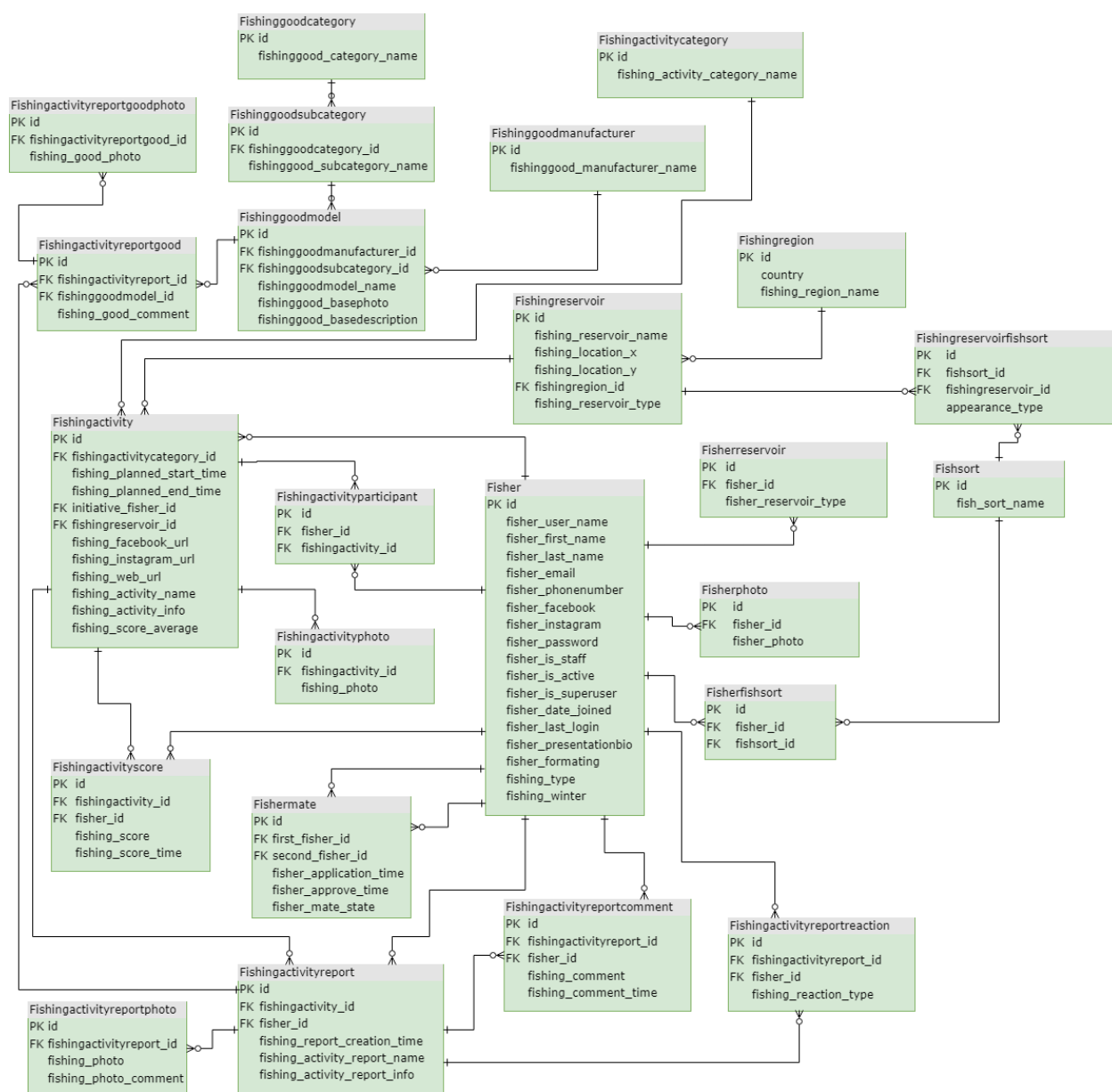


Рисунок 2.3 – Схема БД програми пошуку знайомих для спільної риболовлі

Таблиця Fishingactivityreportreaction у БД визначає реакції на звіти рибалок за активностями, пов'язаними з риболовлю, за стовпцями:

– id – ідентифікатор реакції на звіт рибалки за активністю, пов'язаною з риболовлюю;

– fishingactivityreport_id – звіт за активністю, пов’язаною з риболовлюю, реакцію на яку встановив рибалка, визначений зовнішнім ключем за таблицею Fishingactivityreport;

– fisher_id – рибалка, який таким чином прореагував на активність, пов’язану з риболовлюю, визначений зовнішнім ключем за таблицею Fisher;

– fishing_reaction_type – тип реакції, застосований рибалкою щодо звіту за активністю, пов’язаною з риболовлюю.

2.4 Висновки за розділом 2

Результатами розділу з проектування програми пошуку знайомих для спільної риболовлі є визначений набір сценаріїв роботи неавторизованого користувача та рибалки після авторизації, за якими побудовано роботу з програмою, визначена структура БД, обрані засоби для розробки, включаючи для набору технологій для роботи з базою даних – систему управління PostgreSQL, для створення вебзастосунку – мову Python, фреймворк Django.

3 РОЗРОБКА ПРОГРАМИ

3.1 Порядок зміни станів основної сутності програми

Знайомство для спільної риболовлі є головним вираженням мети роботи серед сутностей предметної області в програмі. Від станів цієї сутності залежить те, яким чином відбувається взаємодія між користувачами, які в програмі є рибалками після авторизації.

Спершу реалізується одним з рибалок після авторизації подача заявки на знайомство для спільної риболовлі. З цього моменту знайомство для спільної риболовлі переходить до стану «подана заявка» (рис. 3.1). Після цього може втрутитись або сам автор заявки, або її адресат. Автор заявки може не дочекатись її розгляду адресатом та відхилити її, тоді знайомство для спільної риболовлі переходить до стану «відхилена заявка». До цього ж стану знайомство переходить тоді, коли її відхилив адресат. Якщо ж адресат навпаки підтвердив заявку зі свого боку, то знайомство для спільної риболовлі переходить до стану «підтверджене знайомство». Після цього підтверджене знайомство може будь-якою зі сторін знайомства для спільної риболовлі бути скасованим. Тоді знайомство переходить до стану «відхилена заявка».

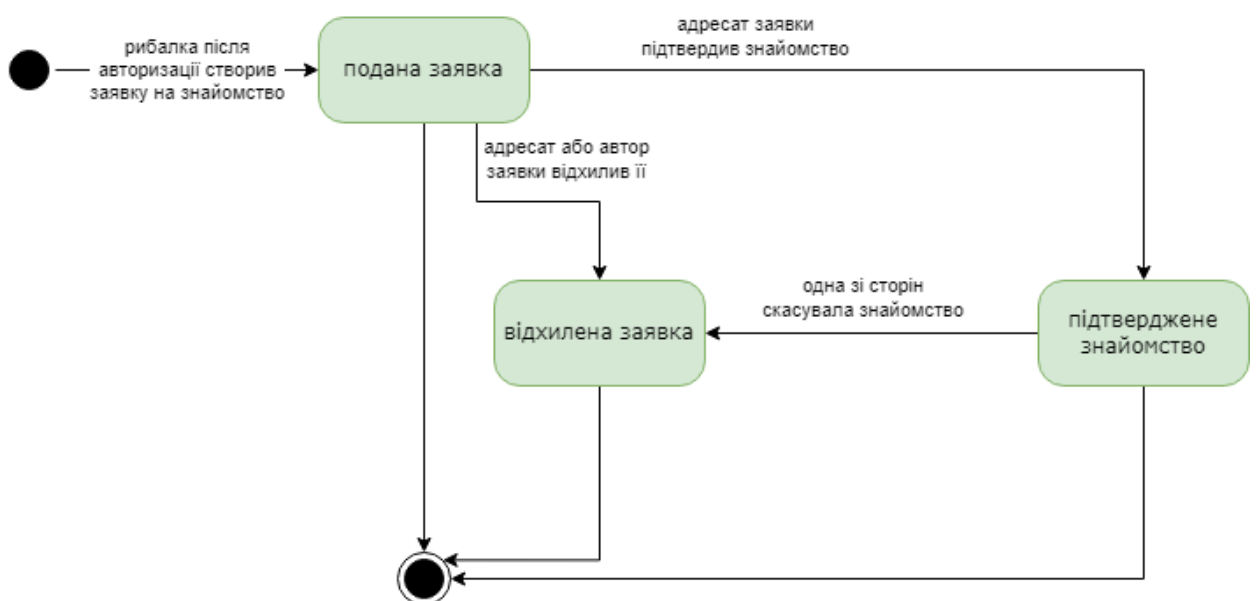


Рисунок 3.1 – Діаграма станів знайомства для спільної риболовлі

3.2 Реалізація основних класів програми пошуку знайомих для спільної риболовлі

Робота з основними сутностями предметної області в Django реалізується через створення моделей, які забезпечуються в коді відповідними класами моделей. На основі цього коду вже за виконання команд міграцій забезпечуються та виконуються дії зі створення аналогічної структури БД зокрема за використання системи PostgreSQL у цій роботі.

Клас `Fishinggoodcategory` серед моделей забезпечує модель категорії засобів для риболовлі. Клас `Fishinggoodcategory` містить тільки поле `fishinggood_category_name` для назви категорії засобів для риболовлі, маючи тип `CharField`.

Клас `Fishinggoodsubcategory` серед моделей забезпечує модель підкатегорії засобів для риболовлі.

Клас `Fishinggoodsubcategory` містить поле `fishinggoodcategory_id` для категорії засобів для риболовлі, куди відноситься ця підкатегорія, маючи тип `ForeignKey` зі зв'язком за моделлю `Fishinggoodcategory`.

Клас `Fishinggoodsubcategory` містить поле `fishinggood_subcategory_name` для назви підкатегорії засобів для риболовлі, маючи тип `CharField`.

Клас `Fishinggoodmanufacturer` серед моделей забезпечує модель виробників засобів для риболовлі. Клас `Fishinggoodmanufacturer` містить тільки поле `fishinggood_manufacturer_name` для назви виробника засобів для риболовлі, маючи тип `CharField`.

Клас `Fishinggoodmodel` серед моделей забезпечує модель засобів для риболовлі.

Клас `Fishinggoodmodel` містить поле `fishinggoodmanufacturer_id` для виробника засобу для риболовлі цієї моделі, маючи тип `ForeignKey` зі зв'язком за моделлю `Fishinggoodmanufacturer`.

Клас Fishinggoodmodel містить поле fishinggoodsubcategory_id для підкатегорії засобів для риболовлі, куди слід віднести цю модель засобів для риболовлі, маючи тип ForeignKey зі зв'язком за моделлю Fishinggoodsubcategory.

Клас Fishinggoodmodel містить поле fishinggoodmodel_name для назви моделі засобу для риболовлі, маючи тип CharField.

Клас Fishinggoodmodel містить поле fishinggood_basephoto для основної фотографії засобу для риболовлі в цій моделі, маючи тип ImageField.

Клас Fishinggoodmodel містить поле fishinggood_basedescription для опису засобу для риболовлі в цій моделі, маючи тип TextField.

Клас Fishingactivitycategory серед моделей забезпечує модель категорії активностей. Клас Fishingactivitycategory містить поле fishing_activity_category_name для назви категорії активностей, пов'язаних з риболовлю, маючи тип CharField.

Клас Fishingregion серед моделей забезпечує модель області країни, де може відбуватися риболовля.

Клас Fishingregion містить поле country для країни, в якій ця область, в якій може відбуватися риболовля, знаходиться, маючи тип CharField.

Клас Fishingregion містить поле fishing_region_name для назви області країни, де може відбуватися риболовля, маючи тип CharField.

Клас Fishingreservoir серед моделей забезпечує модель водойми для риболовлі.

Клас Fishingreservoir містить поле fishing_reservoir_name для назви водойми для риболовлі, маючи тип CharField.

Клас Fishingreservoir містить поле fishing_location_x для координати водойми для риболовлі за широтою, маючи тип DecimalField.

Клас Fishingreservoir містить поле fishing_location_y для координати водойми для риболовлі за довготою, маючи тип DecimalField.

Клас Fishingreservoir містить поле fishingregion_id для області країни, в якій знаходиться ця водойма для риболовлі, маючи тип ForeignKey зі зв'язком за моделлю Fishingregion.

Клас Fishingreservoir містить поле fishing_reservoir_type для типу водойми для риболовлі, маючи тип PositiveSmallIntegerField.

Клас Fishingreservoirfishsort серед моделей забезпечує модель видів риби, що водяться в цій водоймі.

Клас Fishingreservoirfishsort містить поле fishsort_id для виду риби, який водиться в цій водоймі для риболовлі, маючи тип ForeignKey зі зв'язком за моделлю Fishsort;

Клас Fishingreservoirfishsort містить поле fishingreservoir_id для водойми, в якій водиться цей вид риби, маючи тип ForeignKey зі зв'язком за моделлю Fishingreservoir.

Клас Fishingreservoirfishsort містить поле appearance_type для типу частоти появи цього виду риби в цій водоймі, маючи тип PositiveSmallIntegerField.

Клас Fishingactivity серед моделей забезпечує модель активностей, пов'язаних з риболовлею, які проводяться рибалками.

Клас Fishingactivity містить поле fishingactivitycategory_id для категорії активностей, пов'язаних з риболовлею, до якої належить ця активність, що проводиться рибалкою, маючи тип ForeignKey зі зв'язком за моделлю Fishingactivitycategory.

Клас Fishingactivity містить поле fishing_planned_start_time для запланованого часу початку активності, пов'язаної з риболовлею, що проводиться рибалкою, маючи тип DateTimeField.

Клас Fishingactivity містить поле fishing_planned_end_time для запланованого часу закінчення активності, пов'язаної з риболовлею, що проводиться рибалкою, маючи тип DateTimeField.

Клас Fishingactivity містить поле initiative_fisher_id для рибалки, який визначив проведення цієї активності, пов'язаної з риболовлею, у програмі, маючи тип ForeignKey зі зв'язком за моделлю Fisher.

Клас Fishingactivity містить поле fishingreservoir_id для водойми, на якій має відбуватися активність, пов'язана з риболовлею, маючи тип ForeignKey зі зв'язком за моделлю Fishingreservoir.

Клас Fishingactivity містить поле fishing_facebook_url для адреси Facebook-сторінки з представленням інформації про активність, пов'язану з риболовлею, маючи тип URLField.

Клас Fishingactivity містить поле fishing_instagram_url для адреси Instagram-сторінки з представленням інформації про активність, пов'язану з риболовлею, маючи тип URLField.

Клас Fishingactivity містить поле fishing_web_url для адреси вебсторінки з представленням інформації про активність, пов'язану з риболовлею, маючи тип URLField.

Клас Fishingactivity містить поле fishing_activity_name для назви активності, пов'язаної з риболовлею, маючи тип CharField.

Клас Fishingactivity містить поле fishing_activity_info для опису проведення активності, пов'язаної з риболовлею, маючи тип TextField.

Клас Fishingactivity містить поле fishing_score_average для середньої оцінки активності, пов'язаної з риболовлею, маючи тип DecimalField.

Клас Fishingactivity містить поле fishers для учасників активності, пов'язаної з риболовлею, маючи тип ManyToManyField зі зв'язком за моделлю Fisher.

Клас Fishingactivityphoto серед моделей забезпечує модель фотографії активностей, пов'язаних з риболовлею.

Клас Fishingactivityphoto містить поле fishingactivity_id для активності, пов'язаної з риболовлею, яку проілюстровано цією фотографією, маючи тип ForeignKey зі зв'язком за моделлю Fishingactivity.

