

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проекту (роботи)

бакалавра

(ступінь вищої освіти)

на тему ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ПОШУКУ І

(назва теми)

РЕЙТИНГУВАННЯ ЕКСКУРСОВОДІВ

SOFTWARE TO SEARCH AND RATE THE TOUR GUIDES

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

Спеціальності 121 Інженерія програмного

(код і найменування спеціальності)

забезпечення

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

КОВАЛЬ Є.В.

(ПРИЗВИЩЕ та ініціали)

Керівник КАЗУРОВА А.Є.

(ПРИЗВИЩЕ та ініціали)

Рецензент ПОЛЯКОВ М.О.

(ПРИЗВИЩЕ та ініціали)

2023__

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет ФКНТ

Кафедра програмних засобів

Ступінь вищої освіти бакалавр

Спеціальність 121 Інженерія програмного забезпечення

(код і найменування)

Освітня програма (спеціалізація) Інженерія програмного забезпечення

(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
Сергій СУББОТІН

« » 2023 року

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

КОВАЛЯ Єгора Володимировича

(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Програмне забезпечення пошуку і рейтингування експурсоводів. Software to Search and Rate the Tour Guides

керівник проєкту (роботи) к.т.н., доцент КАЗУРОВА Аліна Євгенівна,
(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

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

2. Строк подання студентом проєкту (роботи) 7 червня 2023 року

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

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

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

Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	ПРИЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	КАЗУРОВА А.Є., доцент		
Нормоконтроль	КАМІНСЬКА Ж.К., асистент		

7. Дата видачі завдання « 7 » квітня 2023 року.

КАЛЕНДАРНИЙ ПЛАН

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

Студент(ка)

(підпис)

Єгор КОВАЛЬ

(Ім'я ПРИЗВИЩЕ)

Керівник проєкту (роботи)

(підпис)

Аліна КАЗУРОВА

(Ім'я ПРИЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 95 с., 27 рис., 2 табл., 3 дод., 10 джерел.

ЕКСКУРСІЯ, ЕКСКУРСОВОД, ВІДВІДУВАЧ, БРОНЮВАННЯ, ОЦІНЮВАННЯ, БАЗА ДАНИХ.

Об'єкт роботи – процес підтримки проведення екскурсій.

Предмет роботи – програмне забезпечення пошуку і рейтингування екскурсоводів.

Мета роботи – розробити програмне забезпечення пошуку і рейтингування екскурсоводів для підвищення зручності роботи екскурсоводів та відвідувачів екскурсій стосовно взаємодії між ними.

Матеріали, методи та технічні засоби: мова python з підтримкою веброзробки через мови HTML, CSS, javascript, система для роботи з базами даних MySQL.

Результати. Усі завдання на роботу було виконано і представлено основні результати за кожним завданням у розділах пояснювальної записки.

Висновки. Програма надає екскурсоводу можливість формувати представлення своїх екскурсій, визначати дати проведення екскурсій, керувати внесеними даними, забезпечувати його можливістю розподілу місць на екскурсії, порівнювати враження різних відвідувачів від різних власних екскурсій, переглядати різноманітні рейтинги і відповідну позицію власної пропозиції у них. Відвідувачу програма надає можливість шукати, ознайомлюватися з варіантами існуючих екскурсій, обирати з них ті, які він вважає за потрібне, бронювати на них місця, створювати власні запити з проханням організувати для них персональну екскурсію у певному місті.

Галузь використання. Проведення екскурсій індивідуальними екскурсоводами, як персональних екскурсій, так і для груп відвідувачів.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 95 pages, 27 figures, 2 tables, 3 appendixes, 10 sources.

EXCURSION, TOUR GUIDE, VISITOR, BOOKING, RATING, DATABASE.

The object of work is the process of supporting excursions.

The subject of the work is software for searching and rating tour guides.

The purpose of the work is to develop software for the search and rating of tour guides to increase the convenience of the work of tour guides and tour visitors in terms of interaction between them.

Materials, methods and technical means: python language with support for web development through HTML, CSS, javascript languages, a system for working with MySQL databases.

The results. All tasks for the work were completed and the main results for each task were presented in the sections of the explanatory note.

Conclusions. The program provides the tour guide with the opportunity to create a presentation of his tours, determine the dates of the tours, manage the entered data, provide him with the possibility of allocating places on tours, compare the impressions of different visitors from different tours of his own, view various ratings and the corresponding position of his own offer in them. The program gives the visitor the opportunity to search, familiarize himself with the options of existing excursions, choose from them the ones he considers necessary, reserve places for them, create his own requests with a request to organize a personal excursion for them in a certain city.

Field of use. Conducting tours by individual tour guides, both for personal tours and for groups of visitors.

ЗМІСТ

	С.
Перелік скорочень та умовних познач.....	7
Вступ.....	8
1 Аналіз предметної області та постановка завдань дослідження.....	9
1.1 Наявне програмне забезпечення пошуку і рейтингування екскурсоводів.....	9
1.2 Порівняння визначених аналогів.....	14
1.3 Висновки за розділом 1	15
2 Проєктування програми.....	16
2.1 Вибір мови веброзробки.....	16
2.2 Вимоги до програми	17
2.3 Проєктування структури бази даних.....	21
2.4 Висновки за розділом 2	33
3 Розробка програми.....	34
3.1 Порядок функціонування програми.....	34
3.2 Реалізація функцій програмної системи.....	35
3.3 Висновки за розділом 3	42
4 Експлуатація, тестування та експериментальне дослідження програми	43
4.1 Призначення програми	43
4.2 Умови виконання програми	43
4.3 Принципи експлуатації та тестування програми.....	44
4.4 Висновки за розділом 4	50
Висновки	51
Перелік джерел посилання	53
Додаток А Технічне завдання	54
Додаток Б Текст програми	59
Додаток В Слайди презентації.....	89

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

БД – база даних;

ПЗ – програмне забезпечення.

ВСТУП

Туристична галузь базується не тільки на роботі туристичних агентств, але і індивідуальних працівників, зокрема екскурсоводів. Екскурсія – один з найкращих способів дізнатися про місто більше або познайомитись з ним. Екскурсії відбуваються на різну тематику, але загалом проведення екскурсій місцевими ентузіастами зокрема останнім часом активно привертає увагу багатьох туристів. Нерідко такі екскурсії містять більш особливу інформацію, є більш персоналізованими, надають новий погляд, який у масових екскурсіях туристичних агентств, які постійно повторюються за одним і тим самим сценарієм, не представлений.

Робота індивідуальних екскурсоводів – це фактично підсектор світового ринку туризму, але до того ж цей підсектор швидко розвивається: так за оцінками на 2019 рік загальна вартість цього підсектору склала 783,5 мільйонів доларів США, до того ж спорстерігалось значне збільшення за останні роки, а фактичні прогнози щодо розвитку визначили, що очікується подальше збільшення обсягів на 12,1% з 2021 до 2026 року [1]. Також потрібно відзначити, що саме Європа має найбільшу частку цього підсектору, охоплюючи 38 % від нього, а загалом туризм, який пов'язаний з послугами індивідуальних екскурсоводів, оцінювався в 566 мільярдів доларів США [1].

За підходу до екскурсій на основі використання роботи індивідуальних екскурсоводів виникає проблема: відвідувачі екскурсій є окремими особами, екскурсоводи також ніяким чином не об'єднані, тому як в багатьох сучасних рішеннях потрібна програмна система, яка дозволить об'єднати їх разом, щоб екскурсоводи могли сформулювати пропозицію, а відвідувачі – знайти ту пропозицію, яка найкраще відповідає їхнім інтересам. Саме таке програмне забезпечення (ПЗ), до того ж розширене за рахунок виділення в подальшому інших особливостей, потрібно створити в роботі.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАВДАНЬ ДОСЛІДЖЕННЯ

1.1 Наявне програмне забезпечення пошуку і рейтингування екскурсоводів

ПЗ, яке використовується для підтримки різних процесів у туризмі, є часто дуже популярним. Це стосується і ПЗ оренди житла, і ПЗ для визначення найкращих місць, і ПЗ для пошуку квитків і маршрутів, окрім того існує величезна кількість вебсайтів туристичних компаній. Але ще однією стороною є підтримка туристичних процесів екскурсоводами. Люди, які цікавляться своїм містом, які мають певні знання проводять екскурсії без участі туристичних компаній, до того ж останнім часом популярними є екскурсії від місцевих екскурсоводів, тобто окремих зацікавлених осіб, ентузіастів, а розвиватися це стало з появою вебдодатків, одним з яких є TourHQ [2] (рис. 1.1).

The screenshot displays the TourHQ profile of a guide named Eszter. The header features navigation links: DESTINATIONS, HOW IT WORKS, and ONLINE EXPERIENCES, along with a 'Plan My Holiday' button and a search icon. The profile is organized into several sections:

- Header:** Includes a globe icon, navigation links, and a 'Plan My Holiday' button.
- My Tour Calendar:** A calendar for June 2023 showing dates from Sunday to Saturday.
- Profile Information:**
 - TourHQ ID:** CZ92859
 - Experience:** Since 2013
 - Gender:** Female
 - Private Tour Guide In:** Cesky Krumlov
 - Other Guiding Areas:** Ceske Budejovice, Třeboň, Trebitsch, Kutná Hora, Tabor
 - Languages:** English, French, Hungarian
- Professional License/Certificate:** A section showing a professional license for Eszter, with a 'View All Photos' button.
- About me:** A bio stating: 'I am a hungarian graphic designer learned english as a child in California, lived in Paris for 8 years, studied on the Sorbonne art history, and moved to Cesky Krumlov in 2010. I am a real Central European with family origins from all over the territory of the formal Austro-Hungarian Empire. First as a visitor, then a local, I would love to show you the highlights and the hidden treasures of this city which changed my life.'
- Transportation:** A section stating: 'For private day trips or scenic transfers rented car/minibus (max 8 ppl) is available'.
- Guide Rating:** A section showing a rating of 2 reviews, with a bar chart indicating the distribution of ratings (Excellent, Very Good, Average, Poor, Terrible).
- Recent Reviews:** A section showing a review from Kevin: 'WE - would highly recommend Eszter, she is such a fount of knowledge, experience and good cheer. My wife and I spent the ...'.

Рисунок 1.1 – Вебдодаток TourHQ

TourHQ дозволяє працювати з ним і екскурсоводам, і мандрівникам. Екскурсоводи можуть визначати власні пропозиції, зазначаючи те, які екскурсії, в яких саме містах вони готові проводити. Тому основою доступу є як раз перелік країн, звідки можна отримати перелік міст, а за містом обрати потрібного екскурсовода. Пошук виконується саме за містами. За кожним екскурсоводом надається його номер, сертифікат, мови, якими він володіє і проводить екскурсії, місто проведення приватних екскурсій, стаття, фотографія, опис, статті екскурсовода. За кожним екскурсоводом відображаються відгуки про нього, що і є рейтингуванням у програмі. Надається можливість оформити запит на екскурсію, який передбачає ініціативу мандрівника. Тобто він визначає, що йому потрібно, а екскурсовод відповідає. Також є можливість оформити запит за конкретною екскурсією, яку проводить екскурсовод. Загалом це все саме персональні екскурсії.

Наступним вебдодатком є ToursByLocals [3] (рис. 1.2).

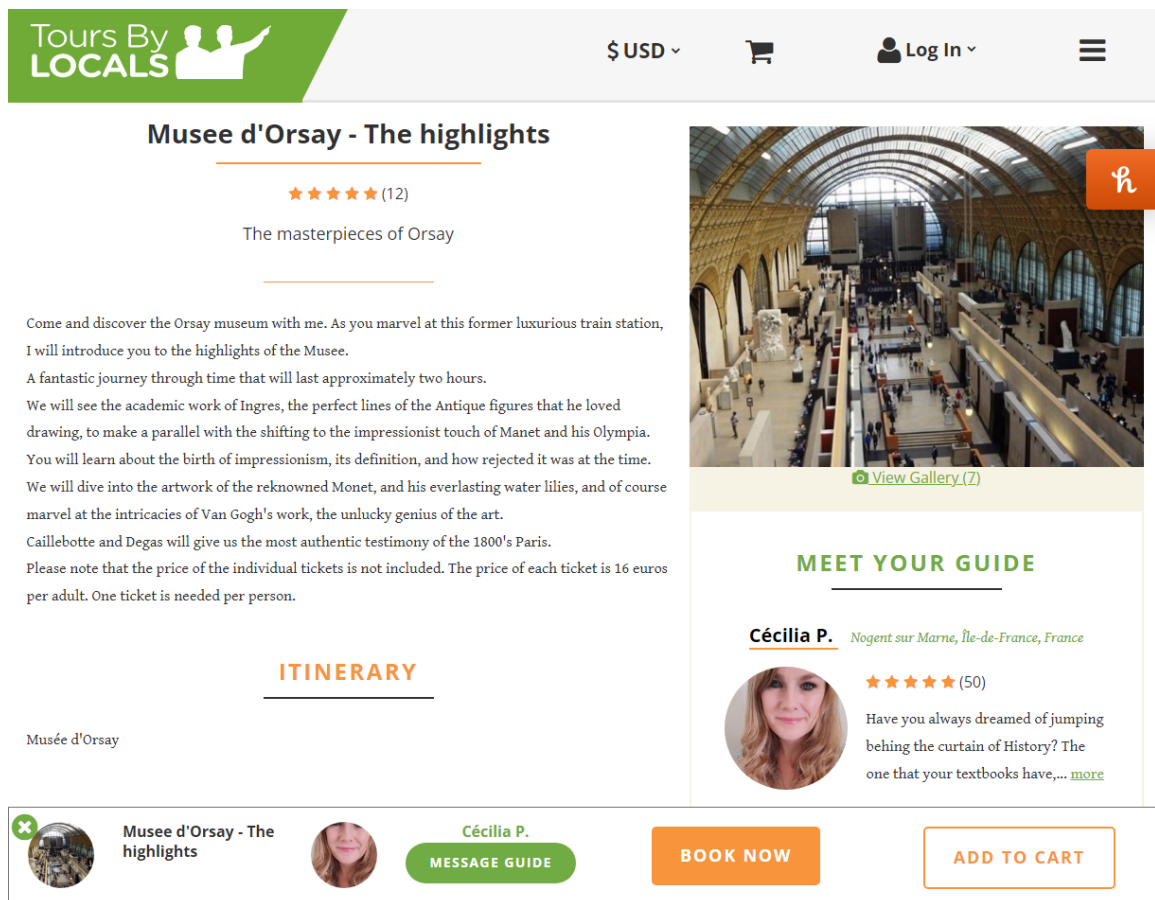


Рисунок 1.2 – Вебдодаток ToursByLocals

ToursByLocals визначає концепцію з'єднання мандрівників з екскурсоводами, які працюють поза межами туристичних агентств. Надається можливість шукати екскурсії, визначаючи місто, країну або назву екскурсії. Вже далі окремо можна конкретизувати дати проведення екскурсій, щоб визначити конкретні пропозиції. Кожну екскурсію можна забронювати на потрібні дати. Рейтингування відбувається за конкретними екскурсіями. З цього складається оцінка екскурсії, а з оцінок екскурсій складається оцінка екскурсовода. Шукати можна не тільки екскурсію, але і екскурсовода, визначивши попередньо місто. Сам мандрівник може переглядати дані власного облікового запису, змінювати їх, історію бронювань та створені ним рейтинги.

Private Tour Guide [4] (рис. 1.3) також надає можливості і для мандрівників, і для екскурсоводів. Пошук відбувається саме екскурсоводів, визначаючи мову проведення екскурсії та місто проведення.



Рисунок 1.3 – Вебдодаток Private Tour Guide

Кожен екскурсовод виставляє свою фотографію, задає інформацію про себе, визначає мови, якими володіє, валюту платежів, самі екскурсії. За екскурсією визначається кількість місць, вартість з одного учасника, фотографії. Екскурсоводу можна відправити повідомлення. Можна зробити масове відправлення повідомлень різним екскурсоводам з заданого міста. Можливостей забронювати екскурсію вебсайт не має.

Showaround [5] (рис. 1.4) дозволяє виконувати пошук екскурсоводів. Пошук відбувається за містом базово, а розширений варіант включає мову, стаття екскурсовода та напрямки екскурсій, які цікавлять мандрівника. За кожним екскурсоводом визначається його ім'я, місто, фотографія, вартість години роботи. Про екскурсоводів можна залишати відгуки, формуючи так їх рейтинг, який в підсумку виражається кількістю оцінювань та середньою оцінкою.

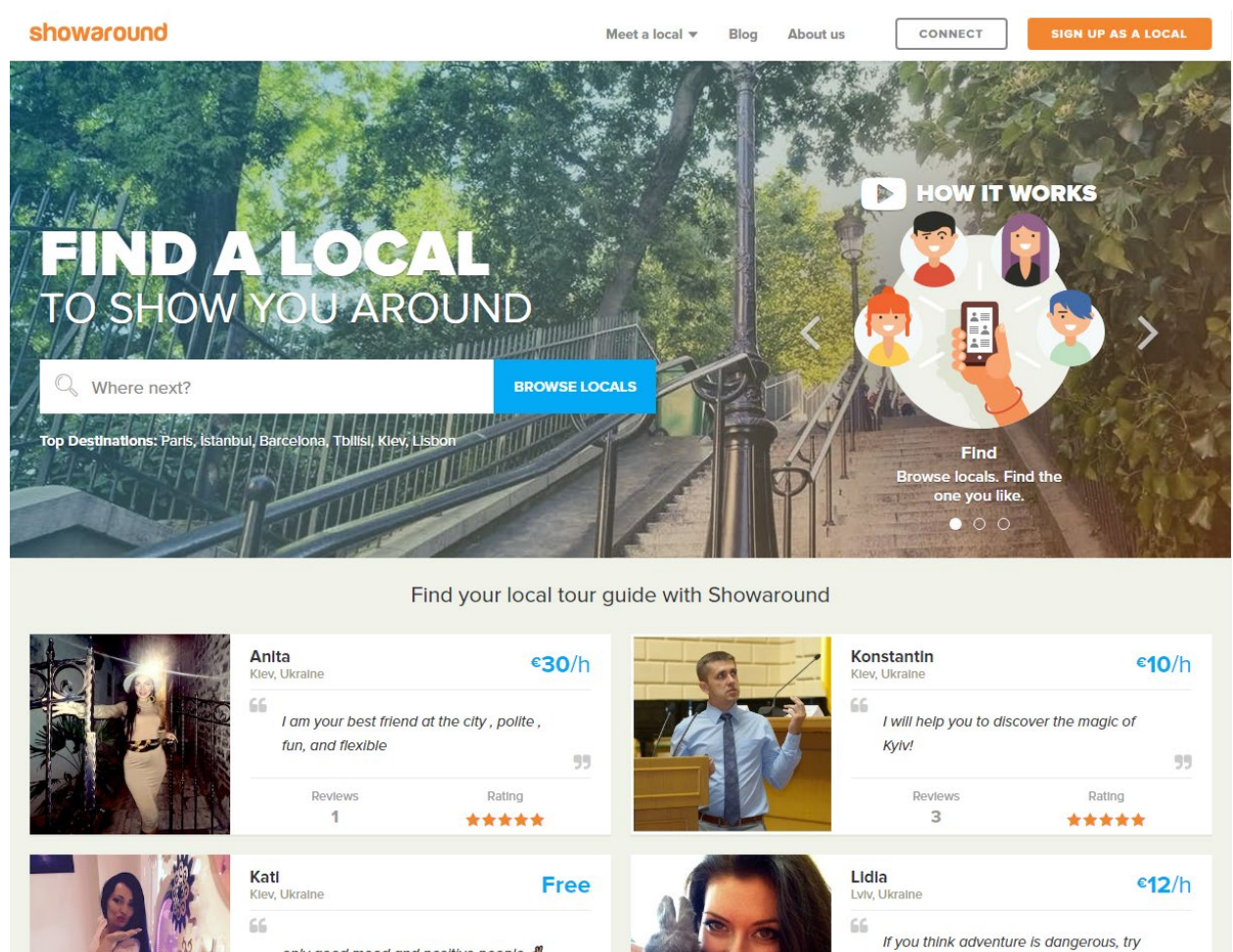


Рисунок 1.4 – Вебдодаток Showaround

GetYourGuide [6] (рис. 1.5) дозволяє бронювати місця на екскурсіях, представляючи процес взаємодії як ринок, на якому екскурсоводи створюють пропозицію, а відвідувачі екскурсій обирають з пропозицій ті, які хочуть. Рейтингування реалізується за екскурсією, але з цієї оцінки складається і оцінка екскурсовода в підсумку шляхом усереднення. Пошук екскурсій реалізується за містом з обмеженням вартості, визначенням мови проведення, встановлення проміжку тривалості, періодом проведення (вранці, вдень або ввечері). Можна визначити в розширеному варіанті конкретні місця у місті, які мандрівник бажає відвідати, власні інтереси, категорії екскурсій, рівень рейтингу. Також у пошуку можна задати конкретну дату екскурсії.