Клас Fishingactivityphoto містить поле fishing_photo для фотографії, на якій показана ця активність, пов'язана з риболовлю, маючи тип ImageField.

Клас Fisher серед моделей забезпечує модель користувачів, які є рибалками.

Клас Fisher містить поле fisher_user_name для логіну користувача, який є рибалкою, маючи тип CharField.

Клас Fisher містить поле fisher_first_name для імені користувача, який є рибалкою, маючи тип CharField.

Клас Fisher містить поле fisher_last_name для прізвища користувача, який є рибалкою, маючи тип CharField.

Клас Fisher містить поле fisher_email для електронної пошти користувача, який є рибалкою, маючи тип EmailField.

Клас Fisher містить поле fisher_phonenumber для номеру телефону користувача, який є рибалкою, маючи тип CharField.

Клас Fisher містить поле fisher_facebook для адреси Facebook-сторінки користувача, який є рибалкою, маючи тип URLField.

Клас Fisher містить поле fisher_instagram для адреси Instagram-сторінки користувача, який є рибалкою, маючи тип URLField.

Клас Fisher містить поле fisher_password для паролю користувача, який є рибалкою, маючи тип CharField.

Клас Fisher містить поле fisher_is_staff для визначення користувача, який є рибалкою, що має доступ до адміністративної панелі, маючи тип BooleanField.

Клас Fisher містить поле fisher_is_active для визначення користувача, який є рибалкою, що має активний акаунт, маючи тип BooleanField.

Клас Fisher містить поле fisher_is_superuser для визначення користувача, який є рибалкою, що є користувачем з правами адміністратора, маючи тип BooleanField.

Клас Fisher містить поле `fisher_last_login` для часу авторизації користувача, який є рибалкою, яка була виконана востаннє, маючи тип `DateTimeField`.

Клас Fisher містить поле `fisher_date_joined` для дати, коли користувач, який є рибалкою, зареєструвався, маючи тип `DateTimeField`.

Клас Fisher містить поле `fisher_presentationbio` для опису свого захоплення риболовлю користувача, який є рибалкою, маючи тип `TextField`.

Клас Fisher містить поле `fisher_formating` для готовності користувача, який є рибалкою, брати участь у пошуку знайомих для спільної риболовлі, маючи тип `BooleanField`.

Клас Fisher містить поле `fishing_type` для типу риболовлі, якому віддає перевагу користувач, який є рибалкою, маючи тип `PositiveSmallIntegerField`.

Клас Fisher містить поле `fishing_winter` для визначення користувача, який є рибалкою, що бере участь в зимовій риболовлі, маючи тип `BooleanField`.

Клас Fisher містить поле `fishsorts` для видів риб, які звик ловити цей рибалка, маючи тип `ManyToManyField` зі зв'язком за моделлю `Fishsort`.

Клас `Fisherreservoir` серед моделей забезпечує модель типи водойми для риболовлі, на яких конкретні рибалки беруть участь.

Клас `Fisherreservoir` містить поле `fisher_id` для рибалки, який бере участь в риболовлі на цьому типі водойми, маючи тип `ForeignKey` зі зв'язком за моделлю `Fisher`.

Клас `Fisherreservoir` містить поле `fisher_reservoir_type` для типу водойми для риболовлі, на якій звик брати участь цей рибалка, маючи тип `PositiveSmallIntegerField`.

Клас `Fishsort` серед моделей забезпечує модель види риби. Клас `Fishsort` містить поле `fish_sort_name` для назви виду риби, маючи тип `CharField`.

Клас `Fisherphoto` серед моделей забезпечує модель фотографії самих рибалок.

Клас `Fisherphoto` містить поле `fisher_id` для рибалки, який зображений на фотографії, маючи тип `ForeignKey` зі зв'язком за моделлю `Fisher`.

Клас Fisherphoto містить поле fisher_photo для фотографії, на якій рибалка зображений, маючи тип ImageField.

Клас Fishermate серед моделей забезпечує модель знайомства між рибалками для спільної риболовлі, які можуть бути тільки поданими заявками, відхиленими заявками, а можуть бути безпосередньо підтвердженими знайомствами.

Клас Fishermate містить поле first_fisher_id для рибалки, який є однією зі сторін знайомства для спільної риболовлі (ініціатор початкової заявки щодо знайомства), маючи тип ForeignKey зі зв'язком за моделлю Fisher.

Клас Fishermate містить поле second_fisher_id для рибалки, який є другою зі сторін знайомства для спільної риболовлі, маючи тип ForeignKey зі зв'язком за моделлю Fisher.

Клас Fishermate містить поле fisher_application_time для часу, коли була створена заявка на знайомство для спільної рибалки, маючи тип DateTimeField.

Клас Fishermate містить поле fisher_approve_time для часу, коли заявка на знайомство для спільної рибалки була підтверджена, маючи тип DateTimeField.

Клас Fishermate містить поле fisher_mate_state для стану заявки на знайомство для спільної рибалки, маючи тип PositiveSmallIntegerField.

Клас Fishingactivityreport серед моделей забезпечує модель звітування рибалок за активностями, пов'язаними з риболовлею.

Клас Fishingactivityreport містить поле fishingactivity_id для активності, пов'язаної з риболовлею, за якою звітує рибалка, маючи тип ForeignKey зі зв'язком за моделлю Fishingactivity.

Клас Fishingactivityreport містить поле fisher_id для рибалки, який звітує за цією активністю, пов'язаною з риболовлею, маючи тип ForeignKey зі зв'язком за моделлю Fisher.

Клас Fishingactivityreport містить поле fishing_report_creation_time для часу, коли відбулось звітування за цією активністю, пов'язаною з риболовлюю, цього рибалки, маючи тип DateTimeField.

Клас Fishingactivityreport містить поле fishing_activity_report_name для назви звіту за активністю, пов'язаною з риболовлюю, маючи тип CharField.

Клас Fishingactivityreport містить поле fishing_activity_report_info для тексту звіту за активністю, пов'язаною з риболовлюю, маючи тип TextField.

Клас Fishingactivityreportphoto серед моделей забезпечує модель фотографії, долучені до звіту за активністю, пов'язаною з риболовлюю.

Клас Fishingactivityreportphoto містить поле fishingactivityreport_id для звіту рибалки за активністю, пов'язаною з риболовлюю, до якого прикріплено цю фотографію, маючи тип ForeignKey зі зв'язком за моделлю Fishingactivityreport.

Клас Fishingactivityreportphoto містить поле fishing_photo для фотографії, яка прикріплена до звіту рибалки за активністю, пов'язаною з риболовлюю, маючи тип ImageField.

Клас Fishingactivityreportphoto містить поле fishing_photo_comment для коментаря самого рибалки до фотографії, яка прикріплена до звіту рибалки за активністю, пов'язаною з риболовлюю, маючи тип CharField.

Клас Fishingactivityreportgood серед моделей забезпечує модель засоби для риболовлі, які були використані під час активності, пов'язаної з риболовлюю, рибалки, який сформував звіт за результатами.

Клас Fishingactivityreportgood містить поле fishingactivityreport_id для звіту рибалки за активністю, пов'язаною з риболовлюю, до якого прикріплено цю фотографію, маючи тип ForeignKey зі зв'язком за моделлю Fishingactivityreport.

Клас Fishingactivityreportgood містить поле fishinggoodmodel_id для моделі засобу для риболовлі, яка була застосована рибалкою під час участі в активності, пов'язаної з риболовлюю, за якою він сформував звіт, маючи тип ForeignKey зі зв'язком за моделлю Fishinggoodmodel.

Клас `Fishingactivityreportgood` містить поле `fishing_good_comment` для коментаря рибалки до моделі засобу для риболовлі, яка була застосована ним під час участі в активності, пов'язаної з риболовлею, маючи тип `CharField`.

Клас `Fishingactivityreportgoodphoto` серед моделей забезпечує модель фотографії засобів для риболовлі, використаних рибалками під час їхньої участі в активностях, пов'язаних з риболовлею.

Клас `Fishingactivityreportgoodphoto` містить поле `fishingactivityreportgood_id` для моделі засобу для риболовлі, що був використаний рибалкою під час проведення активності, пов'язаної з риболовлею, про участь в якій відбулось звітування, маючи тип `ForeignKey` зі зв'язком за моделлю `Fishinggoodmodel`.

Клас `Fishingactivityreportgoodphoto` містить поле `fishing_good_photo` для фотографії, на якій продемонстровано використання рибалкою засобу для риболовлі під час проведення активності, пов'язаної з риболовлею, щодо участі в якій він звітував, маючи тип `ImageField`.

Клас `Fishingactivityscore` серед моделей забезпечує модель оцінювання активностей, пов'язаних з риболовлею.

Клас `Fishingactivityscore` містить поле `fishingactivity_id` для активності, пов'язаної з риболовлею, оцінювання якої було виконано рибалкою, маючи тип `ForeignKey` зі зв'язком за моделлю `Fishingactivity`.

Клас `Fishingactivityscore` містить поле `fisher_id` для рибалки, який оцінив цю активність, пов'язану з риболовлею, маючи тип `ForeignKey` зі зв'язком за моделлю `Fisher`.

Клас `Fishingactivityscore` містить поле `fishing_score` для оцінки, яку рибалка встановив цій активності, пов'язаній з риболовлею, маючи тип `PositiveSmallIntegerField`.

Клас `Fishingactivityscore` містить поле `fishing_score_time` для часу, коли рибалка встановив оцінку цій активності, пов'язаній з риболовлею, маючи тип `DateTimeField`.

Клас `Fishingactivityreportcomment` серед моделей забезпечує модель коментарів до звітів рибалок про активності, пов'язані з риболовлею.

Клас `Fishingactivityreportcomment` містить поле `fishingactivityreport_id` для звіту за активністю, пов'язаною з риболовлею, за яким цей коментар створено, маючи тип `ForeignKey` зі зв'язком за моделлю `Fishingactivityreport`.

Клас `Fishingactivityreportcomment` містить поле `fisher_id` для рибалки, який створив коментар до звіту за активністю, пов'язаною з риболовлею, маючи тип `ForeignKey` зі зв'язком за моделлю `Fisher`.

Клас `Fishingactivityreportcomment` містить поле `fishing_comment` для коментаря, який було створено рибалкою до звіту за активністю, пов'язаною з риболовлею, маючи тип `CharField`.

Клас `Fishingactivityreportcomment` містить поле `fishing_comment_time` для часу, коли було створено цей коментар рибалки до звіту за активністю, пов'язаною з риболовлею, маючи тип `DateTimeField`.

Клас `Fishingactivityreportreaction` серед моделей забезпечує модель реакції на звіти рибалок за активностями, пов'язаними з риболовлею.

Клас `Fishingactivityreportreaction` містить поле `fishingactivityreport_id` для звіту за активністю, пов'язаною з риболовлею, реакцію на яку встановив рибалка, маючи тип `ForeignKey` зі зв'язком за моделлю `Fishingactivityreport`.

Клас `Fishingactivityreportreaction` містить поле `fisher_id` для рибалки, який таким чином прореагував на активність, пов'язану з риболовлею, маючи тип `ForeignKey` зі зв'язком за моделлю `Fisher`. Також клас містить поле `fishing_reaction_type` для типу реакції, застосованого рибалкою щодо звіту за активністю, пов'язаною з риболовлею, маючи тип `PositiveSmallIntegerField`.

3.3 Реалізація основних програмних функцій для пошуку знайомих для спільної риболовлі

Створення усіх шаблонів вебсторінок вебзастосунку пошуку знайомих для спільної риболовлі витримано в єдиному дизайні і стилі з застосуванням

для цього шаблону [12], який надається за вільною ліцензією. Кожна функціональна можливість передбачала розробку щонайменше одного шаблону вебсторінки та представлення, що реалізовувалось функцією в коді, яка заповнює даними цей шаблон та повертає його в результаті роботи.

Перегляд активностей, пов'язаних з риболовлю, запланованих рибалкою, реалізовано функцією `fisher_planned_fishingactivities_view`, що є представленням з параметрами: `fisher_username` – логін рибалки, за яким відбираються активності, пов'язані з рибалкою, а якщо значення не задано, то це мають бути дані відносно того, як авторизувався користувач, `fishingactivities_state` – стан активностей, які потрібно відобразити. Окрім того кожне Django-представлення має стандартний параметр `request`, який містить інформацію про виконаний запит, про користувача, який авторизувався або не авторизувався. Розроблений шаблон вебсторінки має назву `fisheractivities.html`.

Перегляд активностей, пов'язаних з риболовлю, в яких планує взяти участь рибалка, реалізовано функцією `fisher_planned_participant_fishingactivities_view`, що є представленням з параметром `fishingactivities_state`, що задає стан активностей, які потрібно відобразити. Розроблений шаблон вебсторінки має назву `fisherpartactivities.html`.

Перегляд запланованих активностей, пов'язаних з риболовлю, які влаштовуються знайомими, реалізовано функцією `fishermate_planned_fishingactivities_view`, що є представленням з параметром `fishingactivities_state`, що задає стан активностей, які потрібно відобразити. Відбираються встановлені знайомства рибалок, а тоді за ними відбираються опубліковані цими користувачами активності, пов'язані з риболовлю. Розроблений шаблон вебсторінки має назву `fishermateactivities.html`.

Перегляд запланованих активностей, пов'язаних з риболовлю, в яких планують взяти участь знайомі, реалізовано функцією `fishermate_participant_planned_fishingactivities_view`, що є представленням з

параметром `fishingactivities_state`, що задає стан активностей, які потрібно відобразити. Розроблений шаблон вебсторінки має назву `fishermatepartactivities.html`.

Перегляд звітувань рибалки за досвідом активностей, пов'язаних з риболовлею, реалізовано функцією `fisher_fishingactivityreports_view`, що є представленням з параметрами: `fisher_username` – логін рибалки, за яким відбираються звітування за активностями, пов'язаними з рибалкою, а якщо значення не задано, то це мають бути дані відносно того, як авторизувався користувач, `fishingactivreport_state` – варіант звітів за активностями, пов'язаними з риболовлею, які потрібно відобразити. Розроблений шаблон вебсторінки має назву `fisherreports.html`.

Перегляд звітувань знайомих рибалок за досвідом активностей, пов'язаних з риболовлею, реалізовано функцією `fishermate_fishingactivityreports_view`, що є представленням з параметром `fishingactivreport_state`, який задає варіант звітів за активностями, пов'язаними з риболовлею, які потрібно відобразити. Розроблений шаблон вебсторінки має назву `fishermatereports.html`.

Перегляд звітувань за досвідом активностей, пов'язаних з риболовлею, що отримали власні реакції, реалізовано функцією `fisher_reacted_fishingactivityreports_view`, що є представленням з параметрами: `reaction_type` – тип реакції, за яким потрібно відобразити активності, пов'язані з риболовлею, `fishingactivreport_state` – варіант звітів за активностями, пов'язаними з риболовлею, які потрібно відобразити. Розроблений шаблон вебсторінки має назву `reactfishreports.html`.

Перегляд звітувань за досвідом активностей, пов'язаних з риболовлею, в яких застосовано певний засіб реалізовано функцією `fishgoodmodel_fishingactivityreports_view`, що є представленням з параметрами: `fishgoodmodel_id` – ідентифікатор моделі засобів для риболовлі, використання якої має бути зафіксовано в звітуваннях за досвідом

активностей, пов'язаних з риболовлею, що відображаються. Розроблений шаблон вебсторінки має назву `fishgoodactreports.html`.