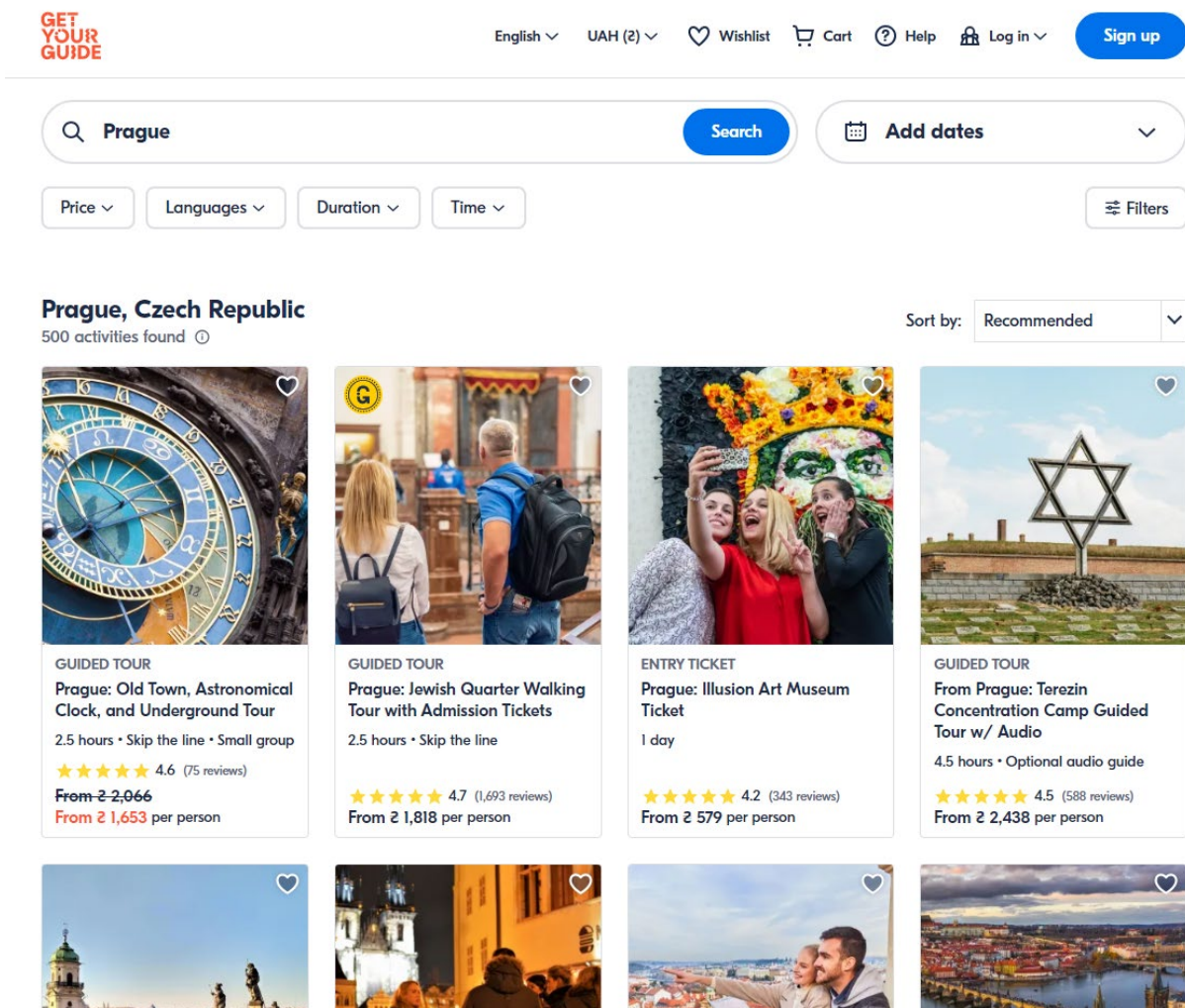


Рисунок 1.5 – Вебдодаток GetYourGuide

Окремо було виділено можливості вебдодатку Guidesfinder [7], який фактично має дошку оголошень, де мандрівники можуть визначити свої бажання щодо екскурсій, тобто напрямок взаємодії змінюється у такому випадку на протилежний.

1.2 Порівняння визначених аналогів

Порівняння виділених з усього проаналізованого набору аналогів, що включали TourHQ, GetYourGuide, Showaround, визначило, що наявні функції у програмах різняться в залежності від обраної концепції (табл. 1.1). При цьому об'єднуюча концепція відсутня. Частина програм надає можливість шукати екскурсії, але тоді не можна виконувати пошук екскурсоводів і навпаки. У частини відсутня можливість рейтингування екскурсій, але можливе рейтингування екскурсоводів і навпаки.

Таблиця 1.1 – Порівняння аналогів з розробкою роботи

Показник	TourHQ	GetYourGuide	Showaround	Розробка
Пошук екскурсій	Ні	Так	Ні	Так
Пошук екскурсоводів	Так	Ні	Так	Так
Бронювання екскурсій	Так	Так	Так	Так
Рейтингування екскурсій	Ні	Так	Ні	Так
Рейтингування екскурсоводів	Так	Пряме відсутнє	Так	Так
Створення оголошень про пошук екскурсовода	Ні	Ні	Ні	Так

У виконуваний роботі вирішено було об'єднати ці ідеї разом, щоб в підсумку можна було виконувати широкий пошук, щоб до того ж можна було створювати оголошення про пошуку екскурсоводів і виконувати екскурсоводам пошук за такими оголошеннями. Це має дозволити обом сторонам активно долучатися до створення пропозицій і забезпечувати результативний пошук.

1.3 Висновки за розділом 1

Завдання для виконання в роботі:

- проаналізувати ПЗ, що може застосовуватися для пошуку і рейтингування екскурсоводів;
- визначити мову і засоби розробки;
- визначити вимоги до програми;
- визначити структуру роботи з даними;
- виконати реалізацію ПЗ.

У першому розділі проаналізовано ПЗ що може застосовуватися для пошуку і рейтингування екскурсоводів, включаючи зокрема TourHQ, GetYourGuide, Showaround. Усі вони підтримують можливості бронювання, хоча не всі подібні програмні засоби мають таку можливість. Однак вони відрізняються в направленості пошуку, в підходах до рейтингування. Ці підходи вирішено об'єднати, розширити і розвинути у програмі, яка створюється.

2 ПРОЄКТУВАННЯ ПРОГРАМИ

2.1 Вибір мови веброзробки

На даний момент java є однією з найбільш широко використовуваних мов програмування, яка використовується і для веброзробки. Тому вибір мови веброзробки виконувався між більш класичним рішенням, яким є Java, і більш гнучким варіантом – мовою Python [8]-[9] (табл. 2.1).

Таблиця 2.1 – Порівняння мов веброзробки

Показник	Python	Java
Підтримувані парадигми програмування	Об'єктно-орієнтована, процедурна, функціональна	Об'єктно-орієнтована
Створення читабельного коду	Закладено в концепцію мови	Потребує додаткових зусиль, складніше
Варіант типізації	Динамічна	Сильно типізована
Продуктивність програми	Менша за java	Більша за python
Продуктивність розробки	Більша за java	Менша за python

У результаті розробка виконана на python, враховуючи, що це більш гнучка мова, адже зокрема надає доступ до більшої кількості парадигм програмування, відповідно можна вирішувати, яку в який момент краще використати, дозволяє без додаткових зусиль створювати читабельний код, до того ж створювати код швидше. Недоліком є менша продуктивність самої програми в підсумку, але вона для цього проєкту не є критичною, адже на практиці в результаті такі проєкти не є надто повільними і застосовуються багатьма великими компаніями у проєктах, які мають дуже широку аудиторію на даний момент.

2.2 Вимоги до програми

Вимоги до програми частково були визначені в технічному завданні, але після цього, коли було виконано аналізу предметної області, було повторно і додатково проаналізовано зокрема ті функції, які виконують аналоги і з визначеними особливостями сформульовано і промодельовано функціональні вимоги до програми.

Вимоги до програми в частині функцій, які мають надаватися неавторизованому користувачу:

- пошук екскурсій;
- пошук екскурсій для бронювання;
- пошук екскурсіводів;
- пошук запитів відвідувачів на персональні екскурсії;
- авторизація і реєстрація;
- перегляд рейтингу екскурсіводів загалом, за містом, за країною, за місцем;
- перегляд рейтингу екскурсій загалом, за містом, за країною, за місцем;
- перегляд профілю екскурсівода;
- перегляд профілю відвідувача;
- перегляд екскурсій, які проводить екскурсівод;
- перегляд варіантів екскурсії;
- перегляд запитів відвідувача на персональну екскурсію;
- перегляд країн;
- перегляд міст за країнами;
- перегляд місць за містом;
- перегляд параметрів виконуваної екскурсії;
- перегляд запиту відвідувача на персональну екскурсію (рис. 2.1).

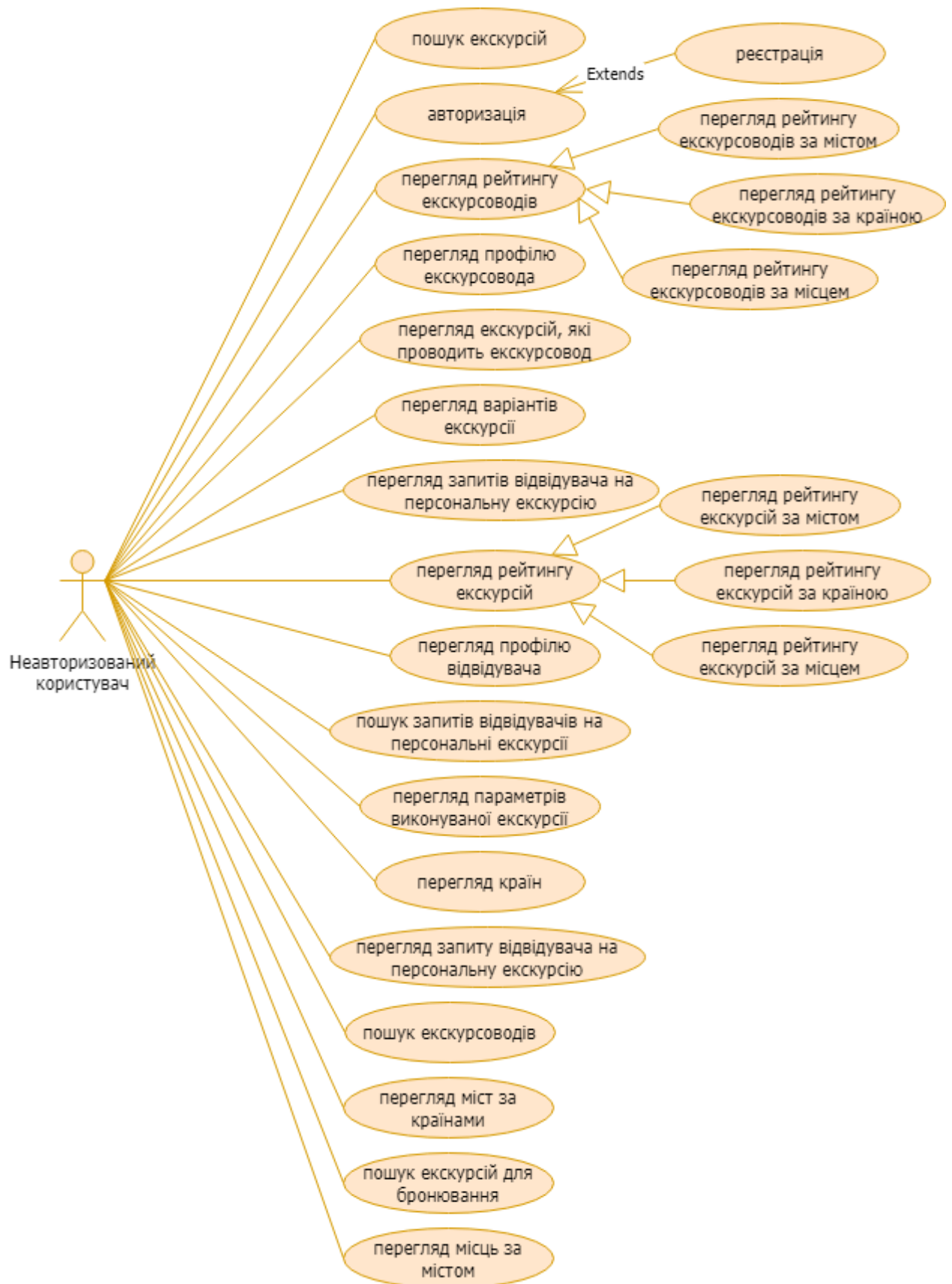


Рисунок 2.1 – Діаграма варіантів використання неавторизованим користувачем програми

Вимоги до програми в частині функцій, які мають надаватися екскурсоводу:

- створення екскурсії;
- редагування екскурсії;
- визначення дати проведення екскурсії;
- редагування дати проведення екскурсії;
- вилучення проведення екскурсії;
- перегляд всіх учасників екскурсії;
- коментування екскурсії;
- створення варіанта екскурсії;
- редагування варіанта екскурсії;
- подання відповіді на запит відвідувача на персональну екскурсію;
- перегляд власних відповідей на запити відвідувачів на персональні екскурсії;
- перегляд плану проведення екскурсії;
- перегляд плану проведення всіх екскурсій екскурсоводом;
- визначення кількості місць на екскурсії (рис. 2.2).

Вимоги до програми в частині функцій, які мають надаватися відвідувачу екскурсії:

- оцінювання екскурсовода;
- оцінювання екскурсії;
- редагування персональних даних користувача;
- бронювання місця на екскурсії;
- коментування екскурсії;
- голосування за варіант екскурсії;
- створення запиту на персональну екскурсію;
- перегляд заброньованих екскурсій;
- затвердження або скасування відповіді на запит відвідувача на персональну екскурсію;
- перегляд власних оцінювань (рис. 2.3).

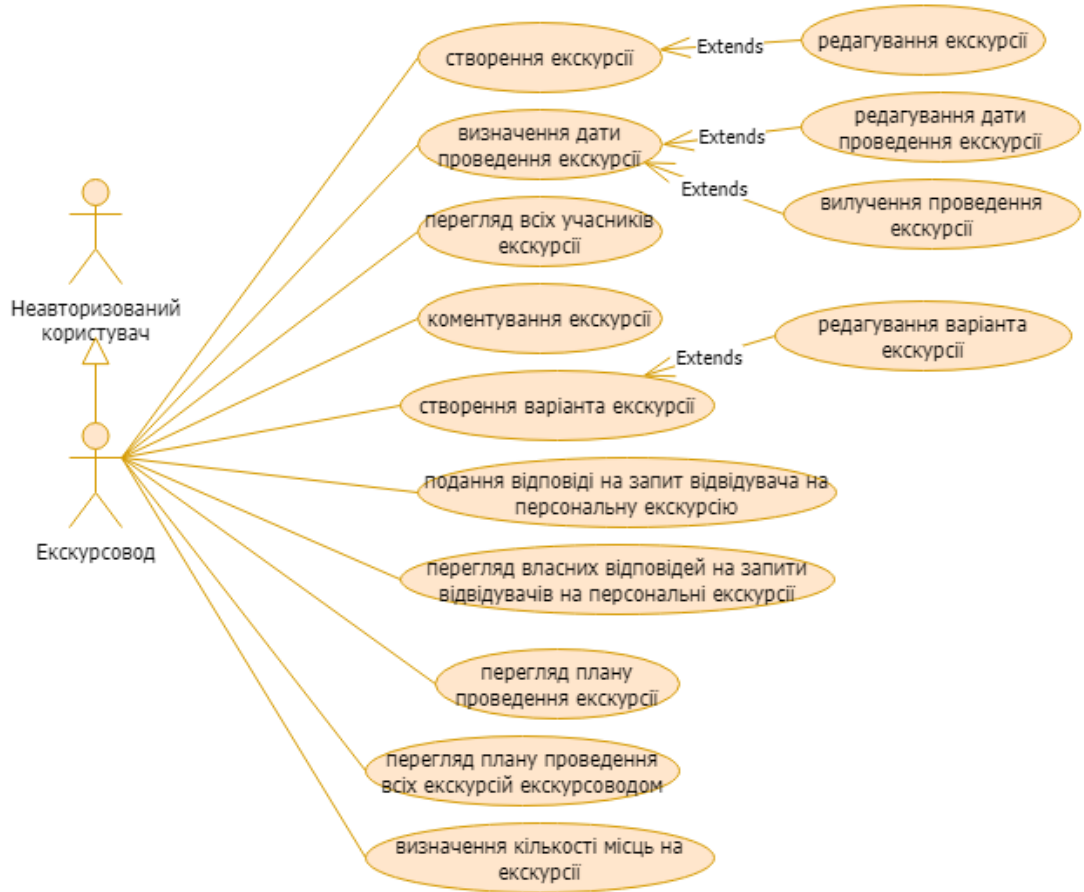


Рисунок 2.2 – Діаграма варіантів використання екскурсоводом програми



Рисунок 2.3 – Діаграма варіантів використання відвідувачем екскурсії програми

2.3 Проектування структури бази даних

База даних (БД) є важливою для створення даної програмної системи, адже всі функції, які нею виконуються залежать від даних у БД. Відповідно структура даних визначається зокрема логікою роботи програми на основі тих функцій, які були розглянуті. Забезпечується робота БД через MySQL.

З самого формулювання теми дипломної кваліфікаційної роботи зрозуміло, що центральною сутністю для роботи ПЗ є екскурсовод. Тому було виділено цю сутність і визначено при побудові реляційної БД таблицю Tourguide, яка має поля:

- id – номер екскурсовода, визначає первинний ключ таблиці;
- user – запис користувача, за яким екскурсовод працює в програмі, визначається зовнішнім ключем з таблиці User;
- tourid – ідентифікатор екскурсовода за реєстрацією у місті або країні, тобто фактично номер, визначений у ліцензії на діяльність, якщо в країні передбачено отримання такого сертифікату;
- about – інформація, надана екскурсоводом про себе;
- photo – фотографія екскурсовода;
- city – місто, в якому проживає екскурсовод або фактично основне його місто (поняття основного або неосновного може бути важливим, наприклад, коли міста знаходяться поруч в певній конгломерації, а екскурсовод дає екскурсії в різних містах з конгломерації), визначається зовнішнім ключем з таблиці City;
- experience – досвід проведення екскурсій: вказується рік початку діяльності;
- rating – середній бал, виставлений екскурсоводу всіма бажаючими;
- verifiedrating – середній бал, виставлений екскурсоводу всіма відвідувачами його екскурсій;
- phone – номер телефону екскурсоводу.

Основним результатом роботи екскурсовода є екскурсія, тому це наступна важлива сутність, яку було представлено у вигляді таблиці Excursion, яка має поля:

- id – номер екскурсії, визначає первинний ключ таблиці;
- name – назва екскурсії;
- about – детальний опис екскурсії;
- tourguide – екскурсовод, який проводить цю екскурсію, визначається зовнішнім ключем з таблиці Tourguide;
- rating – середній бал, виставлений окремо цій екскурсії всіма бажаючими;
- verifiedrating – середній бал, виставлений цій екскурсії всіма її відвідувачами;
- plan – план екскурсії у формі опису;
- duration – орієнтовна тривалість екскурсії, визначена в хвилинах;
- highlights – важливі особливості екскурсії;
- requirements – вимоги для участі в екскурсії: якщо екскурсія вимагає наявності якогось спеціального взуття, якоїсь техніки або має ще якісь вимоги, то це має бути зазначено;
- meetingpoint – місце зустрічі для початку екскурсії, визначається повною адресою цієї локації;
- additional – додаткові витрати під час екскурсії: платежі для входу в окремі місця, сплата проїзду мають бути тут зазначені;
- cashneeded – необхідність мати готівку з собою: при відвідуванні певних закладів окремо може сплачуватися вхідний квиток і вимогою є наявність саме готівки;
- restrictions – обмеження щодо екскурсії: наприклад, якщо екскурсовод не має ліцензії для проведення екскурсії в певних місцях, то такі обмеження можуть зазначатися;
- monotour – екскурсія в межах тільки одного міста або місця чи екскурсія, яка охоплює декілька місць або міст;

- transportation – наявність переїздів під час проведення екскурсій;
- activitylevel – рівень фізичної активності під час даної екскурсії;
- participants – стандартна максимальна кількість відвідувачів цієї екскурсії;

- groups – можливість участі в екскурсії організованих груп: екскурсовод може визначати, що в екскурсії можуть брати індивідуальні учасники, друзі, групи, але не організовані групи, тому що наприклад, для них екскурсії організовуються окремо;

- disabled – можливість участі людей з інвалідністю в екскурсії: таким чином визначається, чи наявний безбар'єрний доступ до всіх місць екскурсії;

- kids – можливість участі дітей (людей віком до 18) у екскурсії;

- pay – необхідність попередньої оплати за участь у екскурсії, тобто виконання сплати під час бронювання, якщо екскурсія не є безкоштовною.