Пошук серед активностей, пов'язаних з риболовлею, реалізовано функцією `fishingactivity_search_view`, що є представленням з параметрами, сформованими через ключі GET-запиту `fishcategory`, що задає ідентифікатор категорії активності, пов'язаної з риболовлею, `plannedstart`, що задає найбільш ранній час початку активностей, пов'язаних з риболовлею, `plannedend`, що задає найбільш пізній час початку активностей, пов'язаних з риболовлею, `reservoir`, що задає тип водойми, де мають відбуватися активності, пов'язані з риболовлею, `fishscore`, що задає мінімальну допустиму оцінку активності, `activityway`, що задає напрям сортування результатів, `participantmin` – мінімальна зареєстрована кількість учасників, `participantmax` – максимальна зареєстрована кількість учасників. Розроблений шаблон вебсторінки має назву `fishactivitysearch.html`.

Перегляд інформації про активність, пов'язану з риболовлею, реалізовано функцією `fishingactivity_view`, що є представленням з параметром `fishingactivity_id`, що задає ідентифікатор цієї активності, пов'язаної з риболовлею. Розроблений шаблон вебсторінки має назву `fishact.html`.

Перегляд інформації про звітування за активністю, пов'язаною з риболовлею, реалізовано функцією `fishingactivityreport_view`, що є представленням з параметром `fishingactivityreport_id`, що задає ідентифікатор цього звітування за активністю, пов'язаною з риболовлею. Розроблений шаблон вебсторінки має назву `fishactreport.html`.

Перегляд профільних даних рибалки реалізовано функцією `fisher_view`, що є представленням з параметром `fisher_username`, який задає логін рибалки, що може бути не заданий, а в тому випадку використовуються дані користувача після авторизації. Розроблений шаблон вебсторінки має назву `fisher.html`.

Перегляд інформації про рибальський засіб реалізовано функцією `fishinggoodmodel_view`, що є представленням з параметром

fishinggoodmodel_id, що задає ідентифікатор моделі засобу для риболовлі. Розроблений шаблон вебсторінки має назву fishgood.html.

Планування активності, пов'язаної з риболовлею реалізовано функцією fishingactivity_create_view, яка працює в першу чергу на основі класу FishingactivityForm, що підтримує роботу форми за структурою активності, пов'язаної з риболовлею. Розроблений шаблон вебсторінки має назву fishactivityproc.html. Редагування активності, пов'язаної з риболовлею реалізовано функцією fishingactivity_edit_view, що є представленням з параметром fishingactivity_id, який задає ідентифікатор активності, пов'язаної з риболовлею, редагування даних якої має відбуватись.

Додавання до активності, пов'язаної з риболовлею, в якості учасника реалізовано функцією fishingactivity_participant_view, що є представленням з параметром fishingactivity_id, що задає ідентифікатор активності, пов'язаної з риболовлею, учасником якої збирається стати рибалка, який авторизувався. Цією ж функцією реалізовано скасування участі в активності, пов'язаній з риболовлею. Якщо рибалка вже є учасником цієї активності, то участь скасовується, а якщо ні, то додається.

Перегляд знайомих для спільної риболовлі реалізовано функцією fishermates_view. Розроблений шаблон вебсторінки має назву fishermates.html. Він же використовується для перегляду заявок на знайомство для спільної риболовлі, що подані або отримані. Для цього реалізовано функцію, що є представленням з параметром mate_type, що визначає, чи це подані заявки, чи отримані.

Подання заявки на знайомство для спільної риболовлі реалізовано функцією fishermate_create_view, що є представленням, яке працює головним чином на основі класу форми FishermateForm.

Підтвердження заявки на знайомство для спільної риболовлі реалізовано функцією fishermate_approve_view, що є представленням з параметром fishermate_id, який задає ідентифікатор знайомства для спільної риболовлі, заявка за яким має бути підтверджена. Скасування заявки на

знайомство для спільної риболовлі реалізовано функцією `fishermate_cancel_view`, що є представленням з параметром `fishermate_id`, який задає ідентифікатор знайомства для спільної риболовлі, заявка за яким має бути скасована.

Пошук знайомих для спільної риболовлі реалізовано функцією, що є представленням з параметрами: `fishingtype` – необхідний варіант риболовлі, `fishingwinter` – необхідність зимової рибалки, `fishingreservoir` – тип водойми, на якій має рибалити знайомий, `fishsort` – рід риби, яку має рибалити знайомий, `fishregion` – область, в якій є активності, пов'язані з риболовлею, у знайомого. Розроблений шаблон вебсторінки має назву `fishersearch.html`.

Звітування за власним досвідом активності, пов'язаної з риболовлею реалізовано функцією `fishingactivityreport_create_view`, що є представленням, що головним чином працює на основі класу форми `FishingactivityreportForm`. Редагування звітування за власним досвідом активності, пов'язаної з риболовлею, реалізовано функцією `fishingactivityreport_edit_view` з параметром `fishingactivityreport_id`, який задає ідентифікатор звітування за досвідом активності, пов'язаної з риболовлею. В основі також використовується клас `FishingactivityreportForm`.

Оцінювання активності, пов'язаної з риболовлею, реалізовано функцією `fishingactivityscore_create_view`, що є представленням з параметрами: `fishingactivity_id` – ідентифікатор активності, пов'язаної з риболовлею, `fishing_score` – виставлена оцінка активності.

Коментування звітування за досвідом активності, пов'язаної з риболовлею, реалізовано функцією `fishingactivityreportcomment_create_view`, що є представленням з параметрами: `fishingactivityreport_id` – ідентифікатор звітування за досвідом активності, пов'язаної з риболовлею, `fishing_comment` – текст коментаря.

Реагування на звітування за досвідом активності, пов'язаної з риболовлею, реалізовано функцією `fishingactivityreport_reaction_create_view`, що є представленням з параметрами: `fishingactivityreport_id` – ідентифікатор

звітування за досвідом активності, пов'язаної з риболовлею, `fishing_reaction_type` – тип виставленої реакції. Ця ж функція реалізує і скасування реагування на звітування за досвідом активності, пов'язаної з риболовлею. Це відбувається тоді, коли така реакція вже була проставлені цьому ж звітуванню цим же рибалкою.

3.4 Висновки за розділом 3

Знайомства для спільної риболовлі як ключова сутність предметної області була визначена стосовно можливих станів та переходів між цими станами створенням та описом діаграми станів. Описано результати реалізації основних програмних функцій та класів для пошуку знайомих для спільної риболовлі.

4 ЕКСПЛУАТАЦІЯ ТА ТЕСТУВАННЯ ПРОГРАМИ

4.1 Призначення програми пошуку знайомих для спільної риболовлі

Створена програма призначена для організації пошуку знайомих для спільної риболовлі, включаючи як пошук самих нових знайомих, так і пошук активностей, пов'язаних з риболовлею. Таким чином, застосування розробленої програми базується не тільки на створенні відповідного пошуку за параметрами, але і на перегляді того досвіду, який потенційні знайомі раніше мали під час риболовлі, тих підходів до риболовлі, яким віддає переваги рибалка.

4.2 Умови виконання програми пошуку знайомих для спільної риболовлі

Умови виконання програми пошуку знайомих для спільної риболовлі містять вимоги до технічного рівня сервера, визначаючи його на мінімальному рівні як такий, що має 2 Гб оперативної пам'яті, 2 ядра процесора, 100 Гб для організації сховища, що має надати можливість виконання сумарно 25000 запитів на місяць до вебзастосунку. Для безпосередньо експлуатації програми необхідно також забезпечити вимоги до спеціаліста-адміністратора, який має забезпечити хостинг, розгортання вебзастосунку на сервері з застосуванням технологій, що включають Python, Django, PostgreSQL.

4.3 Експлуатація програми пошуку знайомих для спільної риболовлі

При зверненні за базовою адресою вебзастосунку пошуку знайомих для спільної риболовлі користувач у перший раз працює як неавторизований, реалізуючи роботу через доступ до основних пунктів меню, а далі переходячи

за відповідними функціями, які надаються цими сторінками та ведуть до інших сторінок. Якщо сторінка, яка відображається, безпосередньо доступна з меню, то пункт меню або розділ меню підсвічуються. Якщо відбувається перехід на наступні сторінки, то підсвічування вже не реалізовується.

Неавторизований користувач може скористатися пошуком знайомих для спільної риболовлі, визначаючи при цьому власні переваги щодо типів водойм, варіантів риболовлі тощо, а пошук здійснюється стосовно співпадіння інтересів серед тих рибалок, які висловили бажання в такому пошуку брати участь. Відповідно зі сторінки рибалки користувач може перейти до перегляду запланованих ним активностей (рис. 4.1). Усі активності, пов'язані з риболовлею, розбиті на вибір всіх, тих, які будуть відбуватися в майбутньому, та тих, які вже відбулись. Стосовно обраного варіанту далі задається варіант сортування.

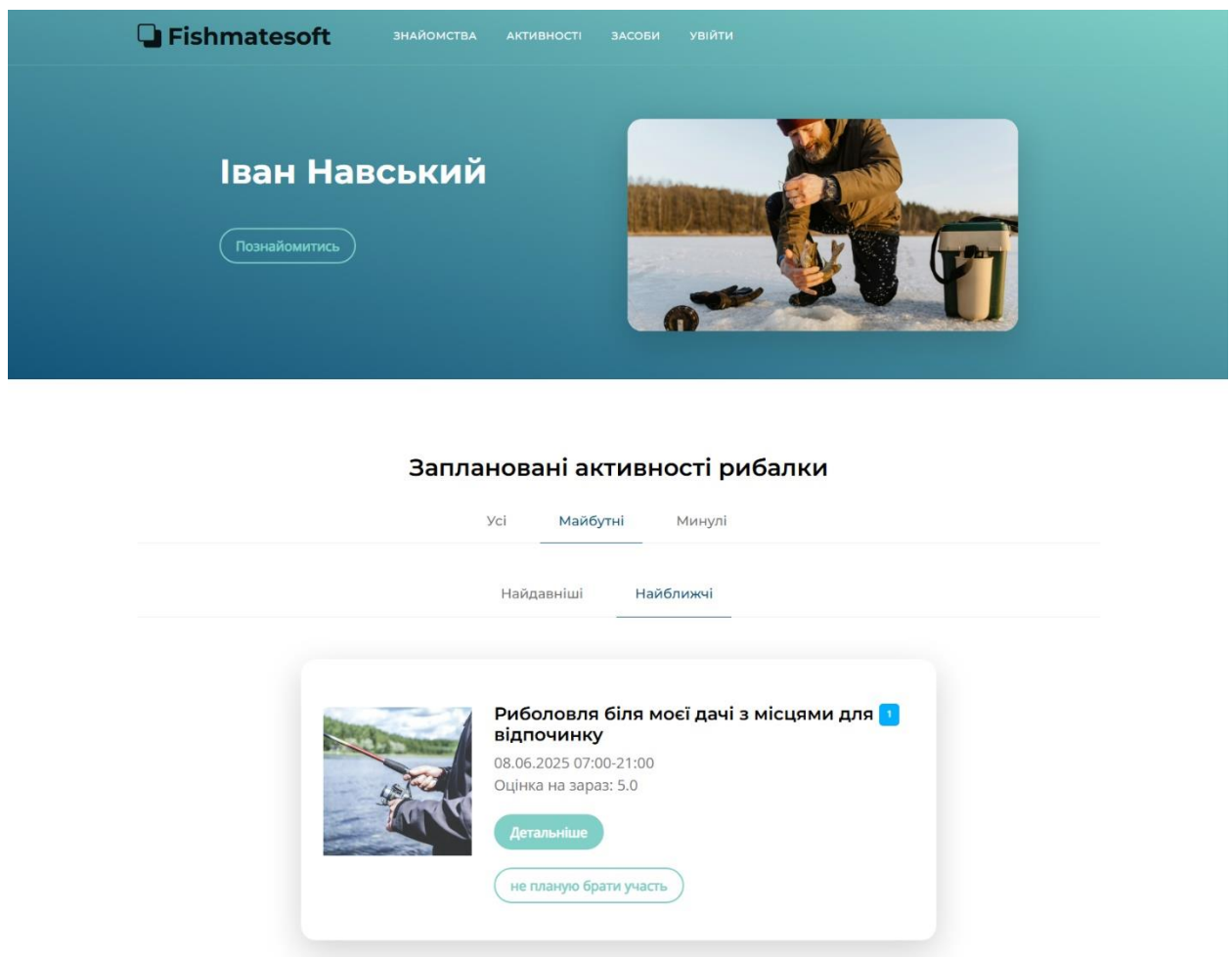


Рисунок 4.1 – Заплановані активності, пов'язані з риболовлею, у рибалки

І неавторизованому користувачу демонструються посилання у вигляді кнопок для роботи з якими необхідно авторизуватись. Проте у такому випадку він вимушений буде спочатку авторизуватись. Так користувач має посилання для того, щоб оформити заявку на знайомство для спільної риболовлі, посилання для визначення власної участі в запланованій активності. За кожною активністю, пов'язаною з риболовлею, відображається у синьому квадраті кількість учасників, які виявили таке бажання.

Пункти меню неавторизованого користувача також дозволяють переходити до пошуку активностей, пов'язаних з риболовлею, до пошуку засобів для риболовлі, а також переходити до авторизації.

Коли ж користувач авторизувався, то як рибалці після авторизації йому доступно ширше меню, яке дозволяє переходити до тих функцій, які передбачають його початкову ідентифікацію.

Розділ меню «Знайомства» для рибалки після авторизації включають можливості пошуку знайомих та відображення сторінки зі знайомими для спільної риболовлі та запитами на знайомства (рис. 4.2).

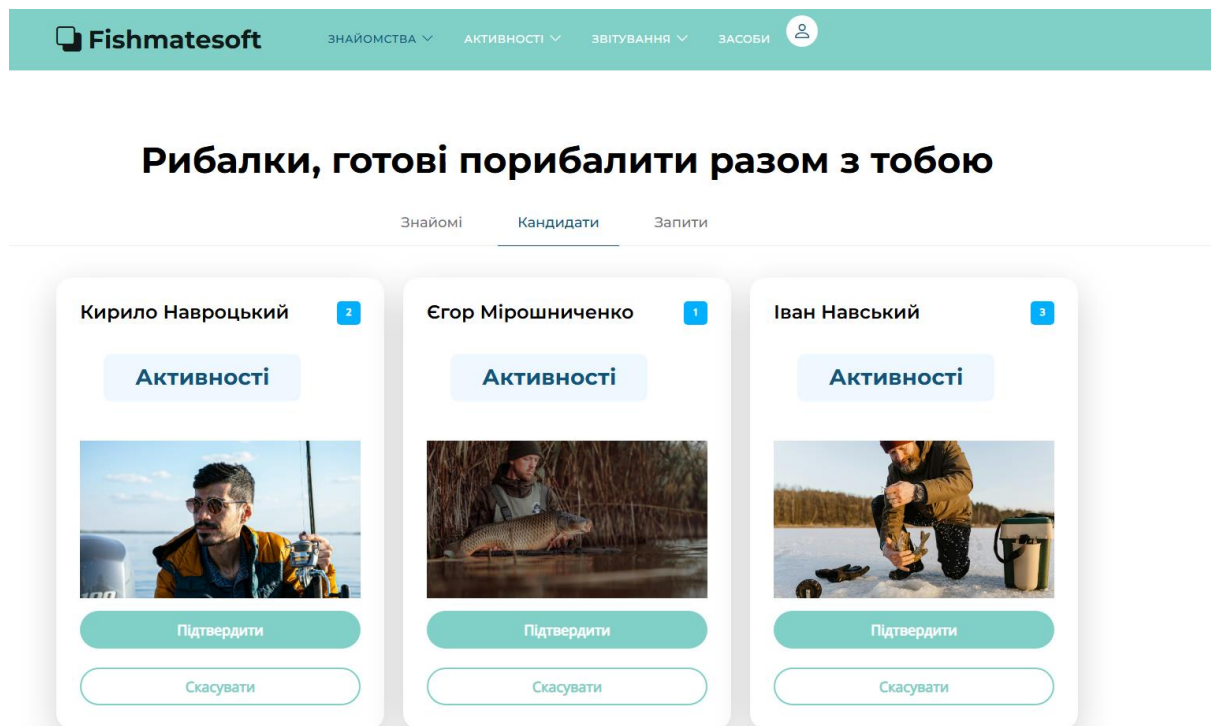


Рисунок 4.2 – Сторінка зі знайомими для спільної риболовлі та запитами

Сторінка зі знайомими для спільної риболовлі та запитами на знайомства містить можливість вибору відображення тільки спільних знайомих, тільки тих запитів, які були подані на адресу рибалки, який авторизувався («Кандидати»), та тільки тих запитів, які подав сам авторизований користувач («Запити»). Користувач може вилучати ті знайомства, які не є більше актуальними, скасовувати свої запити, а також підтверджувати або скасовувати запити на його адресу.