Тип екскурсії може включати фактично певну тематику екскурсії, наприклад, культурна, історична, розважальна тощо. Тип екскурсії представлено у вигляді таблиці Tourtype, яка має поля:

- id – номер типу екскурсії, визначає первинний ключ таблиці;

- name – назва типу екскурсії.

Кожна екскурсія може мати більше одного типу, а кожен тип відповідно має декілька екскурсій, тому зв'язування між цими таблицями виконується зв'язуючою таблицею ExcursionType, яка має поля:

- excursion – екскурсія, яка характеризується цим типом, визначається зовнішнім ключем з таблиці Excursion;

- tourtype – тип, до якого належить ця екскурсія, визначається зовнішнім ключем з таблиці Tourtype.

Екскурсія в різні дати може проводитися різними мовами, якщо звичайно екскурсовод знає їх. Окрім того під час проведення екскурсії екскурсовод може надавати аудіогід або певні матеріали іншою мовою. Такі варіанти представлені у вигляді таблиці ExcursionLanguage, яка має поля:

- id – номер варіанта проведення екскурсії деякою мовою, визначає первинний ключ таблиці;

- excursion – екскурсія, яка проводиться цією мовою, визначається зовнішнім ключем з таблиці Excursion;

- live – проведення екскурсії цією мовою самим екскурсоводом чи за допомогою аудіогіда;

- language – мова, якою проводиться ця екскурсія, визначається зовнішнім ключем з таблиці Language.

Цінова група квитків на екскурсію представлена у вигляді таблиці ExcursionPricegroup, яка має поля:

- id – номер цінової групи квитків на екскурсію, визначає первинний ключ таблиці;

- name – назва цінової групи;

- included – що включено в цю цінову групу;

- excursion – екскурсія, для якої визначена ця цінова група, визначається зовнішнім ключем з таблиці Excursion;

- participants – кількість квитків (місць) на цю екскурсію за саме цією ціновою пропозицією;

- group – розмір групи, для якої застосовується ця вартість;

- price – вартість квитків у цій ціновій категорії.

Для екскурсій користувачі можуть створювати коментарі, які представлені у вигляді таблиці Comment, яка має поля:

- id – номер, визначає первинний ключ таблиці;

- excursion – екскурсія, для якої створено цей коментар, визначається зовнішнім ключем з таблиці Excursion;

- user – користувач, який створив цей коментар (може бути і відвідувач, і інший екскурсовод), визначається зовнішнім ключем з таблиці User;

- comment_text – текст коментаря;

- publication – час публікації коментаря.

Екскурсія може охоплювати декілька міст, хоча може включати і лише одне місто. Інформація про місто, в якому відбувається екскурсія (певний маршрут), представлена у вигляді таблиці `ExcursionCity`, яка має поля:

- `id` – номер міста в екскурсії, визначає первинний ключ таблиці;
- `excursion` – екскурсія, до якої відноситься це місто, визначається зовнішнім ключем з таблиці `Excursion`;
- `city` – місто, яке буде відвідано в цій екскурсії, визначається зовнішнім ключем з таблиці `City`;
- `position` – порядковий номер міста саме в цій екскурсії серед всіх міст і місць;
- `details` – деталі щодо відвідування цього міста в екскурсії;
- `duration` – тривалість відвідування цього міста під час екскурсії.

Екскурсія може охоплювати декілька місць (музеїв, наприклад), хоча може включати і лише одне місце. Інформація про місце, в якому відбувається екскурсія, представлена у вигляді таблиці `ExcursionPlace`, яка має поля:

- `id` – номер місця в екскурсії, визначає первинний ключ таблиці;
- `excursion` – екскурсія, до якої відноситься це місце, визначається зовнішнім ключем з таблиці `Excursion`;
- `position` – порядковий номер місця саме в цій екскурсії серед всіх міст і місць;
- `place` – місце, яке буде відвідано в цій екскурсії, визначається зовнішнім ключем з таблиці `Place`;
- `details` – деталі щодо відвідування цього місця в екскурсії;
- `duration` – тривалість відвідування цього місця під час екскурсії.

Екскурсії можуть мати деякі особливі варіанти. Вони можуть створюватися для того, щоб визначити який варіант більше є перспективним для відвідувачів. Це визначення відбувається шляхом голосування. Варіант екскурсії представлений у вигляді таблиці `Excursionvariant`, яка має поля:

- `id` – номер варіанта екскурсії, визначає первинний ключ таблиці;

- excursion – екскурсія, до якої відноситься цей варіант, визначається зовнішнім ключем з таблиці Excursion;

- name – назва варіанта екскурсії;

- feature – особливості цього варіанта екскурсії.

Голоси, які були віддані за варіанти екскурсії, представлені у вигляді таблиці ExcursionvariantVote, яка має поля:

- id – номер голосу, відданого за варіант екскурсії, визначає первинний ключ таблиці;

- excursionvariant – варіант екскурсії, до якої відноситься цей варіант, визначається зовнішнім ключем з таблиці Excursionvariant;

- visitor – відвідувач, який проголосував за цей варіант екскурсії, визначається зовнішнім ключем з таблиці Visitor;

- creation – час, коли було віддано цей голос за цей варіант екскурсії.

Варіанти екскурсій можуть відрізнятися тим, що до їх складу входить якесь особливе місто, яке не входить до плану стандартної екскурсії. Всі такі міста представлені у вигляді таблиці ExcursionvariantCity, яка має поля:

- excursionvariant – варіант екскурсії, до якого входить це місто, визначається зовнішнім ключем з таблиці Excursionvariant;

- city – місто, яке входить тільки до цього варіанта цієї екскурсії, визначається зовнішнім ключем з таблиці City.

З іншого боку частина міст може повторювати стандартний варіант базової екскурсії, тобто ті міста, які асоціюються з таблицею Excursion. Інформація про такі міста представлені у вигляді таблиці ExcursionvariantBasecity, яка має поля:

- excursionvariant – варіант екскурсії, до якого входить це місто, визначається зовнішнім ключем з таблиці Excursionvariant;

- excursioncity – місто основної екскурсії, яке входить до цього варіанта екскурсії, визначається зовнішнім ключем з таблиці ExcursionCity.

Варіанти екскурсій можуть відрізнятися тим, що до їх складу входить не тільки якесь особливе місто, але і місце, яке не входить до плану

стандартної екскурсії. Всі такі місця представлені у вигляді таблиці `ExcursionvariantPlace`, яка має поля:

- `excursionvariant` – варіант екскурсії, до якого входить це місце, визначається зовнішнім ключем з таблиці `Excursionvariant`;
- `place` – місце, яке входить тільки до цього варіанта цієї екскурсії, визначається зовнішнім ключем з таблиці `Place`.

З іншого боку частина місць може повторювати стандартний варіант базової екскурсії, тобто ті місця, які асоціюються з таблицею `Excursion`. Інформація про такі місця представлені у вигляді таблиці `ExcursionvariantBaseplace`, яка має поля:

- `excursionvariant` – варіант екскурсії, до якого входить це місце, визначається зовнішнім ключем з таблиці `Excursionvariant`;
- `excursionplace` – місце основної екскурсії, яке входить до цього варіанта екскурсії, визначається зовнішнім ключем з таблиці `ExcursionPlace`.

Країни в БД представлені у вигляді окремої таблиці `Country`, яка має поля:

- `id` – номер країни, визначає первинний ключ таблиці;
- `name` – назва країни;
- `photo` – фотографія країни.

Міста в БД представлені у вигляді таблиці `City`, яка має поля:

- `id` – номер міста, визначає первинний ключ таблиці;
- `country` – країна, в якій знаходиться це місто, визначається зовнішнім ключем з таблиці `Country`;
- `name` – назва міста;
- `photo` – фотографія міста.

Місця, тобто музеї, особливі визначні місця представлені у вигляді таблиці `Place`, яка має поля:

- `id` – номер місця, визначає первинний ключ таблиці;
- `placetype` – тип місця, визначається зовнішнім ключем з таблиці `Placetype`;

- name – назва місця;
- photo – фотографія місця;
- city – місто, в якому знаходиться це місце, визначається зовнішнім ключем з таблиці City;

- address – адреса місця;
- website – вебсайт місця.

Тип місця, який може використовувється в БД, представлений у вигляді таблиці Placetype, яка має поля:

- id – номер типу місця, визначає первинний ключ таблиці;
- name – назва типу місця.

Рейтингування екскурсії представлено у вигляді таблиці ExcursionRate, яка має поля:

- id – номер рейтингування екскурсії, визначає первинний ключ таблиці;

- excursion – екскурсія, якої стосується це рейтингування, визначається зовнішнім ключем з таблиці Excursion;

- visitor – відвідувач, який виконав це рейтингування, визначається зовнішнім ключем з таблиці Visitor;

- verified визначає, чи це перевірене рейтингування, чи ні: якщо воно перевірене, це означає, що цей відвідувач бронював цю екскурсію;

- rate – рейтинг, виставлений відвідувачем екскурсії за шкалою від 1 до 10;

- rate_comment – текст відгуку відвідувача, який асоціюється з цим рейтингом;

- creation – час створення рейтингування.

До рейтингування екскурсії відвідувач може визначити фотографію. Ці фотографії представлені у вигляді таблиці ExcursionRatePhoto, яка має поля:

- id – номер фотографії, яка віднесена до рейтингування, визначає первинний ключ таблиці;

- excursionrate – рейтингування екскурсії, до якого віднесена ця фотографія, визначається зовнішнім ключем з таблиці ExcursionRate;
- photo – фотографія.

Рейтингування екскурсовода представлено у вигляді таблиці TourguideRate, яка має поля:

- id – номер рейтингування екскурсовода, визначає первинний ключ таблиці;
- tourguide – екскурсовод, рейтинг якому було виставлено, визначається зовнішнім ключем з таблиці Tourguide;
- visitor – відвідувач, який виконав це рейтингування, визначається зовнішнім ключем з таблиці Visitor;
- verified визначає, чи це перевірене рейтингування, чи ні: якщо воно перевірене, це означає, що цей відвідувач бронював екскурсію у цього екскурсовода;
- rate – рейтинг, виставлений відвідувачем екскурсії екскурсоводу за шкалою від 1 до 10;
- rate_comment – текст відгуку відвідувача, який асоціюється з цим рейтингом;
- creation – час створення рейтингування.

Час проведення конкретних екскурсій представлений у вигляді таблиці ExcursionDate, яка має поля:

- id – номер проведення екскурсії, визначає первинний ключ таблиці;
- excursionlanguage – мова з усіх варіантів мов для проведення екскурсії, якою проводиться екскурсія в цю дату, визначається зовнішнім ключем з таблиці ExcursionLanguage;
- feature – особливість екскурсії цього дня, якщо вона є;
- date – дата, коли буде проведено або було проведено екскурсію;
- participants – кількість місць в екскурсії цього разу: визначається тоді, коли кількість відвідувачів обмежена, тоді може бути застосована можливість придбання квитків різних цінових категорій, але тільки до того моменту, поки

сумарна кількість всіх придбаних квитків не стану перевищувати це сумарне значення;

- timestart – час початку екскурсії;
- state – чи продовжується бронювання на ці дати ще.

Бронювання екскурсій відвідувачами представлено у вигляді таблиці Booking, яка має поля:

- id – номер бронювання екскурсії відвідувачем, визначає первинний ключ таблиці;
- visitor – відвідувач, який забронював місце на цій екскурсії або місця, визначається зовнішнім ключем з таблиці Visitor;
- excursiondate – дата екскурсії, за якою це бронювання виконано, визначається зовнішнім ключем з таблиці ExcursionDate;
- creation – час створення бронювання;
- price – вартість бронювання;
- state – стан бронювання: сплачене або ні;
- participants – кількість місць, яку заброньовано в бронюванні.

Якщо в бронюванні заброньовано декілька місць, тобто це місця не тільки для себе, то можуть бути визначені прізвища осіб, які відвідають екскурсію. Ця інформація представлена у вигляді таблиці BookingGroup, яка має поля:

- id – номер особи з групи з бронювання, визначає первинний ключ таблиці;
- booking – бронювання, до якого відноситься ця особа з групи, визначається зовнішнім ключем з таблиці Booking;
- person – прізвище, ім'я особи, яка входить до групи в бронюванні.

Фотографії екскурсій для їх візуального представлення представлені у вигляді таблиці ExcursionPhoto, яка має поля:

- id – номер фотографії екскурсії, визначає первинний ключ таблиці;
- excursion – екскурсія, якої стосується ця фотографія, визначається зовнішнім ключем з таблиці Excursion;

– photo – фотографія.

Відео екскурсій для їх візуального представлення представлені у вигляді таблиці ExcursionVideo, яка має поля:

- id – номер відео екскурсії, визначає первинний ключ таблиці;
- excursion – екскурсія, якої стосується це відео, визначається зовнішнім ключем з таблиці Excursion;
- video – відео (посилання на відео).

Відвідувач екскурсій, тобто фактично користувач, який працює з цими функціями в програмі, представлений у вигляді таблиці Visitor, яка має поля:

- id – номер відвідувача, визначає первинний ключ таблиці;
- user – запис користувача, за яким відвідувач працює в програмі, визначається зовнішнім ключем з таблиці User;
- about – інформація, надана відвідувачем про себе;
- photo – фотографія відвідувача;
- city – місто, в якому проживає відвідувач, визначається зовнішнім ключем з таблиці City;
- phone – номер телефону відвідувача.

Користувач за своїм обліковим записом представлений у вигляді таблиці User, яка має поля:

- id – номер користувача, визначає первинний ключ таблиці;
- username – логін користувача;
- password – пароль;
- email – електронна пошта;
- is_active – дозвіл чи заборона працювати;
- is_staff – менеджер або звичайний користувач.

Запити відвідувачів на персональні екскурсії створюються як певні оголошення, коли відвідувач просить провести для нього певну екскурсію, а вже екскурсоводи можуть звертатися з пропозиціями. Ці запити представлені у вигляді таблиці Personalrequest, яка має поля:

- id – номер запиту відвідувача на персональну екскурсію, визначає первинний ключ таблиці;
- visitor – відвідувач, який створив цей запит, визначається зовнішнім ключем з таблиці Visitor;
- about – опис вимог;
- language – мова, якою потрібно провести екскурсію, визначається зовнішнім ключем з таблиці Language;
- city – місто, в якому відвідувачу потрібно провести екскурсію, визначається зовнішнім ключем з таблиці City;
- creation – час створення запиту;
- date – дата проведення екскурсії;
- duration – потрібна тривалість екскурсії;
- state – стан запиту, тобто чи ще він активний.

Мови, якими можуть проводитися екскурсіями, представлені у вигляді таблиці Language, яка має поля:

- id – номер мови, визначає первинний ключ таблиці;
- name – назва мови.

Відповіді на запити відвідувачів на персональні екскурсії, представлений у вигляді таблиці PersonalrequestAnswer, яка має поля:

- id – номер відповіді на запит відвідувача на персональну екскурсію, визначає первинний ключ таблиці;
- personalrequest – запит відвідувача на персональну екскурсію, до якого належить ця відповідь, визначається зовнішнім ключем з таблиці Personalrequest;
- tourguide – екскурсовод, який створив цю відповідь, визначається зовнішнім ключем з таблиці Tourguide;
- details – деталі відповіді;
- price – пропозиція з вартістю екскурсії;
- creation – час створення відповіді;
- state – стан відповіді (рис. 2.4).

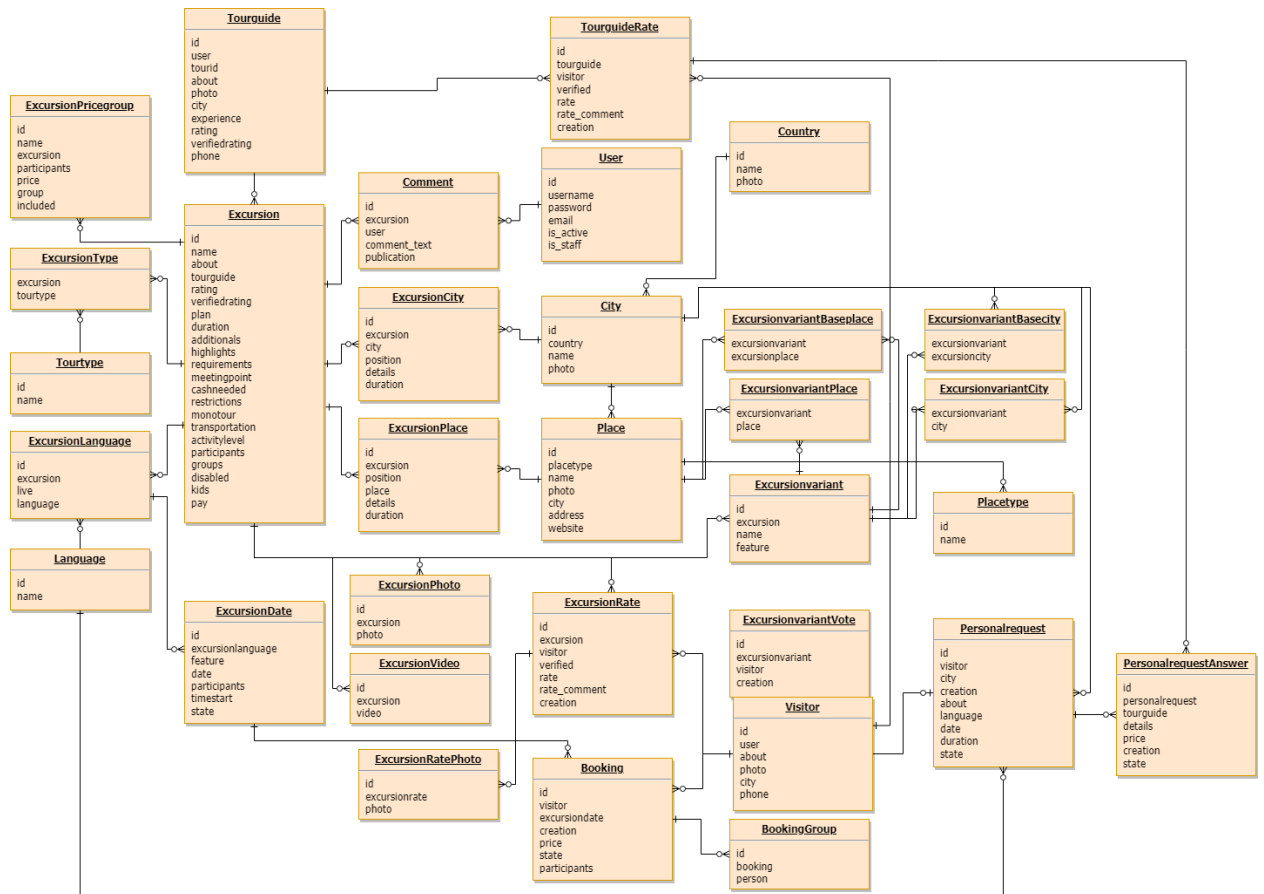


Рисунок 2.4 – Схема БД

2.4 Висновки за розділом 2

У другому розділі забезпечено виконання завдань:

- визначено мову і засоби розробки, що охоплюють мову python, систему для роботи з БД MySQL, базові мови веброзробки, включаючи HTML, CSS, javascript;
- визначено вимоги до ПЗ і представлено діаграми з визначенням сценаріїв роботи користувачів;
- визначено структуру роботи з даними за допомогою фактично логічної схеми БД та її опису.

3 РОЗРОБКА ПРОГРАМИ

3.1 Порядок функціонування програми

Пошук є одним з визначних процесів у роботі програми. Він представлений чотирма варіантами, але виходячи з теми, базовим є пошук екскурсоводів (рис. 3.1).