За підтвердженими знайомствами для спільної риболовлі можна переглянути ті активності, пов'язані з риболовлею, які такі користувачі створили (рис. 4.3). Відображаються тоді активності, пов'язані з риболовлею, всіх користувачів, з якими оформлено знайомства.

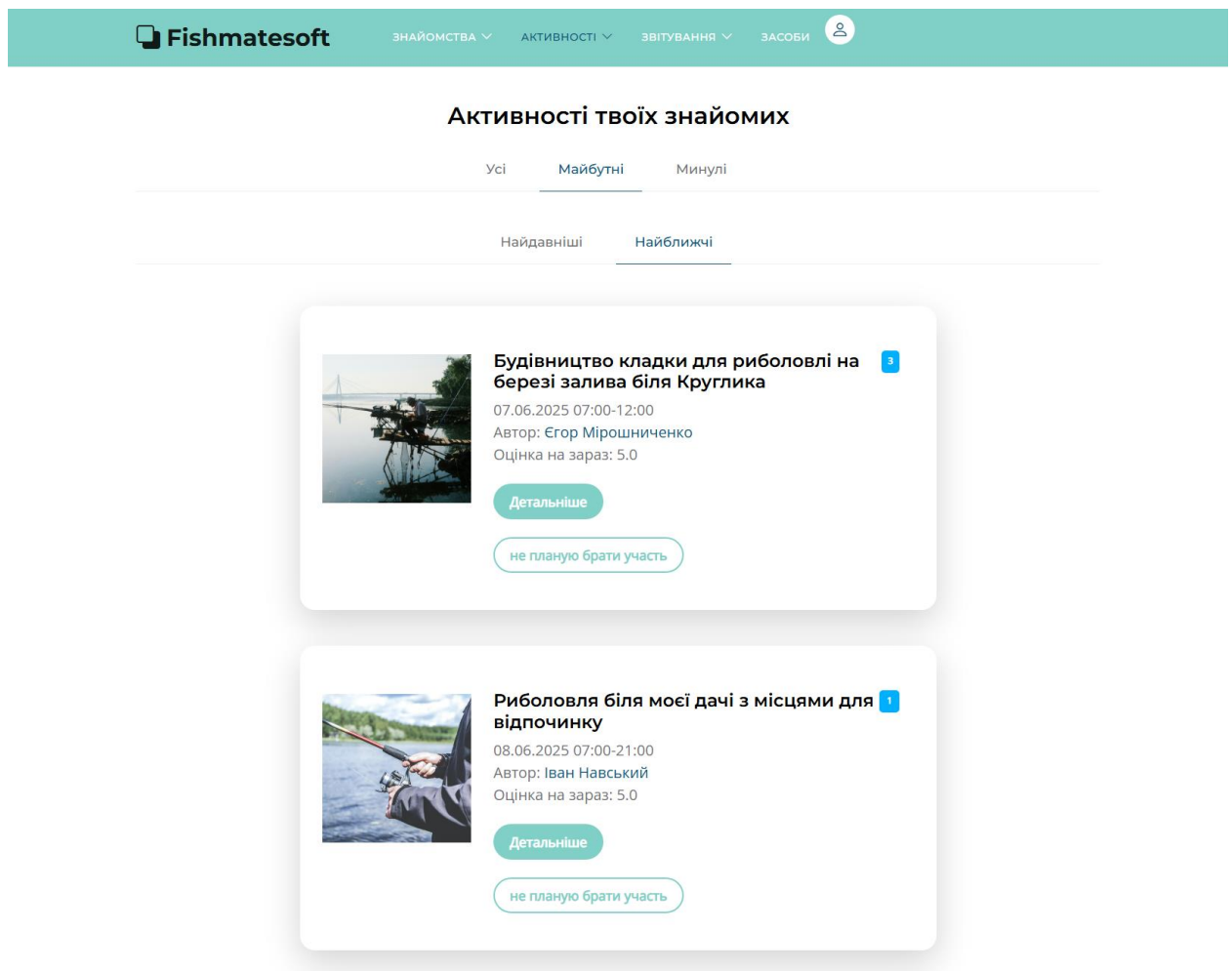
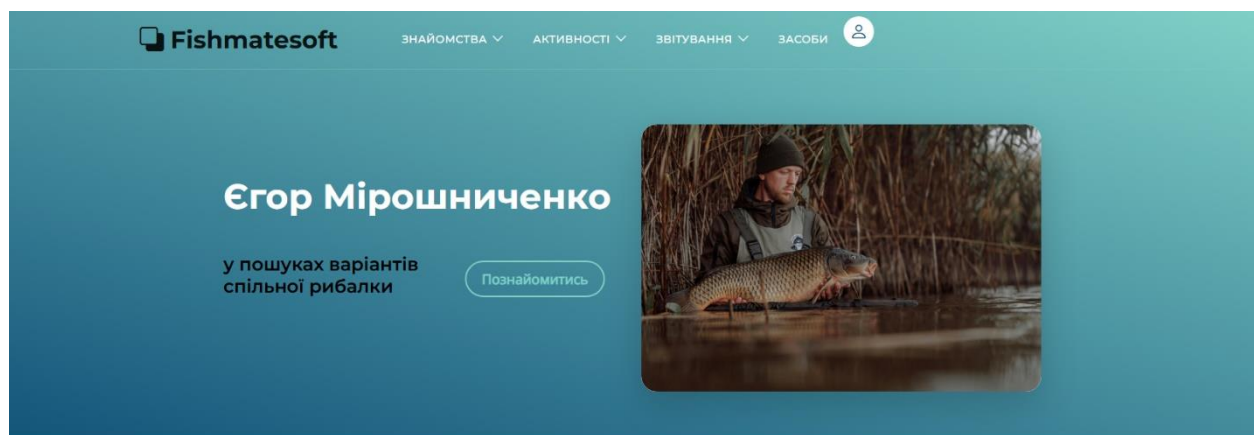


Рисунок 4.3 – Активності, пов'язані з риболовлею

З розділу активностей доступні також можливості пошуку активностей, відображення активностей, пов'язаних з риболовлю, в яких беруть участь знайомі, в яких запланував взяти участь сам рибалка, що авторизувався, активності, які він сам створив, а також можливість запланувати нову активність, пов'язану з риболовлю.

Якщо натиснути на автора однієї з активностей, пов'язаних з риболовлю, відкрити сторінку з результатів пошуку знайомих для спільної риболовлі, то відкриється сторінка рибалки (рис. 4.4). З неї можна перейти до активностей, пов'язаних з риболовлю, у цьому рибалки, а можна переглянути його смаки у риболовлі. Також нижче на сторінці розташовані всі надані його фотографії.



всього знайомих зараз: 1

всього активностей: 1

Улюблені місця риболовлі...

затоки річок
озера

Улюблені улови...

короп
щука

Як люблю рибалити...

з човна
з берега
спортивна
зимова

Люблю активно проводити час. У вихідні часто на риболовлі. Колег поки не дуже багато, але справа затягує. Когось знаю на місцях, але хотілось би мати більше знайомих. Маю човен, машину. І любов до риболовлі!

Рисунок 4.4 – Сторінка рибалки в програмі

З розділу звітувань доступні можливості відтворення тих звітувань за активностями, пов'язаними з риболовлею, на які поставив реакції рибалка, що авторизувався. Також можна відобразити звітування всіх знайомих зі спільної риболовлі. Звітування оформлюються за активностями, відповідно доступні зі сторінки активності, пов'язаної з риболовлею. Якщо активності рибалки можуть оцінювати, то на звітування вони встановлюють реакції, можуть писати коментарі.

Визначити ж, чи доступна його сторінка для пошуку знайомств для спільної риболовлі рибалка може зі сторінки профілю, що доступна з меню під іконкою з позначенням користувача. Звідти контексте меню дозволяє вибрати засоби виходу з облікового запису, редагування профільних даних, відображення цих даних.

4.4 Тестування програми пошуку знайомих для спільної риболовлі

Тестування програми пошуку знайомих для спільної риболовлі відбувалось після розміщення її на сервері з виділенням адреси для віддаленого доступу та використанням різних пристроїв, включаючи як персональний комп'ютер, ноутбук, так і мобільні пристрої під керуванням системи Android. При цьому перевірялась робота кожної функції стосовно того, чи було за результатами отримано очікуваний результат. Було створено чек-лист (табл. 4.1), в якому навпроти кожної функції проставлялась позначка «+», коли було отримано очікуваний результат, а позначка «-», коли результат відрізнявся від очікуваного. Позначка не застосовується (НЗ) визначалась тоді, коли для такого типу користувача така функція не може бути використана. Тестування усіх функцій призвело до отримання очікуваних результатів і завершено успішно.

Таблиця 4.1 – Чек-лист тестування програми пошуку знайомих для спільної риболовлі

Функція	Неавторизований користувач	Рибалка після авторизації
1	2	3
Перегляд активностей, пов'язаних з риболовлею, запланованих рибалкою	+	+
Пошук серед активностей, пов'язаних з риболовлею	+	+
Пошук знайомих для спільної риболовлі	+	+
Перегляд інформації про активність, пов'язану з риболовлею	+	+
Перегляд профільних даних рибалки	+	+
Перегляд звітувань рибалки за досвідом активностей, пов'язаних з риболовлею	+	+
Перегляд звітувань за досвідом активностей, пов'язаних з риболовлею, в яких застосовано певний засіб	+	+
Авторизація рибалки	+	НЗ
Пошук рибальських засобів	+	+
Реєстрація рибалки	+	НЗ
Перегляд інформації про рибальський засіб	+	+
Перегляд інформації про звітування рибалки за активністю, пов'язаною з риболовлею	+	+
Планування активності, пов'язаної з риболовлею	НЗ	+
Додавання до активності, пов'язаної з риболовлею, в якості учасника	НЗ	+

Продовження таблиці 4.1

1	2	3
Перегляд активностей, пов'язаних з риболовлю, в яких планує взяти участь рибалка	НЗ	+
Внесення профільних даних рибалки	НЗ	+
Виставлення власної сторінки рибалки для спільної рибалки	НЗ	+
Подання заявки на знайомство для спільної риболовлі	НЗ	+
Підтвердження заявки на знайомство для спільної риболовлі	НЗ	+
Перегляд знайомих для спільної риболовлі	НЗ	+
Перегляд заявок на знайомство для спільної риболовлі, що подані або отримані	НЗ	+
Звітування за власним досвідом активності, пов'язаної з риболовлю	НЗ	+
Оцінювання активності, пов'язаної з риболовлю	НЗ	+
Коментування звітування за досвідом активності, пов'язаної з риболовлю	НЗ	+
Перегляд запланованих активностей, пов'язаних з риболовлю, які влаштовуються знайомими	НЗ	+
Перегляд запланованих активностей, пов'язаних з риболовлю, в яких планують взяти участь знайомі	НЗ	+
Перегляд звітувань знайомих рибалок за досвідом активностей, пов'язаних з риболовлю	НЗ	+
Реагування на звітування за досвідом активності, пов'язаної з риболовлю	НЗ	+

Кінець таблиці 4.1

1	2	3
Перегляд звітувань за досвідом активностей, пов'язаних з риболовлею, що отримали власні реакції	НЗ	+
Скасування реагування на звітування за досвідом активності, пов'язаної з риболовлею	НЗ	+
Редагування звітування за власним досвідом активності, пов'язаної з риболовлею	НЗ	+
Скасування заявки на знайомство для спільної риболовлі	НЗ	+
Скасування власної сторінки рибалки в якості кандидата для спільної рибалки	НЗ	+
Редагування профільних даних рибалки	НЗ	+
Скасування участі в активності, пов'язаній з риболовлею	НЗ	+
Редагування активності, пов'язаної з риболовлею	НЗ	+

4.5 Висновки за розділом 4

Створений інтерфейс програми пошуку знайомих для спільної риболовлі у вигляді набору відповідних вебсторінок є простим і зрозумілим щодо вибору необхідних функціональних можливостей під виконання відповідних завдань користувача, як неавторизованого, так і рибалки після авторизації. Тестування було проведено повністю за всім набором функцій, що були визначені для реалізації, з отриманням успішних результатів у всіх ситуаціях.

ВИСНОВКИ

У роботі шляхом успішного виконання всіх поставлених завдань було досягнуто мету розробити програмне забезпечення пошуку знайомих для спільної риболовлі на основі інформування про власний досвід риболовлі.

Першочергово проведений огляд можливостей існуючих програм комунікації всередині спільнот рибалок продемонстрував, що безпосередніх засобів пошуку знайомих для спільної риболовлі вони не мають, а встановлення контактів відбувається у загальних соціальних мережах. У той же час такі соціальні мережі не враховують особливостей риболовлі, а тому не сприяють додатково підтримці даного процесу, роблячи його доволі фрагментованим. На усунення таких недоліків і була спрямована робота.

Стосовно вибору засобів розробки програми пошуку знайомих для спільної риболовлі було прийнято такі рішення: для роботи з БД – система управління PostgreSQL, для створення вебзастосунку – мова Python, фреймворк Django.

Розроблена програма пошуку знайомих для спільної риболовлі враховує сучасні вимоги до такого типу вебзастосунків, використовує сучасні технології, сприяючи універсальності реалізації функціональних можливостей та враховуючи ці вимоги під час проєктування.