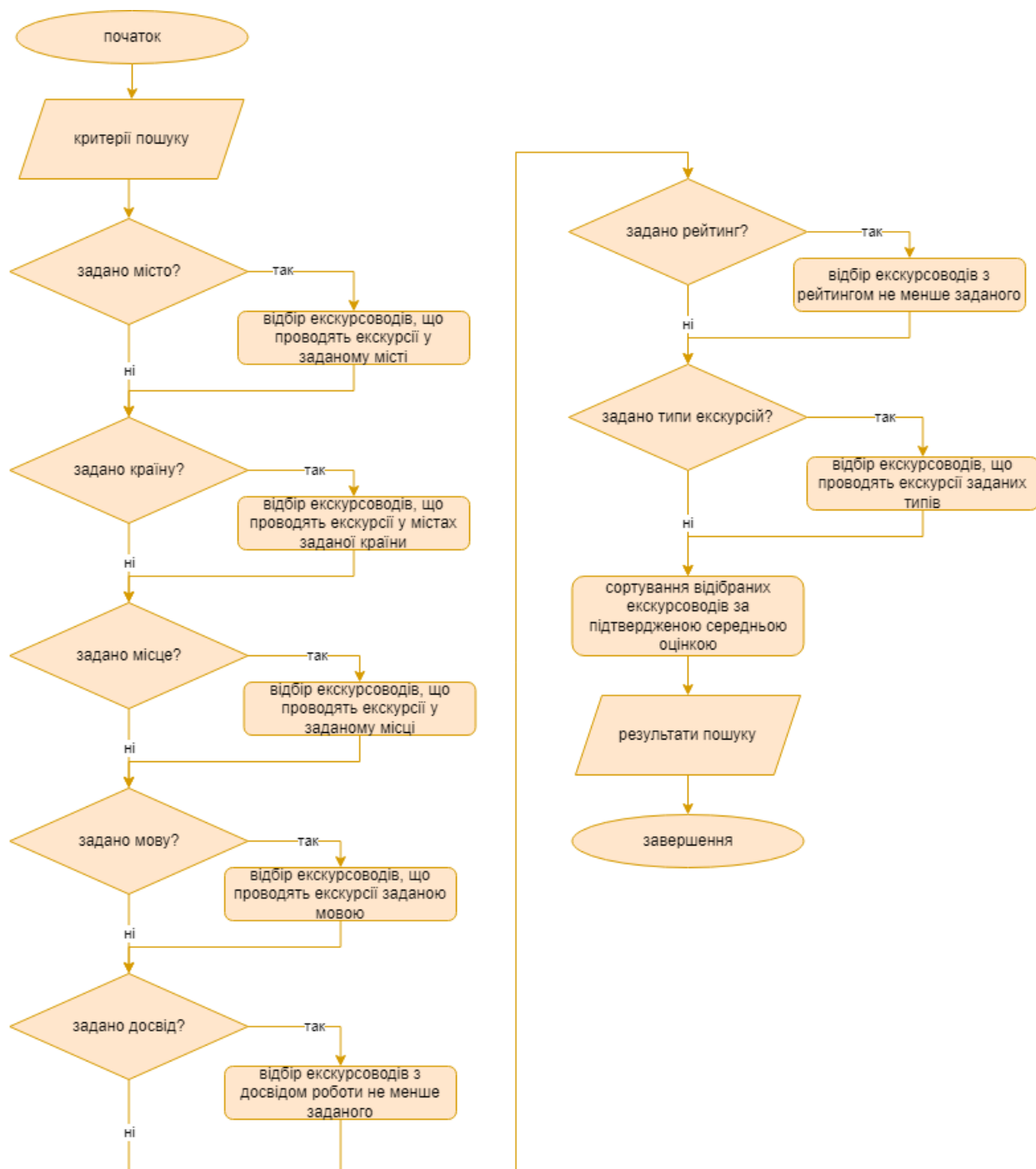


Рисунок 3.1 – Порядок виконання пошуку екскурсоводів

Пошук екскурсоводів складається з визначення відповідності екскурсовода тому набору значень параметрів пошуку, які були встановлені користувачем. Записи відібраних таким чином екскурсоводів далі відсортовуються та повертаються в результаті роботи.

3.2 Реалізація функцій програмної системи

При реалізації вебдодатку мову python було застосовано з використанням фреймворку Django, що сприяло більш швидкій реалізації. Кожна функція, запроєктована і змодельована у розділі 2, потребувала створення певного шаблону сторінки. Для реалізації використовувався безкоштовний шаблон [10]. Кожна сторінка потребувала наповнення необхідними даними. Для цього було створено функції вебдодатку, які таке наповнення виконували. Вони нижче описані. Усі вони були поділені на 4 окремі моделі, перший з яких охоплював усі функції, пов'язані з переглядом даних, другий – функції для екскурсовода, третій – відвідувача екскурсій, четвертий – засоби авторизації та реєстрації.

Пошук екскурсій було програмно реалізовано через функцію `find_excursions_view`. Вона має параметри, які визначають параметри пошуку, встановлюючи відповідні умови, які перевірялись для кожної екскурсії, залишаючи в результаті тільки ті, які відповідали цим значенням: `city` – місто, екскурсія в якому має відбуватися, `country` – країна, екскурсія в якій має відбуватися, `place` – місце, екскурсія в якому має відбуватися, `rating_from` – мінімальний рейтинг, якому має відповідати екскурсія, `duration_from` – мінімальна тривалість екскурсії, `duration_to` – максимальна тривалість екскурсії, `mono` – необхідність охоплення екскурсією тільки одного міста (1 – так, 2 – ні, тільки екскурсії з багатьма містами), `transportation` – наявність транспортування під час екскурсії (1 – так, 2 – ні, тільки без транспортування), `activitylevel_from` – мінімальний рівень активності під час екскурсії, `activity_to` – максимальний прийнятний рівень активності під час екскурсії,

group – доступ організованих груп (1 – потрібний, 2 – без груп обов’язково), participants – кількість відвідувачів, kids – необхідність відвідування з дітьми, pay – необхідність сплати за бронювання, disabled – можливість доступу юдей з інвалідністю, excursiontypes – перелік типів екскурсій, languages – прелік мов, якими має вестися екскурсія, live – достатність наявності перекладача або аудіогіда відповідною мовою чи ні, cities – перелік міст, які потрібно відвідати в цій екскурсії, cities_time – мінімальна тривалість відвідування кожного переліченого міста, city_and – необхідність наявності всіх місць у екскурсії чи достатність тільки одного, places – перелік місць, які потрібно відвідати в цій екскурсії, places_time – мінімальна тривалість відвідування кожного переліченого місця. За кожним містом і місцем відбувається перевірка, що воно відвідується не меншу кількість хвилин, ніж вказано в параметрах пошуку. Також за вказаним параметром перевіряється, що це виконується для всіх міст і місць або тільки для хоча би одного.

Аналогічно влаштований пошук екскурсій для бронювання на основі функції `find_booked_excursions_view`. Однак оскільки в цій функції пошук вже знаходиться на конкретні дати, то в параметрах пошуку також наявні додаткові параметри: `excursiondate` – необхідна дата проведення екскурсії, `time_begin` – найбільш ранній час початку екскурсії, `time_end` – найбільш пізній час завершення екскурсії, `price` – максимальна вартість екскурсії, `tickets` – необхідна кількість місць. За кожною екскурсією виділяються ті дати, які підпадають під вказану дату та під вказаний інтервал часу. За відібраними датами виділяється кількість заброньованих місць і обчислюється кількість вільних для кожної цінової категорії. Якщо потрібна кількість місць є в наявності і вартість не перевищує задану, то екскурсія за цією датою вноситься в результати.

Пошук екскурсоводів було програмно реалізовано через функцію `find_tourguides_view`. Вона має параметри: `city` – місто, яке повинно входити в екскурсію цього екскурсовода, `country` – країна, місто з якого має входити до екскурсії цього екскурсовода, `place` – місце, яке повинно входити в екскурсію

цього екскурсовода, `experience` – мінімальний досвід, з якого працює екскурсовод (вказується рік), `rating_from` – мінімальний рейтинг екскурсовода, `excursiontypes` – типи екскурсій, які має виконувати екскурсовод, `languages` – мови, якими екскурсовод проводить екскурсії.

Пошук запитів відвідувачів на персональні екскурсії було програмно реалізовано через функцію `find_persrequests_view`, яка має параметри `city`, що визначає місто, яке в запитів вказав відвідувач екскурсій, `country`, що визначає країну, в місті якої потрібно організувати екскурсію відвідувачу, `date_from` – дата, з якої може виконуватися екскурсія, `date_to` – дата, до якої може виконуватися екскурсія, `duration_from` – мінімальна тривалість, `duration_to` – максимальна тривалість потрібної екскурсії, `language` – потрібна мова екскурсії.

Перегляд рейтингу екскурсоводів загалом реалізовано на основі функції `tourguides_rating_view`, де відбираються 50 найкращих за підтвердженою оцінкою екскурсоводів.

Перегляд рейтингу екскурсоводів за містом реалізовано на основі функції `city_tourguides_rating_view`. Номер міста задається параметром `city`. Місто є містом проживання або основної діяльності екскурсовода, тобто містом, яке сам визначив для себе екскурсовод. Для створення рейтингу за містами, які входять в екскурсії екскурсовода, реалізовано функцію `city_excurses_tourguides_rating_view`.

Перегляд рейтингу екскурсоводів за країною реалізовано на основі функції `country_tourguides_rating_view`. Номер країни задається параметром `country`. Це країна, в місті якої проживає екскурсовод.

Перегляд рейтингу екскурсоводів за місцем реалізовано на основі функції `place_tourguides_rating_view`, параметр `place` якої визначає номер місця, яке має входити до екскурсії, щоб її було розглянуто протягом створення рейтингу.

Перегляд рейтингу екскурсій загалом реалізовано на основі функції `excursion_rating_view`, де відбираються 50 найкращих за підтвердженою оцінкою екскурсій.

Перегляд рейтингу екскурсій за містом реалізовано на основі функції `city_excursion_rating_view`. Номер міста задається параметром `city`. Місто є містом проживання або основної діяльності екскурсовода, тобто містом, яке сам визначив для себе екскурсовод.

Перегляд рейтингу екскурсій за країною реалізовано на основі функції `country_tourguides_rating_view`. Номер країни задається параметром `country`. Це країна, в місті якої проживає екскурсовод.

Перегляд рейтингу екскурсій за місцем реалізовано на основі функції `place_tourguides_rating_view`, параметр `place` якої визначає номер місця, яке має входити до екскурсії, щоб її було розглянуто протягом створення рейтингу.