Застосування розробленої програми для пошуку знайомих для спільної риболовлі базується не тільки на створенні відповідного пошуку за параметрами, але і на перегляді того досвіду, який потенційні знайомі раніше мали під час риболовлі, тих підходів до риболовлі, яким віддає переваги рибалка. Для цього програма надає ряд спрямованих інструментів, що полегшують ці процеси та сприяють прийняттю рішень, які можуть призвести до позитивного досвіду спільної риболовлі. Усе це дозволяє використовувати її для створення платформи для комунікації між рибалками та планування спільних активностей з забезпеченням розвитку спільноти рибалок, рибальських клубів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. U.S. Fishing Goods Market Size. Industry Report, 2030 [Electronic resource]. – Access mode : <https://www.grandviewresearch.com/industry-analysis/us-fishing-goods-market-report>.
2. Новицька, А. ГО «Клуб рибалок України» з Краматорська відкрив школу риболовного мистецтва для дітей у Києві [Електрон. ресурс] / А. Новицька. – Режим доступу : <https://pro100media.com.ua/eksklyuzyv/go-klub-rybalok-ukrayiny-z-kramatorska-vidkryv-shkolu-rybolovnogo-mystetstva-dlya-ditej-u-kyievi/>.
3. Каразуб, І. Чим рибалили в давнину. В Івано-Франківську відкрили рибальський клуб-музей [Електрон. ресурс] / І. Каразуб. – Режим доступу : <https://suspilne.media/ivano-frankivsk/208721-cim-ribalili-v-davninu-v-ivano-frankivsku-vidkrili-ribalskij-klub-muzej/>.
4. У мережі опублікували перелік приватних водойм для риболовлі на Київщині [Електрон. ресурс]. – Режим доступу : <https://mykyivregion.com.ua/news/u-merezi-opublikovali-perelik-privatnix-vodoim-dlya-ribolovli-v-peredmisti-kijeva>.
5. Відкриття спортивного спінінгового сезону 2022. Facebook [Електрон. ресурс]. – Режим доступу : <https://www.facebook.com/events/317720396972401/>.
6. Головна. WINFISH – Озеро Чемпіонів [Електрон. ресурс]. – Режим доступу : <https://lake.winfish.com.ua/>.
7. Головна – Selfish Club [Електрон. ресурс]. – Режим доступу : <https://selfish.ua/>.
8. Клуб Рибалок «Відпочинок» [Електрон. ресурс]. – Режим доступу : <https://www.clubrybalok.com.ua/>.
9. Django FAQ. Django documentation. Django [Electronic resource]. – Access mode : <https://docs.djangoproject.com/en/5.2/faq/>.
10. 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.

11. Deery, M. What Is Flask and How Do Developers Use It? A Quick Guide [Electronic resource] / M. Deery. – Access mode : <https://careerfoundry.com/en/blog/web-development/what-is-flask/>.

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

ДОДАТОК А
Технічне завдання

Вступ

Риболовля є одним з відомих активних хобі для багатьох людей. Спортивна риболовля привертає велику увагу зокрема і в Україні навіть під час війни. Проводиться велика кількість заходів зі спортивної риболовлі. Створюється велика кількість клубів для риболовлі навколо окремих водойм або місць біля цих водойм. У той же час всі ці спільноти існують достатньо відокремлено. Комунікація пов'язана з персональними контактами, які мають налагоджуватись. Тому логічним є створення програмного засобу, який би підтримував налагодження таких контактів. Для цього розробка повинна надавати ряд спрямованих інструментів, що полегшують ці процеси та сприяють прийняттю рішень, які можуть призвести до позитивного досвіду спільної риболовлі.

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 пошуку знайомих для спільної риболовлі» як тему роботи.

A.2 Призначення розробки

Розробка призначена для організації пошуку знайомих для спільної риболовлі. Застосування розроблюваної програми повинно базуватися не тільки на створенні відповідного пошуку за параметрами, але і на перегляді того досвіду, який потенційні знайомі раніше мали під час риболовлі, тих підходів до риболовлі, яким віддає переваги рибалка.

А.3 Основні вимоги до програми

А.3.1 Вимоги до функціональних характеристик

Вимоги до програми пошуку знайомих для спільної риболовлі за набором функцій включають:

- планування активності, пов’язаної з риболовлею;
- редагування активності, пов’язаної з риболовлею;
- додавання до активності, пов’язаної з риболовлею, в якості учасника;
- скасування участі в активності, пов’язаній з риболовлею;
- перегляд активностей, пов’язаних з риболовлею, в яких планує взяти участь рибалка;
- перегляд активностей, пов’язаних з риболовлею, запланованих рибалкою;
- внесення та редагування профільних даних рибалки;
- виставлення власної сторінки рибалки для спільної рибалки;
- пошук серед активностей, пов’язаних з риболовлею;
- пошук знайомих для спільної риболовлі;
- подання заявки на знайомство для спільної риболовлі;
- підтвердження або скасування заявки на знайомство для спільної риболовлі;
- перегляд знайомих для спільної риболовлі;
- перегляд заявок на знайомство для спільної риболовлі, що подані або отримані;
- звітування за власним досвідом активності, пов’язаної з риболовлею;
- оцінювання активності, пов’язаної з риболовлею;
- коментування звітування за досвідом активності, пов’язаної з риболовлею;
- перегляд інформації про активність, пов’язану з риболовлею;
- перегляд профільних даних рибалки;
- перегляд запланованих активностей, пов’язаних з риболовлею, які

влаштовуються знайомими;

- перегляд запланованих активностей, пов'язаних з риболовлею, в яких планують взяти участь знайомі;

- перегляд звітувань знайомих рибалок за досвідом активностей, пов'язаних з риболовлею;

- редагування звітування за власним досвідом активності, пов'язаної з риболовлею;

- перегляд звітувань рибалки за досвідом активностей, пов'язаних з риболовлею;

- перегляд звітувань за досвідом активностей, пов'язаних з риболовлею, в яких застосовано певний засіб;

- реагування на звітування за досвідом активності, пов'язаної з риболовлею;

- пошук рибальських засобів;

- скасування власної сторінки рибалки в якості кандидата для спільної рибалки;

- перегляд інформації про рибальський засіб.

- авторизація і реєстрація рибалок;

- перегляд інформації про звітування рибалки за активністю, пов'язаною з риболовлею.

A.3.2 Умови експлуатації

Експлуатаційні вимоги до вебзастосунку пошуку знайомих для спільної риболовлі включають вимоги до спеціаліста-адміністратора, який має забезпечити хостинг, розгортання вебзастосунку на сервері з застосуванням обраного для реалізації вебфреймворку, підтримку роботи бази даних на основі обраної в роботі системи керування.

Відповідно експлуатація такого вебзастосунку пошуку знайомих для спільної риболовлі має бути забезпечена можливістю доступу до технологій,

на основі яких реалізовано вебзастосунок та які відповідно були обрані в роботі як підсумок.

A.3.3 Вимоги до складу та параметрів технічних засобів

Сервер на технічному рівні повинен щонайменше мати 2 Гб оперативної пам'яті, два ядра процесора, 100 Гб для організації сховища, що має надати можливість виконання сумарно 25000 запитів на місяць до вебзастосунку, але в подальшому кількість запитів та інші варіанти навантажень мають бути оцінені додатково, що може призвести до розширення мінімальних вимог, окрім того відсоток вільного простору для організації сховища та темпи його змінювання мають оцінюватися та призводити до прийняття рішень щодо розширення доступного простору за необхідності.

A.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 django.shortcuts import render
from .models import Fishinggoodcategory
from .models import Fishinggoodsubcategory
from .models import Fishinggoodmanufacturer
from .models import Fishinggoodmodel
from .models import Fishingactivitycategory
from .models import Fishingregion
from .models import Fishingreservoir
from .models import Fishingreservoirfishsort
from .models import Fishingactivity
from .models import Fishingactivityphoto
from .models import Fisher
from .models import Fisherreservoir
from .models import Fishsort
from .models import Fisherphoto
from .models import Fishermate
from .models import Fishingactivityreport
from .models import Fishingactivityreportphoto
from .models import Fishingactivityreportgood
from .models import Fishingactivityreportgoodphoto
from .models import Fishingactivityscore
from .models import Fishingactivityreportcomment
from .models import Fishingactivityreportreaction
from django.core.paginator import Paginator

#Перегляд активностей, пов'язаних з риболовлю,
запланованих рибалкою
def fisher_planned_fishingactivities_view(request,
fisher_username = None, fishingactivities_state = 0):
    if not fisher_username:
        if request.user.is_authenticated:
            try:
                thefisher =
Fisher.objects.get(fisher_user_name = request.user.username)
            except Fisher.DoesNotExist:
                thefisher = None
        else:
            thefisher = None
    else:
        try:
            thefisher = Fisher.objects.get(fisher_user_name
= fisher_username)
        except Fisher.DoesNotExist:
            thefisher = None

```

```

        if not thefisher:
            return render(request, "fishmatesoft/error.html",
{"comment": "Такий рибалка відсутній, дані було надано некоректно"})
        import datetime
        fishingactivity_currenttime = datetime.datetime.now()
        if fishingactivities_state == 0:
            fishingactivities =
Fishingactivity.objects.prefetch_related("fishers").select_relat
ed("fishingreservoir_id",
"fishingactivitycategory_id").filter(initiative_fisher_id =
thefisher)
            fishingactivities = fishingactivities.order_by("-
fishing_planned_start_time")
        elif fishingactivities_state == 1:
            fishingactivities =
Fishingactivity.objects.prefetch_related("fishers").select_relat
ed("fishingreservoir_id",
"fishingactivitycategory_id").filter(initiative_fisher_id =
thefisher)
            fishingactivities =
fishingactivities.filter(fishing_planned_start_time__gte =
fishingactivity_currenttime)
            fishingactivities = fishingactivities.order_by("-
fishing_planned_start_time")
        elif fishingactivities_state == 2:
            fishingactivities =
Fishingactivity.objects.prefetch_related("fishers").select_relat
ed("fishingreservoir_id",
"fishingactivitycategory_id").filter(initiative_fisher_id =
thefisher)
            fishingactivities =
fishingactivities.filter(fishing_planned_start_time__gte =
fishingactivity_currenttime)
            fishingactivities =
fishingactivities.order_by("fishing_planned_start_time")
        elif fishingactivities_state == 3:
            fishingactivities =
Fishingactivity.objects.prefetch_related("fishers").select_relat
ed("fishingreservoir_id",
"fishingactivitycategory_id").filter(initiative_fisher_id =
thefisher)

```

```

        fishingactivities =
fishingactivities.filter(fishing_planned_start_time__lt =
fishingactivity_currenttime)
        fishingactivities = fishingactivities.order_by("-
fishing_planned_start_time")
        elif fishingactivities_state == 4:
            fishingactivities =
Fishingactivity.objects.prefetch_related("fishers").select_relat
ed("fishingreservoir_id",
"fishingactivitycategory_id").filter(initiative_fisher_id =
thefisher)

            fishingactivities =
fishingactivities.filter(fishing_planned_start_time__lt =
fishingactivity_currenttime)
            fishingactivities =
fishingactivities.order_by("fishing_planned_start_time")
            fishingactivities_paginator =
Paginator(fishingactivities, 10)
            fishingactivities_requested_portion_number =
request.GET.get('page')
            fishingactivities =
fishingactivities_paginator.get_page(fishingactivities_requested
_portion_number)
            for fishingactivity in fishingactivities:
                fishingactivity_photos =
Fishingactivityphoto.objects.filter(fishingactivity_id =
fishingactivity)
                fishingactivity.photo = fishingactivity_photos[0]
                fishingactivity.fishers_number =
fishingactivity.fishers.count()
                fishingactivity_scores =
Fishingactivityscore.objects.filter(fishingactivity_id =
fishingactivity)
                fishingactivity_scores_number =
fishingactivity_scores.count()
                if request.user.is_authenticated:
                    try:
                        theafisher =
Fisher.objects.get(fisher_user_name = request.user.username)
                    except Fisher.DoesNotExist:
                        theafisher = None
                    if theafisher:
                        fishingactivity_observer_scores =
Fishingactivityscore.objects.filter(fishingactivity_id =
fishingactivity, fisher_id = theafisher)

```

```

        fishingactivity_observer_scores =
fishingactivity_observer_scores.order_by("-fishing_score_time")
        if fishingactivity_observer_scores:
            fishingactivity_observer_score =
fishingactivity_observer_scores[0]
        else:
            fishingactivity_observer_score = -1
    else:
        fishingactivity_observer_score = -1
    else:
        fishingactivity_observer_score = -1
    return render(request,
"fishmatesoft/fisheractivities.html", {"fishingactivities":
fishingactivities, "fisher": thefisher })

#Перегляд активностей, пов'язаних з риболовлю, в яких
планує взяти участь рибалка
def
fisher_planned_participant_fishingactivities_view(request,
fishingactivities_state = 0):
    if request.user.is_authenticated:
        try:
            thefisher = Fisher.objects.get(fisher_user_name
= request.user.username)
        except Fisher.DoesNotExist:
            thefisher = None
    else:
        thefisher = None
    if not thefisher:
        return render(request, "fishmatesoft/error.html",
{"comment": "Не вдалось вас повністю ідентифікувати. Сталась
помилка. Можливо дані рибалки було некоректно змінено"})
    import datetime
    fishingactivity_currenttime = datetime.datetime.now()
    if fishingactivities_state == 0:
        fishingactivities =
Fishingactivity.objects.prefetch_related("fishers").select_relat
ed("initiative_fisher_id", "fishingreservoir_id",
"fishingactivitycategory_id").filter(fishers = thefisher)
        fishingactivities = fishingactivities.order_by("-
fishing_planned_start_time")
    elif fishingactivities_state == 1:
        fishingactivities =
Fishingactivity.objects.prefetch_related("fishers").select_relat

```

```

ed("fishingreservoir_id",
"fishingactivitycategory_id").filter(fishers = thefisher)
    fishingactivities =
fishingactivities.filter(fishing_planned_start_time__gte =
fishingactivity_currenttime)
    fishingactivities = fishingactivities.order_by("-
fishing_planned_start_time")
    elif fishingactivities_state == 2:
        fishingactivities =
Fishingactivity.objects.prefetch_related("fishers").select_relat
ed("initiative_fisher_id", "fishingreservoir_id",
"fishingactivitycategory_id").filter(fishers = thefisher)
        fishingactivities =
fishingactivities.filter(fishing_planned_start_time__gte =
fishingactivity_currenttime)
        fishingactivities =
fishingactivities.order_by("fishing_planned_start_time")
        elif fishingactivities_state == 3:
            fishingactivities =
Fishingactivity.objects.prefetch_related("fishers").select_relat
ed("initiative_fisher_id", "fishingreservoir_id",
"fishingactivitycategory_id").filter(fishers = thefisher)
            fishingactivities =
fishingactivities.filter(fishing_planned_start_time__lt =
fishingactivity_currenttime)
            fishingactivities = fishingactivities.order_by("-
fishing_planned_start_time")
            elif fishingactivities_state == 4:
                fishingactivities =
Fishingactivity.objects.prefetch_related("fishers").select_relat
ed("initiative_fisher_id", "fishingreservoir_id",
"fishingactivitycategory_id").filter(fishers = thefisher)
                fishingactivities =
fishingactivities.filter(fishing_planned_start_time__lt =
fishingactivity_currenttime)
                fishingactivities =
fishingactivities.order_by("fishing_planned_start_time")
                fishingactivities_paginator =
Paginator(fishingactivities, 10)
                fishingactivities_requested_portion_number =
request.GET.get('page')
                fishingactivities =
fishingactivities_paginator.get_page(fishingactivities_requested
_portion_number)
                for fishingactivity in fishingactivities:

```

```

        fishingactivity_photos =
Fishingactivityphoto.objects.filter(fishingactivity_id =
fishingactivity)
        fishingactivity.photo = fishingactivity_photos[0]
        fishingactivity.fishers_number =
fishingactivity.fishers.count()
        fishingactivity_scores =
Fishingactivityscore.objects.filter(fishingactivity_id =
fishingactivity)
        fishingactivity.scores_number =
fishingactivity_scores.count()
        fishingactivity_observer_scores =
Fishingactivityscore.objects.filter(fishingactivity_id =
fishingactivity, fisher_id = thefisher)
        fishingactivity_observer_scores =
fishingactivity_observer_scores.order_by("-fishing_score_time")
        if fishingactivity_observer_scores:
            fishingactivity.observer_score =
fishingactivity_observer_scores[0]
        else:
            fishingactivity.observer_score = -1
    return render(request,
"fishmatesoft/fisherpartactivities.html", {"fishingactivities":
fishingactivities, "fisher": thefisher })

from django.db.models import Q

# Перегляд запланованих активностей, пов'язаних з
риболовлю, які влаштовуються знайомими
def fishermate_planned_fishingactivities_view(request,
fishingactivities_state = 0):
    if request.user.is_authenticated:
        try:
            thefisher = Fisher.objects.get(fisher_user_name
= request.user.username)
        except Fisher.DoesNotExist:
            thefisher = None
    else:
        thefisher = None
    if not thefisher:
        return render(request, "fishmatesoft/error.html",
{"comment": "Не вдалось вас повністю ідентифікувати. Сталась
помилка. Можливо дані рибалки було некоректно змінено"})
    import datetime
    fishingactivity_currenttime = datetime.datetime.now()

```

```

        thefishermates =
Fishermate.objects.filter(Q(first_fisher_id = thefisher) |
Q(second_fisher_id = thefisher))
        thefishermates =
thefishermates.filter(fisher_mate_state = 1)
        thefishermates_ones =
list(thefishermates.values("first_fisher_id"))
        thefishermates_seconds =
list(thefishermates.values("second_fisher_id"))
        thefishermates = list(set(thefishermates_ones +
thefishermates_seconds))
        if fishingactivities_state == 0:
            fishingactivities =
Fishingactivity.objects.prefetch_related("fishers").select_relat
ed("initiative_fisher_id", "fishingreservoir_id",
"fishingactivitycategory_id").filter(initiative_fisher_id__in =
thefishermates)
            fishingactivities = fishingactivities.order_by("-
fishing_planned_start_time")
        elif fishingactivities_state == 1:
            fishingactivities =
Fishingactivity.objects.prefetch_related("fishers").select_relat
ed("fishingreservoir_id",
"fishingactivitycategory_id").filter(initiative_fisher_id__in =
thefishermates)
            fishingactivities =
fishingactivities.filter(fishing_planned_start_time__gte =
fishingactivity_currenttime)
            fishingactivities = fishingactivities.order_by("-
fishing_planned_start_time")
        elif fishingactivities_state == 2:
            fishingactivities =
Fishingactivity.objects.prefetch_related("fishers").select_relat
ed("initiative_fisher_id", "fishingreservoir_id",
"fishingactivitycategory_id").filter(initiative_fisher_id__in =
thefishermates)
            fishingactivities =
fishingactivities.filter(fishing_planned_start_time__gte =
fishingactivity_currenttime)
            fishingactivities =
fishingactivities.order_by("fishing_planned_start_time")
        elif fishingactivities_state == 3:
            fishingactivities =
Fishingactivity.objects.prefetch_related("fishers").select_relat

```

```

ed("initiative_fisher_id", "fishingreservoir_id",
"fishingactivitycategory_id").filter(initiative_fisher_id__in =
thefishermates)
        fishingactivities =
fishingactivities.filter(fishing_planned_start_time__lt =
fishingactivity_currenttime)
        fishingactivities = fishingactivities.order_by("-
fishing_planned_start_time")
        elif fishingactivities_state == 4:
            fishingactivities =
Fishingactivity.objects.prefetch_related("fishers").select_relat
ed("initiative_fisher_id", "fishingreservoir_id",
"fishingactivitycategory_id").filter(initiative_fisher_id__in =
thefishermates)
            fishingactivities =
fishingactivities.filter(fishing_planned_start_time__lt =
fishingactivity_currenttime)
            fishingactivities =
fishingactivities.order_by("fishing_planned_start_time")
            fishingactivities_paginator =
Paginator(fishingactivities, 10)
            fishingactivities_requested_portion_number =
request.GET.get('page')
            fishingactivities =
fishingactivities_paginator.get_page(fishingactivities_requested
_portion_number)
            for fishingactivity in fishingactivities:
                fishingactivity_photos =
Fishingactivityphoto.objects.filter(fishingactivity_id =
fishingactivity)
                fishingactivity.photo = fishingactivity_photos[0]
                fishingactivity.fishers_number =
fishingactivity.fishers.count()
                fishingactivity_scores =
Fishingactivityscore.objects.filter(fishingactivity_id =
fishingactivity)
                fishingactivity_scores_number =
fishingactivity_scores.count()
                fishingactivity_observer_scores =
Fishingactivityscore.objects.filter(fishingactivity_id =
fishingactivity, fisher_id = thefisher)
                fishingactivity_observer_scores =
fishingactivity_observer_scores.order_by("-fishing_score_time")
                if fishingactivity_observer_scores:
                    fishingactivity.observer_score =
fishingactivity_observer_scores[0]

```

```

        else:
            fishingactivity.observer_score = -1
        return render(request,
            "fishmatesoft/fishermateactivities.html", {"fishingactivities":
            fishingactivities, "fisher": thefisher })

    # Перегляд запланованих активностей, пов'язаних з
    # риболовлею, в яких планують взяти участь знайомі
    def
fishermate_participant_planned_fishingactivities_view(request,
fishingactivities_state = 0):
    if request.user.is_authenticated:
        try:
            thefisher = Fisher.objects.get(fisher_user_name
            = request.user.username)
        except Fisher.DoesNotExist:
            thefisher = None
    else:
        thefisher = None
    if not thefisher:
        return render(request, "fishmatesoft/error.html",
            {"comment": "Не вдалось вас повністю ідентифікувати. Сталась
            помилка. Можливо дані рибалки було некоректно змінено"})
    import datetime
    fishingactivity_currenttime = datetime.datetime.now()
    thefishermates =
Fishermate.objects.filter(Q(first_fisher_id = thefisher) |
Q(second_fisher_id = thefisher))
    thefishermates =
thefishermates.filter(fisher_mate_state = 1)
    thefishermates_ones =
list(thefishermates.values("first_fisher_id"))
    thefishermates_seconds =
list(thefishermates.values("second_fisher_id"))
    thefishermates = list(set(thefishermates_ones +
thefishermates_seconds))
    if fishingactivities_state == 0:
        fishingactivities =
Fishingactivity.objects.prefetch_related("fishers").select_relat
ed("initiative_fisher_id", "fishingreservoir_id",
"fishingactivitycategory_id").filter(fishers__in =
thefishermates)
        fishingactivities = fishingactivities.order_by("-
fishing_planned_start_time")

```

```

        elif fishingactivities_state == 1:
            fishingactivities =
Fishingactivity.objects.prefetch_related("fishers").select_relat
ed("initiative_fisher_id", "fishingreservoir_id",
"fishingactivitycategory_id").filter(fishers__in =
thefishermates)
            fishingactivities =
fishingactivities.filter(fishing_planned_start_time__gte =
fishingactivity_currenttime)
            fishingactivities = fishingactivities.order_by("-
fishing_planned_start_time")
        elif fishingactivities_state == 2:
            fishingactivities =
Fishingactivity.objects.prefetch_related("fishers").select_relat
ed("initiative_fisher_id", "fishingreservoir_id",
"fishingactivitycategory_id").filter(fishers__in =
thefishermates)
            fishingactivities =
fishingactivities.filter(fishing_planned_start_time__gte =
fishingactivity_currenttime)
            fishingactivities =
fishingactivities.order_by("fishing_planned_start_time")
        elif fishingactivities_state == 3:
            fishingactivities =
Fishingactivity.objects.prefetch_related("fishers").select_relat
ed("initiative_fisher_id", "fishingreservoir_id",
"fishingactivitycategory_id").filter(fishers__in =
thefishermates)
            fishingactivities =
fishingactivities.filter(fishing_planned_start_time__lt =
fishingactivity_currenttime)
            fishingactivities = fishingactivities.order_by("-
fishing_planned_start_time")
        elif fishingactivities_state == 4:
            fishingactivities =
Fishingactivity.objects.prefetch_related("fishers").select_relat
ed("initiative_fisher_id", "fishingreservoir_id",
"fishingactivitycategory_id").filter(fishers__in =
thefishermates)
            fishingactivities =
fishingactivities.filter(fishing_planned_start_time__lt =
fishingactivity_currenttime)
            fishingactivities =
fishingactivities.order_by("fishing_planned_start_time")

```

```

        fishingactivities_paginator =
Paginator(fishingactivities, 10)
        fishingactivities_requested_portion_number =
request.GET.get('page')
        fishingactivities =
fishingactivities_paginator.get_page(fishingactivities_requested
_portion_number)
        for fishingactivity in fishingactivities:
            fishingactivity_photos =
Fishingactivityphoto.objects.filter(fishingactivity_id =
fishingactivity)
            fishingactivity.photo = fishingactivity_photos[0]
            fishingactivity.fishers_number =
fishingactivity.fishers.count()
            fishingactivity_scores =
Fishingactivityscore.objects.filter(fishingactivity_id =
fishingactivity)
            fishingactivity.scores_number =
fishingactivity_scores.count()
            fishingactivity_observer_scores =
Fishingactivityscore.objects.filter(fishingactivity_id =
fishingactivity, fisher_id = thefisher)
            fishingactivity_observer_scores =
fishingactivity_observer_scores.order_by("-fishing_score_time")
            if fishingactivity_observer_scores:
                fishingactivity.observer_score =
fishingactivity_observer_scores[0]
            else:
                fishingactivity.observer_score = -1
        return render(request,
"fishmatesoft/fishermatepartactivities.html",
{"fishingactivities": fishingactivities, "fisher": thefisher })

```

#Перегляд звітувань рибалки за досвідом активностей,
пов'язаних з риболовлю

```

def fisher_fishingactivityreports_view(request,
fisher_username = None, fishingactivreport_state = 0):
    if not fisher_username:
        if request.user.is_authenticated:
            try:
                thefisher =
Fisher.objects.get(fisher_user_name = request.user.username)
            except Fisher.DoesNotExist:
                thefisher = None
        else:
            thefisher = None

```

```

else:
    try:
        thefisher = Fisher.objects.get(fisher_user_name
= fisher_username)
    except Fisher.DoesNotExist:
        thefisher = None
    if not thefisher:
        return render(request, "fishmatesoft/error.html",
{"comment": "Такий рибалка відсутній, дані було надано
некоректно"})
    import datetime
    fishingactivity_currenttime = datetime.datetime.now()
    if fishingactivreport_state == 0:
        fishingactivity_reports =
Fishingactivityreport.objects.select_related("fishingactivity_id
", "fisher_id", "fishingactivity_id__fishingreservoir_id",
"fishingactivity_id__fishingactivitycategory_id").filter(fisher_
id = thefisher)
        fishingactivity_reports =
fishingactivity_reports.order_by("-
fishing_report_creation_time")
    elif fishingactivreport_state == 1:
        fishingactivity_reports =
Fishingactivityreport.objects.select_related("fishingactivity_id
", "fisher_id", "fishingactivity_id__fishingreservoir_id",
"fishingactivity_id__fishingactivitycategory_id").filter(fisher_
id = thefisher)
        fishingactivity_reports =
fishingactivity_reports.filter(fishing_report_creation_time__gte
= fishingactivity_currenttime)
        fishingactivity_reports =
fishingactivity_reports.order_by("-
fishing_report_creation_time")
    elif fishingactivreport_state == 2:
        fishingactivity_reports =
Fishingactivityreport.objects.select_related("fishingactivity_id
", "fisher_id", "fishingactivity_id__fishingreservoir_id",
"fishingactivity_id__fishingactivitycategory_id").filter(fisher_
id = thefisher)
        fishingactivity_reports =
fishingactivity_reports.filter(fishing_report_creation_time__gte
= fishingactivity_currenttime)
        fishingactivity_reports =
fishingactivity_reports.order_by("fishing_report_creation_time")

```

```

        elif fishingactivreport_state == 3:
            fishingactivity_reports =
Fishingactivityreport.objects.select_related("fishingactivity_id
", "fisher_id", "fishingactivity_id__fishingreservoir_id",
"fishingactivity_id__fishingactivitycategory_id").filter(fisher_
id = thefisher)

            fishingactivity_reports =
fishingactivity_reports.filter(fishing_report_creation_time__lt
= fishingactivity_currenttime)

            fishingactivity_reports =
fishingactivity_reports.order_by("-
fishing_report_creation_time")
        elif fishingactivreport_state == 4:
            fishingactivity_reports =
Fishingactivityreport.objects.select_related("fishingactivity_id
", "fisher_id", "fishingactivity_id__fishingreservoir_id",
"fishingactivity_id__fishingactivitycategory_id").filter(fisher_
id = thefisher)

            fishingactivity_reports =
fishingactivity_reports.filter(fishing_report_creation_time__lt
= fishingactivity_currenttime)

            fishingactivity_reports =
fishingactivity_reports.order_by("fishing_report_creation_time")
        elif fishingactivreport_state == 5:
            fishingactivity_reports =
Fishingactivityreport.objects.select_related("fishingactivity_id
", "fisher_id", "fishingactivity_id__fishingreservoir_id",
"fishingactivity_id__fishingactivitycategory_id").filter(fisher_
id = thefisher)

            fishingactivity_reports =
fishingactivity_reports.filter(fishingactivity_id__fishing_plann
ed_start_time__gte = fishingactivity_currenttime)

            fishingactivity_reports =
fishingactivity_reports.order_by("-
fishing_report_creation_time")
        elif fishingactivreport_state == 6:
            fishingactivity_reports =
Fishingactivityreport.objects.select_related("fishingactivity_id
", "fisher_id", "fishingactivity_id__fishingreservoir_id",
"fishingactivity_id__fishingactivitycategory_id").filter(fisher_
id = thefisher)

            fishingactivity_reports =
fishingactivity_reports.filter(fishingactivity_id__fishing_plann
ed_start_time__gte = fishingactivity_currenttime)

```

```

        fishingactivity_reports =
fishingactivity_reports.order_by("fishing_report_creation_time")
        elif fishingactivereport_state == 7:
            fishingactivity_reports =
Fishingactivityreport.objects.select_related("fishingactivity_id
", "fisher_id", "fishingactivity_id__fishingreservoir_id",
"fishingactivity_id__fishingactivitycategory_id").filter(fisher_
id = thefisher)
            fishingactivity_reports =
fishingactivity_reports.filter(fishingactivity_id__fishing_plann
ed_start_time__lt = fishingactivity_currenttime)
            fishingactivity_reports =
fishingactivity_reports.order_by("-
fishing_report_creation_time")
            elif fishingactivereport_state == 8:
                fishingactivity_reports =
Fishingactivityreport.objects.select_related("fishingactivity_id
", "fisher_id", "fishingactivity_id__fishingreservoir_id",
"fishingactivity_id__fishingactivitycategory_id").filter(fisher_
id = thefisher)
                fishingactivity_reports =
fishingactivity_reports.filter(fishingactivity_id__fishing_plann
ed_start_time__lt = fishingactivity_currenttime)
                fishingactivity_reports =
fishingactivity_reports.order_by("fishing_report_creation_time")
                fishingactivityreports_paginator =
Paginator(fishingactivity_reports, 10)
                fishingactivityreports_requested_portion_number =
request.GET.get('page')
                fishingactivity_reports =
fishingactivityreports.get_page(fishingactivityreports_requested
_portion_number)
                for fishingactivityreport in fishingactivity_reports:
                    fishingactivityreport_photos =
Fishingactivityreportphoto.objects.filter(fishingactivityreport_
id = fishingactivityreport)
                    fishingactivityreport.photo =
fishingactivityreport_photos[0]
                    fishingactivityreport_comments =
Fishingactivityreportcomment.objects.filter(fishingactivityrepor
t_id = fishingactivityreport)
                    fishingactivityreport.comments_number =
fishingactivityreport_comments.count()
                    fishingactivityreport_comments =
fishingactivityreport_comments.filter(fisher_id = thefisher)

```

```

        fishingactivityreport_positive_reactions =
Fishingactivityreportreaction.objects.filter(fishingactivityrepo
rt_id = fishingactivityreport, fishing_reaction_type = 0)
        fishingactivityreport_observe_positive_reactions =
fishingactivityreport_positive_reactions.filter(fisher_id =
thefisher)
        if
fishingactivityreport_observe_positive_reactions:
            fishingactivityreport.posobserverreaction = True
        else:
            fishingactivityreport.posobserverreaction =
False
        if fishingactivityreport_positive_reactions:
            fishingactivityreport.posreaction_number =
fishingactivityreport_positive_reactions.count()
        else:
            fishingactivityreport.posreaction_number = 0
        fishingactivityreport_negative_reactions =
Fishingactivityreportreaction.objects.filter(fishingactivityrepo
rt_id = fishingactivityreport, fishing_reaction_type = 1)
        fishingactivityreport_observe_negative_reactions =
fishingactivityreport_negative_reactions.filter(fisher_id =
thefisher)
        if
fishingactivityreport_observe_negative_reactions:
            fishingactivityreport.negobserverreaction = True
        else:
            fishingactivityreport.negobserverreaction =
False
        if fishingactivityreport_negative_reactions:
            fishingactivityreport.negreaction_number =
fishingactivityreport_negative_reactions.count()
        else:
            fishingactivityreport.negreaction_number = 0
    return render(request,
"fishmatesoft/fisherreports.html", {"fishingactreports":
fishingactivity_reports, "fisher": thefisher })

# Перегляд звітувань знайомих рибалок за досвідом
активностей, пов'язаних з риболовлю
    def fishermate_fishingactivityreports_view(request,
fishingactivreport_state = 0):
        if request.user.is_authenticated:
            try:
                thefisher = Fisher.objects.get(fisher_user_name
= request.user.username)
            except Fisher.DoesNotExist:

```

```

        thefisher = None
    else:
        thefisher = None
    if not thefisher:
        return render(request, "fishmatesoft/error.html",
{"comment": "Не вдалось вас повністю ідентифікувати. Сталась помилка. Можливо дані рибалки було некоректно змінено"})
        import datetime
        fishingactivity_currenttime = datetime.datetime.now()
        thefishermates =
Fishermate.objects.filter(Q(first_fisher_id = thefisher) |
Q(second_fisher_id = thefisher))
        thefishermates =
thefishermates.filter(fisher_mate_state = 1)
        thefishermates_ones =
list(thefishermates.values("first_fisher_id"))
        thefishermates_seconds =
list(thefishermates.values("second_fisher_id"))
        thefishermates = list(set(thefishermates_ones +
thefishermates_seconds))
        if fishingactivreport_state == 0:
            fishingactivity_reports =
Fishingactivityreport.objects.select_related("fishingactivity_id
", "fisher_id", "fishingactivity_id__fishingreservoir_id",
"fishingactivity_id__fishingactivitycategory_id").filter(fisher_
id__in = thefishermates)
            fishingactivity_reports =
fishingactivity_reports.order_by("-
fishing_report_creation_time")
            elif fishingactivreport_state == 1:
                fishingactivity_reports =
Fishingactivityreport.objects.select_related("fishingactivity_id
", "fisher_id", "fishingactivity_id__fishingreservoir_id",
"fishingactivity_id__fishingactivitycategory_id").filter(fisher_
id__in = thefishermates)
                fishingactivity_reports =
fishingactivity_reports.filter(fishing_report_creation_time__gte
= fishingactivity_currenttime)
                fishingactivity_reports =
fishingactivity_reports.order_by("-
fishing_report_creation_time")
            elif fishingactivreport_state == 2:
                fishingactivity_reports =
Fishingactivityreport.objects.select_related("fishingactivity_id

```



```

        fishingactivity_reports =
Fishingactivityreport.objects.select_related("fishingactivity_id
", "fisher_id", "fishingactivity_id__fishingreservoir_id",
"fishingactivity_id__fishingactivitycategory_id").filter(fisher_
id__in = thefishermates)
        fishingactivity_reports =
fishingactivity_reports.filter(fishingactivity_id__fishing_plann
ed_start_time__gte = fishingactivity_currenttime)
        fishingactivity_reports =
fishingactivity_reports.order_by("fishing_report_creation_time")
        elif fishingactivereport_state == 7:
            fishingactivity_reports =
Fishingactivityreport.objects.select_related("fishingactivity_id
", "fisher_id", "fishingactivity_id__fishingreservoir_id",
"fishingactivity_id__fishingactivitycategory_id").filter(fisher_
id__in = thefishermates)
            fishingactivity_reports =
fishingactivity_reports.filter(fishingactivity_id__fishing_plann
ed_start_time__lt = fishingactivity_currenttime)
            fishingactivity_reports =
fishingactivity_reports.order_by("-
fishing_report_creation_time")
            elif fishingactivereport_state == 8:
                fishingactivity_reports =
Fishingactivityreport.objects.select_related("fishingactivity_id
", "fisher_id", "fishingactivity_id__fishingreservoir_id",
"fishingactivity_id__fishingactivitycategory_id").filter(fisher_
id__in = thefishermates)
                fishingactivity_reports =
fishingactivity_reports.filter(fishingactivity_id__fishing_plann
ed_start_time__lt = fishingactivity_currenttime)
                fishingactivity_reports =
fishingactivity_reports.order_by("fishing_report_creation_time")
                fishingactivityreports_paginator =
Paginator(fishingactivity_reports, 10)
                fishingactivityreports_requested_portion_number =
request.GET.get('page')
                fishingactivity_reports =
fishingactivityreports.get_page(fishingactivityreports_requested
_portion_number)
                for fishingactivityreport in fishingactivity_reports:
                    fishingactivityreport_photos =
Fishingactivityreportphoto.objects.filter(fishingactivityreport_
id = fishingactivityreport)

```

```

        fishingactivityreport.photo =
fishingactivityreport_photos[0]
        fishingactivityreport_comments =
Fishingactivityreportcomment.objects.filter(fishingactivityrepor
t_id = fishingactivityreport)
        fishingactivityreport.comments_number =
fishingactivityreport_comments.count()
        fishingactivityreport_comments =
fishingactivityreport_comments.filter(fisher_id = thefisher)
        fishingactivityreport_positive_reactions =
Fishingactivityreportreaction.objects.filter(fishingactivityrepo
rt_id = fishingactivityreport, fishing_reaction_type = 0)
        fishingactivityreport_observe_positive_reactions =
fishingactivityreport_positive_reactions.filter(fisher_id =
thefisher)
        if
fishingactivityreport_observe_positive_reactions:
            fishingactivityreport.posobserverreaction = True
        else:
            fishingactivityreport.posobserverreaction =
False
        if fishingactivityreport_positive_reactions:
            fishingactivityreport.posreaction_number =
fishingactivityreport_positive_reactions.count()
        else:
            fishingactivityreport.posreaction_number = 0
        fishingactivityreport_negative_reactions =
Fishingactivityreportreaction.objects.filter(fishingactivityrepo
rt_id = fishingactivityreport, fishing_reaction_type = 1)
        fishingactivityreport_observe_negative_reactions =
fishingactivityreport_negative_reactions.filter(fisher_id =
thefisher)
        if
fishingactivityreport_observe_negative_reactions:
            fishingactivityreport.negobserverreaction = True
        else:
            fishingactivityreport.negobserverreaction =
False
        if fishingactivityreport_negative_reactions:
            fishingactivityreport.negreaction_number =
fishingactivityreport_negative_reactions.count()
        else:
            fishingactivityreport.negreaction_number = 0
    return render(request,
"fishmatesoft/fishermatereports.html", {"fishingactreports":
fishingactivity_reports, "fisher": thefisher })

```

```

#Перегляд звітувань за досвідом активностей, пов'язаних з
риболовлю, що отримали власні реакції
def fisher_reacted_fishingactivityreports_view(request,
reaction_type, fishingactivreport_state = 0):
    if request.user.is_authenticated:
        try:
            thefisher = Fisher.objects.get(fisher_user_name
= request.user.username)
        except Fisher.DoesNotExist:
            thefisher = None
    else:
        thefisher = None
    if not thefisher:
        return render(request, "fishmatesoft/error.html",
{"comment": "Не вдалось вас повністю ідентифікувати. Сталась
помилка. Можливо дані рибалки було некоректно змінено"})
    import datetime
    fishingactivity_currenttime = datetime.datetime.now()
    fishingactivity_report_reactions =
Fishingactivityreportreaction.objects.select_related("fishingact
ivityreport_id").filter(fishing_reaction_type = reaction_type,
fisher_id = thefisher)
    fishingactivity_report_reactions =
fishingactivity_report_reactions.order_by("-id")
    fishingactivity_report_reactions =
fishingactivity_report_reactions.values("fishingactivityreport_i
d__id"))
    if fishingactivreport_state == 0:
        fishingactivity_reports =
Fishingactivityreport.objects.select_related("fishingactivity_id
", "fisher_id", "fishingactivity_id__fishingreservoir_id",
"fishingactivity_id__fishingactivitycategory_id").filter(id__in
= fishingactivity_report_reactions)
        fishingactivity_reports =
fishingactivity_reports.order_by("-
fishing_report_creation_time")
    elif fishingactivreport_state == 1:
        fishingactivity_reports =
Fishingactivityreport.objects.select_related("fishingactivity_id
", "fisher_id", "fishingactivity_id__fishingreservoir_id",
"fishingactivity_id__fishingactivitycategory_id").filter(id__in
= fishingactivity_report_reactions)

```

```

        fishingactivity_reports =
fishingactivity_reports.filter(fishing_report_creation_time__gte
= fishingactivity_currenttime)
        fishingactivity_reports =
fishingactivity_reports.order_by("-
fishing_report_creation_time")
        elif fishingactivreport_state == 2:
            fishingactivity_reports =
Fishingactivityreport.objects.select_related("fishingactivity_id
", "fisher_id", "fishingactivity_id__fishingreservoir_id",
"fishingactivity_id__fishingactivitycategory_id").filter(id__in
= fishingactivity_report_reactions)
            fishingactivity_reports =
fishingactivity_reports.filter(fishing_report_creation_time__gte
= fishingactivity_currenttime)
            fishingactivity_reports =
fishingactivity_reports.order_by("fishing_report_creation_time")
            elif fishingactivreport_state == 3:
                fishingactivity_reports =
Fishingactivityreport.objects.select_related("fishingactivity_id
", "fisher_id", "fishingactivity_id__fishingreservoir_id",
"fishingactivity_id__fishingactivitycategory_id").filter(id__in
= fishingactivity_report_reactions)
                fishingactivity_reports =
fishingactivity_reports.filter(fishing_report_creation_time__lt
= fishingactivity_currenttime)
                fishingactivity_reports =
fishingactivity_reports.order_by("-
fishing_report_creation_time")
                elif fishingactivreport_state == 4:
                    fishingactivity_reports =
Fishingactivityreport.objects.select_related("fishingactivity_id
", "fisher_id", "fishingactivity_id__fishingreservoir_id",
"fishingactivity_id__fishingactivitycategory_id").filter(id__in
= fishingactivity_report_reactions)
                    fishingactivity_reports =
fishingactivity_reports.filter(fishing_report_creation_time__lt
= fishingactivity_currenttime)
                    fishingactivity_reports =
fishingactivity_reports.order_by("fishing_report_creation_time")
                    elif fishingactivreport_state == 5:
                        fishingactivity_reports =
Fishingactivityreport.objects.select_related("fishingactivity_id
", "fisher_id", "fishingactivity_id__fishingreservoir_id",

```

```

"fishingactivity_id__fishingactivitycategory_id").filter(id__in
= fishingactivity_report_reactions)
        fishingactivity_reports =
fishingactivity_reports.filter(fishingactivity_id__fishing_plann
ed_start_time__gte = fishingactivity_currenttime)
        fishingactivity_reports =
fishingactivity_reports.order_by("-
fishing_report_creation_time")
        elif fishingactivreport_state == 6:
            fishingactivity_reports =
Fishingactivityreport.objects.select_related("fishingactivity_id
", "fisher_id", "fishingactivity_id__fishingreservoir_id",
"fishingactivity_id__fishingactivitycategory_id").filter(id__in
= fishingactivity_report_reactions)
            fishingactivity_reports =
fishingactivity_reports.filter(fishingactivity_id__fishing_plann
ed_start_time__gte = fishingactivity_currenttime)
            fishingactivity_reports =
fishingactivity_reports.order_by("fishing_report_creation_time")
            elif fishingactivreport_state == 7:
                fishingactivity_reports =
Fishingactivityreport.objects.select_related("fishingactivity_id
", "fisher_id", "fishingactivity_id__fishingreservoir_id",
"fishingactivity_id__fishingactivitycategory_id").filter(id__in
= fishingactivity_report_reactions)
                fishingactivity_reports =
fishingactivity_reports.filter(fishingactivity_id__fishing_plann
ed_start_time__lt = fishingactivity_currenttime)
                fishingactivity_reports =
fishingactivity_reports.order_by("-
fishing_report_creation_time")
                elif fishingactivreport_state == 8:
                    fishingactivity_reports =
Fishingactivityreport.objects.select_related("fishingactivity_id
", "fisher_id", "fishingactivity_id__fishingreservoir_id",
"fishingactivity_id__fishingactivitycategory_id").filter(id__in
= fishingactivity_report_reactions)
                    fishingactivity_reports =
fishingactivity_reports.filter(fishingactivity_id__fishing_plann
ed_start_time__lt = fishingactivity_currenttime)
                    fishingactivity_reports =
fishingactivity_reports.order_by("fishing_report_creation_time")
                    fishingactivityreports_paginator =
Paginator(fishingactivity_reports, 10)

```

```

        fishingactivityreports_requested_portion_number =
request.GET.get('page')
        fishingactivity_reports =
fishingactivityreports.get_page(fishingactivityreports_requested
_portion_number)
        for fishingactivityreport in fishingactivity_reports:
            fishingactivityreport_photos =
Fishingactivityreportphoto.objects.filter(fishingactivityreport_
id = fishingactivityreport)
            fishingactivityreport.photo =
fishingactivityreport_photos[0]
            fishingactivityreport_comments =
Fishingactivityreportcomment.objects.filter(fishingactivityrepor
t_id = fishingactivityreport)
            fishingactivityreport.comments_number =
fishingactivityreport_comments.count()
            fishingactivityreport_comments =
fishingactivityreport_comments.filter(fisher_id = thefisher)
            fishingactivityreport_positive_reactions =
Fishingactivityreportreaction.objects.filter(fishingactivityrepo
rt_id = fishingactivityreport, fishing_reaction_type = 0)
            fishingactivityreport_observe_positive_reactions =
fishingactivityreport_positive_reactions.filter(fisher_id =
thefisher)
            if
fishingactivityreport_observe_positive_reactions:
                fishingactivityreport.posobserverreaction = True
            else:
                fishingactivityreport.posobserverreaction =
False
            if fishingactivityreport_positive_reactions:
                fishingactivityreport.posreaction_number =
fishingactivityreport_positive_reactions.count()
            else:
                fishingactivityreport.posreaction_number = 0
            fishingactivityreport_negative_reactions =
Fishingactivityreportreaction.objects.filter(fishingactivityrepo
rt_id = fishingactivityreport, fishing_reaction_type = 1)
            fishingactivityreport_observe_negative_reactions =
fishingactivityreport_negative_reactions.filter(fisher_id =
thefisher)
            if
fishingactivityreport_observe_negative_reactions:
                fishingactivityreport.negobserverreaction = True
            else:
                fishingactivityreport.negobserverreaction =
False
            if fishingactivityreport_negative_reactions:

```

```
        fishingactivityreport.negative_reactions.count()
    else:
        fishingactivityreport.negative_reactions.count() = 0
    return render(request,
        "fishmatesoft/reactfishreports.html", {"fishingactreports":
        fishingactivity_reports, "fisher": thefisher })
```

ДОДАТОК В
Слайди презентації

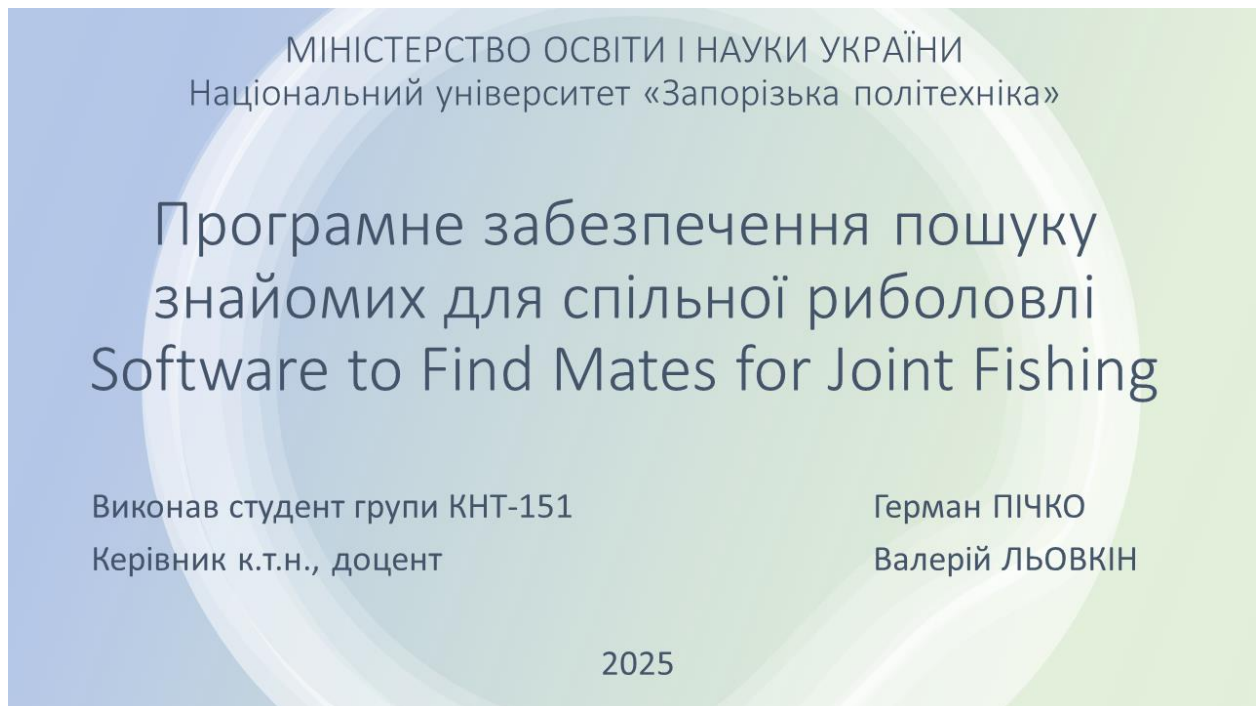


Рисунок В.1 – Початковий слайд

- Об'єкт роботи – процес пошуку знайомих для спільної риболовлі.
- Предмет роботи – програмне забезпечення пошуку знайомих для спільної риболовлі.
- Мета роботи – розробити програмне забезпечення пошуку знайомих для спільної риболовлі на основі інформування про власний досвід риболовлі.

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

Завдання роботи

- аналіз проблеми пошуку знайомих для спільної риболовлі;
- проектування програми пошуку знайомих для спільної риболовлі;
- розробка програми пошуку знайомих для спільної риболовлі.

3

Рисунок В.3 – Завдання роботи

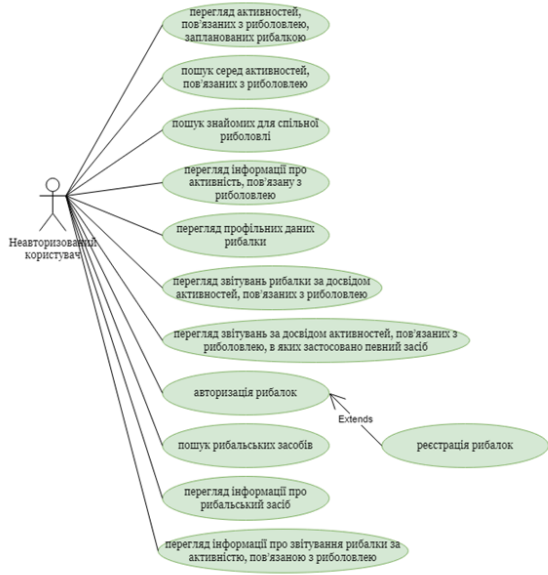
Результати порівняння програм, застосовуваних для пошуку знайомих для спільної риболовлі

Критерій	WinFish	Відпочинок	FishMateSoft
Звітування про активності	Так, без реакцій	Так, з коментарями	З наданням інформації про використані засоби для риболовлі, з реакціями, коментарями
Установлення знайомств серед рибалок через сам вебзастосунок	Ні	Ні	Так
Пошук серед ко-ристувачів знайомих для спільної риболовлі	Ні	Ні	Так
Визначення запланованих активностей користувачами	Ні, тільки адміністратор	Ні, тільки адміністратор	Так

4

Рисунок В.4 – Результати порівняння програм, застосовуваних для пошуку знайомих для спільної риболовлі

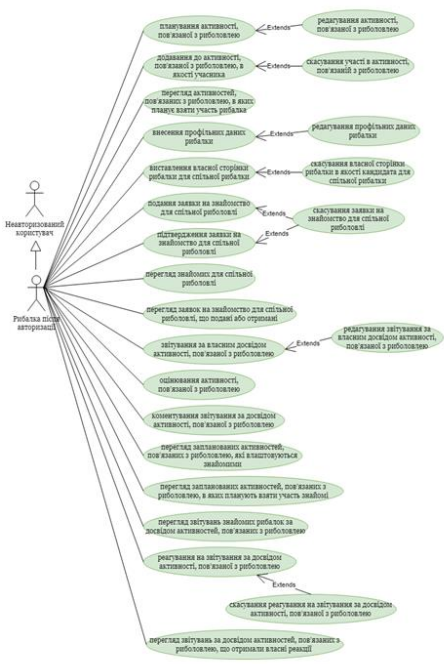
Діаграма варіантів використання програми пошуку знайомих для спільної риболовлі неавторизованим користувачем



5

Рисунок В.5 – Діаграма варіантів використання програми пошуку знайомих для спільної риболовлі неавторизованим користувачем

Діаграма варіантів використання програми пошуку знайомих для спільної риболовлі рибалкою після авторизації



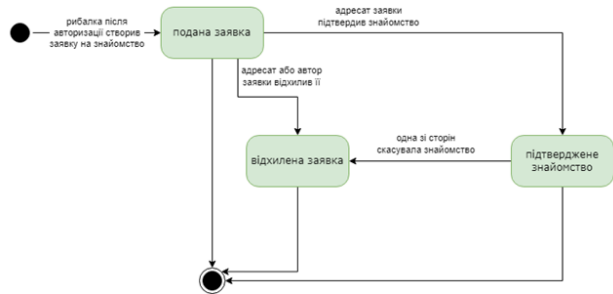
6

Рисунок В.6 – Діаграма варіантів використання програми пошуку знайомих для спільної риболовлі рибалкою після авторизації

Схема бази даних програми пошуку знайомих для спільної риболовлі

Рисунок В.8 – Схема бази даних програми пошуку знайомих для спільної риболовлі

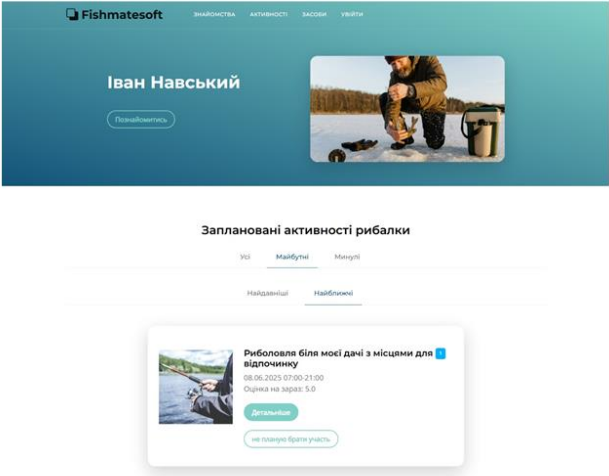
Діаграма станів знайомства для спільної риболовлі



9

Рисунок В.9 – Діаграма станів знайомства для спільної риболовлі

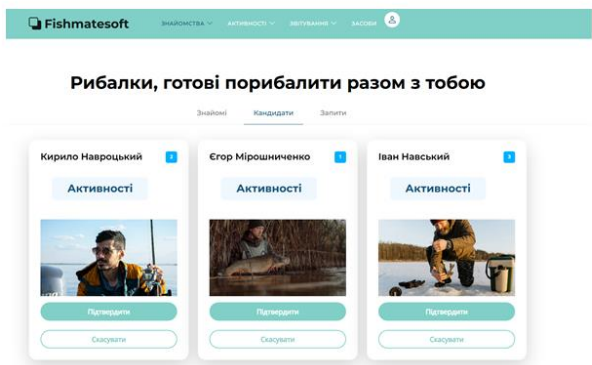
Інтерфейс програми: заплановані активності, пов’язані з риболовлею, у рибалки



10

Рисунок В.10 – Інтерфейс програми: заплановані активності, пов’язані з риболовлею, у рибалки

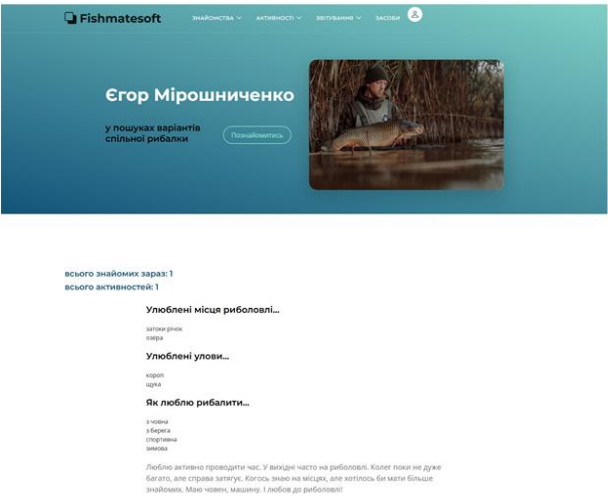
Інтерфейс програми: сторінка зі знайомими для спільної риболовлі та запитами



11

Рисунок В.11 – Інтерфейс програми: сторінка зі знайомими для спільної риболовлі та запитами

Інтерфейс програми: сторінка рибалки в програмі



12

Рисунок В.12 – Інтерфейс програми: сторінка рибалки в програмі

Чек-лист тестування програми пошуку знайомих для спільної риболовлі

Функція	Неавторизований користувач	Рибалка після авторизації
Перегляд активностей, пов'язаних з риболовлею, запланованих рибалкою	+	+
Пошук серед активностей, пов'язаних з риболовлею	+	+
Пошук знайомих для спільної риболовлі	+	+
Перегляд інформації про активність, пов'язану з риболовлею	+	+
Перегляд профільних даних рибалки	+	+
Перегляд звітувань рибалки за досвідом активностей, пов'язаних з риболовлею	+	+
Перегляд звітувань за досвідом активностей, пов'язаних з риболовлею, в яких застосовано певний засіб	+	+
Авторизація рибалки	+	НЗ
Пошук рибальських засобів	+	+
Реєстрація рибалки	+	НЗ
Перегляд інформації про рибальський засіб	+	+
Перегляд інформації про звітування рибалки за активністю, пов'язану з риболовлею	+	+
Планування активності, пов'язаної з риболовлею	НЗ	+
Додавання до активності, пов'язаної з риболовлею, в якості учасника	НЗ	+
Перегляд активностей, пов'язаних з риболовлею, в яких планує взяти участь рибалка	НЗ	+
Внесення профільних даних рибалки	НЗ	+
Виставлення власної сторінки рибалки для спільної риболовлі	НЗ	+
Подання заявки на знайомство для спільної риболовлі	НЗ	+
Підтвердження заявки на знайомство для спільної риболовлі	НЗ	+
Перегляд знайомих для спільної риболовлі	НЗ	+

Функція	Неавторизований користувач	Рибалка після авторизації
Перегляд заявок на знайомство для спільної риболовлі, що подані або отримані	НЗ	+
Звітування за власним досвідом активності, пов'язаної з риболовлею	НЗ	+
Оцінювання активності, пов'язаної з риболовлею	НЗ	+
Коментування звітування за досвідом активності, пов'язаної з риболовлею	НЗ	+
Перегляд запланованих активностей, пов'язаних з риболовлею, які влаштовуються знайомими	НЗ	+
Перегляд запланованих активностей, пов'язаних з риболовлею, в яких планують взяти участь знайомі	НЗ	+
Перегляд звітувань знайомих рибалок за досвідом активностей, пов'язаних з риболовлею	НЗ	+
Реагування на звітування за досвідом активності, пов'язаної з риболовлею	НЗ	+
Перегляд звітувань за досвідом активностей, пов'язаних з риболовлею, що отримали власні реакції	НЗ	+
Скасування реагування на звітування за досвідом активності, пов'язаної з риболовлею	НЗ	+
Редагування звітування за власним досвідом активності, пов'язаної з риболовлею	НЗ	+
Скасування заявки на знайомство для спільної риболовлі	НЗ	+
Скасування власної сторінки рибалки в якості кандидата для спільної риболовлі	НЗ	+
Редагування профільних даних рибалки	НЗ	+
Скасування участі в активності, пов'язаній з риболовлею	НЗ	+
Редагування активності, пов'язаної з риболовлею	НЗ	+

13

Рисунок В.13 – Чек-лист тестування програми пошуку знайомих для спільної риболовлі

Висновки

- У роботі шляхом успішного виконання всіх поставлених завдань було досягнуто мету розробити програмне забезпечення пошуку знайомих для спільної риболовлі на основі інформування про власний досвід риболовлі.
- Застосування розробленої програми для пошуку знайомих для спільної риболовлі базується не тільки на створенні відповідного пошуку за параметрами, але і на перегляді того досвіду, який потенційні знайомі раніше мали під час риболовлі, тих підходів до риболовлі, яким віддає переваги рибалка. Для цього програма надає ряд спрямованих інструментів, що полегшують ці процеси та сприяють прийняттю рішень, які можуть призвести до позитивного досвіду спільної риболовлі. Усе це дозволяє використовувати її для створення платформи для комунікації між рибалками та планування спільних активностей з забезпеченням розвитку спільноти рибалок, рибальських клубів.

14

Рисунок В.14 – Висновки