Перегляд профілю екскурсовода реалізовано функцією `tourguide_view`, яка має параметр `tourguide`, що задає номер екскурсовода. Якщо значення цього параметра пропущено, то дані повертаються за даними облікового запису. Дані включають сам профіль екскурсовода, 3 найкращі за рейтингом екскурсії цього екскурсовода, оцінювання екскурсовода відвідувачами екскурсій.

Перегляд профілю відвідувача реалізовано функцією `visitor_view`, яка має параметр `visitor`, що задає номер відвідувача. Якщо значення цього параметра пропущено, то дані повертаються за даними облікового запису. Дані включають сам профіль відвідувача, оцінювання, створені цим відвідувачем, як щодо екскурсій, так і щодо екскурсоводів, перелік створених відвідувачем запитів на персональні екскурсії. 3 найкращі за рейтингом екскурсії цього екскурсовода, оцінювання екскурсовода відвідувачами екскурсій.

Перегляд екскурсій, які проводить екскурсовод, реалізовано функцією `tourguide_excursions_view`, яка має параметр `tourguide`, що визначає номер

екскурсовода. Якщо пропущено значення, то беруться дані авторизованого екскурсовода.

Перегляд параметрів виконуваної екскурсії реалізовано на основі функції `excursion_view`, яка має параметр `excursion`, що визначає номер такої екскурсії. До результатів, які передаються в файл шаблону, належать: підтримка мови без аудіогіда або з ним – `live_languages`, підтримка певних мов з аудіогідом у тому числі – `languages`, цінові пропозиції за цією екскурсією – `prices`, коментарі до цієї екскурсії – `comments`, місця, які потрібно відвідати, – `places`, можливість голосувати за варіанти екскурсії – `vote`, оцінювання цієї екскурсії відвідувачами – `rates`, дати проведення екскурсії – `dates`, фотографії з екскурсії – `photos`, – відео з екскурсії, `videos` – відео для ілюстрування деталей зустрічі.

Перегляд країн, тобто формування переліку країн, щоб далі можна було перейти до сторінки з вибраною країною реалізовано функцією `countries_view`. Подальший перегляд міст за країною реалізовано функцією `country_cities_view`, яка має параметр `country`, що як раз визначає номер країни, обраної на попередньому етапі взаємодії.

Перегляд власних оцінювань реалізовано через функцію `ratings_view`. За авторизованим відвідувачем відбираються всі оцінювання екскурсій та екскурсоводів, які він виконав.

Перегляд плану проведення екскурсії виконується на основі функції `excursion_dates_view`, параметр якої `excursion` визначає номер екскурсії, для якої повертається перелік всіх дат для проведення, які за нею були створені.

Перегляд плану проведення всіх екскурсій екскурсоводом реалізовано функцією `toureguide_dates_view`, яка за авторизованим екскурсоводом виділяє всі дати екскурсій, які проводить цей екскурсовод.

Перегляд всіх учасників екскурсії здійснюється екскурсоводом за допомогою функції `excursion_participants_view`, в якій параметром `excursiondate` вказується номер проведення конкретної екскурсії. Повертається перелік всіх, хто сплатив за екскурсію.

Перегляд заброньованих екскурсій реалізується функцією `visitor_excursions_view`, яка за авторизованим відвідувачем відбирає дані всіх бронювань, створених цим відвідувачем.

Перегляд запитів відвідувача на персональну екскурсію реалізовано функцією `visitor_requests_view`. Використовуються дані авторизованого відвідувача.

Перегляд власних відповідей на запити відвідувачів на персональні екскурсії реалізовано функцією `tourguide_requests_view`.

Перегляд запиту відвідувача на персональну екскурсію реалізовано функцією `visitor_request_view`, в якій параметром `personalrequest` визначається номер запиту, а за цим запитом окрім нього самого (даних запиту), якщо авторизований відвідувач є також і власником цього запиту, також створюється перелік відповідей, які було отримано від екскурсоводів на цей запит.

Перегляд варіантів екскурсії реалізовано функцією `excursion_variants_view`, яка за екскурсією з номером `excursion` визначає всі варіанти екскурсій, які для неї визначені. Це формування сторінки, яка формує голосування. Саме голосування реалізовано функцією `def excursionvariant_vote_view`, яка значенням параметра `excursionvariant` встановлює номер варіанта, за який авторизований відвідувач віддав голос.

Створення екскурсії реалізовано функцією `new_excursion_view`, яка не має параметрів, а дані отримує через клас форми `ExcursionForm`, в якій визначено всі дані екскурсії разом з даними моделі на основі зв'язків багато-до-багатьох, а атрибут, який визначає екскурсовода, що створив екскурсію, визначається програмно на основі даних авторизованого екскурсовода.

Редагування екскурсії реалізовано функцією `edit_excursion_view`, яка має параметр `excursion`, що є номером екскурсії, дані якої мають бути змінені. Оскільки це ті самі дані, що і під час створення екскурсії, адже власник визначався програмно, то в цій функції також застосовано форму `ExcursionForm`.

Створення варіанта екскурсії реалізовано подібним чином до створення екскурсії, тому це функція без параметрів `new_excursionvariant_view` з класом форми `ExcursionvariantForm`. Відповідно редагування варіанта екскурсії виконується функцією `edit_excursionvariant_view`.

Аналогічно відбувається визначення дати проведення екскурсії. Для цього створені функції `new_excursiondata_view` для самого створення нової дати, `edit_excursiondate_view` для редагування існуючої дати та `delete_excursiondate_view`, яка за номером цієї дати (проведення екскурсії) `excursiondate` виконує вилучення її з БД. Функція `new_excursiondata_view` як параметр має номер екскурсії, до якої створюється це проведення та номер мови, якою буде проводитися ця екскурсія, – `language`.

Подання відповіді на запит відвідувача на персональну екскурсію реалізовано функцією `new_persrequestanswer_view`, яка має параметр `persrequest`, який визначає номер самого запиту, на який відповідає екскурсовод, та яка має клас форми `PersonalrequestAnswerForm`, через який реалізується робота з даними.

Визначення кількості місць на екскурсії разом з визначенням вартості цієї позиції реалізовано функцією `new_price_view`, яка має параметр `excursion`, який визначає номер екскурсії. Класом форми, через який реалізовано внесення даних, є `ExcursionPricegroupForm`. Для редагування кількості місць і вартості також створено функцію – `edit_price_view`, яка для параметра `excursionpricegroup`, який є номером цінової пропозиції, за допомогою того ж класу форми виконує редагування даних у БД.

Перегляд місць за містом реалізовано функцією `city_places_view`, параметр `city` якої визначає місто, місця за яким визначаються.

Бронювання місця на екскурсії виконується функцією `book_view`, в параметрі якої вказується номер дати проведення екскурсії `excursiondate`. Використовується клас форми `BookingForm`. Після внесення даних у БД відбувається перенаправлення до сплати за бронювання. Оброблення отримання успішного виконання платежу реалізовано через функцію

confirm_pay_view. Тоді у БД в стані бронювання зазначається, що він сплачений, а у випадку, якщо всі місця на цю дату екскурсії зайнято, то змінюється стан в даті проведення, позначаючи, що продаж завершено.

Затвердження відповіді на запит відвідувача на персональну екскурсію підтримується функцією confirm_request_view, яка параметром persrequestanswer визначає номер відповіді, яку затверджує відвідувач. Форма не використовується, адже в результаті тільки змінюється стан відповіді. Відповідне скасування підтримується функцією cancel_request_view з тим же самим параметром.

Оцінювання екскурсовода реалізовано функцією tourguide_rate_view, яка працює з параметром guide, який визначає номер екскурсовода та класом форми TourguideRateForm. Оцінювання екскурсії реалізовано функцією excursion_rate_view. Параметр excursion є номером екскурсії, яку оцінюють. Клас форми – ExcursionRateForm.

Редагування персональних даних користувача реалізовано функціями tourguide_edit_view з параметром tourguide, який є номером екскурсовода, та visitor_edit_view з параметром visitor, який є номером відвідувача, який змінює власні дані. При цьому перевіряється, що авторизований користувач є власником відповідного облікового запису, дані за яким він змінює.

Коментування екскурсії підтримується функцією excursion_comment_view, яка має параметр excursion, що є номером екскурсії, та клас форми CommentForm.

3.3 Висновки за розділом 3

У третьому розділі описано реалізацію програми за допомогою визначення параметрів, алгоритмів роботи основних функцій.

4 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ

4.1 Призначення програми

Програма підтримує два типи користувачів: відвідувачів екскурсій та екскурсоводів. Кожного з них вона забезпечує потрібними засобами, включаючи двосторонню ініціативу, тобто і екскурсоводи можуть пропонувати екскурсії, і відвідувачі можуть виставляти свої запити з пропозицією проведення для них персональної екскурсії.

Загалом програма надає екскурсоводу можливість формувати представлення своїх екскурсій, визначати дати проведення екскурсій, керувати внесеними даними, забезпечувати його можливістю розподілу місць на екскурсії через систему. Також програма надає екскурсоводу можливість порівнювати враження різних відвідувачів від різних власних екскурсій, переглядати різноманітні рейтинги і відповідну позицію власної пропозиції у них.

Відвідувачу програма надає можливість шукати, ознайомлюватися з варіантами існуючих екскурсій, обирати з них ті, які він вважає за потрібне, бронювати на них місця, створювати власні запити з проханням організувати для них персональну екскурсію у певному місті.

4.2 Умови виконання програми

Апаратні вимоги встановлені для серверної сторони, адже клієнтська тільки забезпечує доступ через браузер, тому можливості запуску браузера на комп'ютері та наявності доступу до мережі достатньо для роботи. Апаратні вимоги до сервера включають процесор не менше ніж Dual Intel Xeon 3,2 ГГц, оперативну пам'ять обсягом не менше 4 Гб, жорсткий диск з вільним доступом до 36 Гб.

4.3 Принципи експлуатації та тестування програми

Тестування програми було проведено таким чином, що воно в підсумку охоплювало перевірку роботи всіх необхідних функцій для обох ролей користувачів і для неавторизованих користувачів та перевірку роботи цих функцій у різних браузерях, включаючи фактично останню версію Google Chrome та Firefox. На фінальній стадії усі перевірки були завершені успішно.

Робота користувача з клієнтської сторони розпочинається з базової сторінки, яка відображається при зверненні до вебсервера (рис. 4.1). Такою сторінкою є рейтинг екскурсоводів.

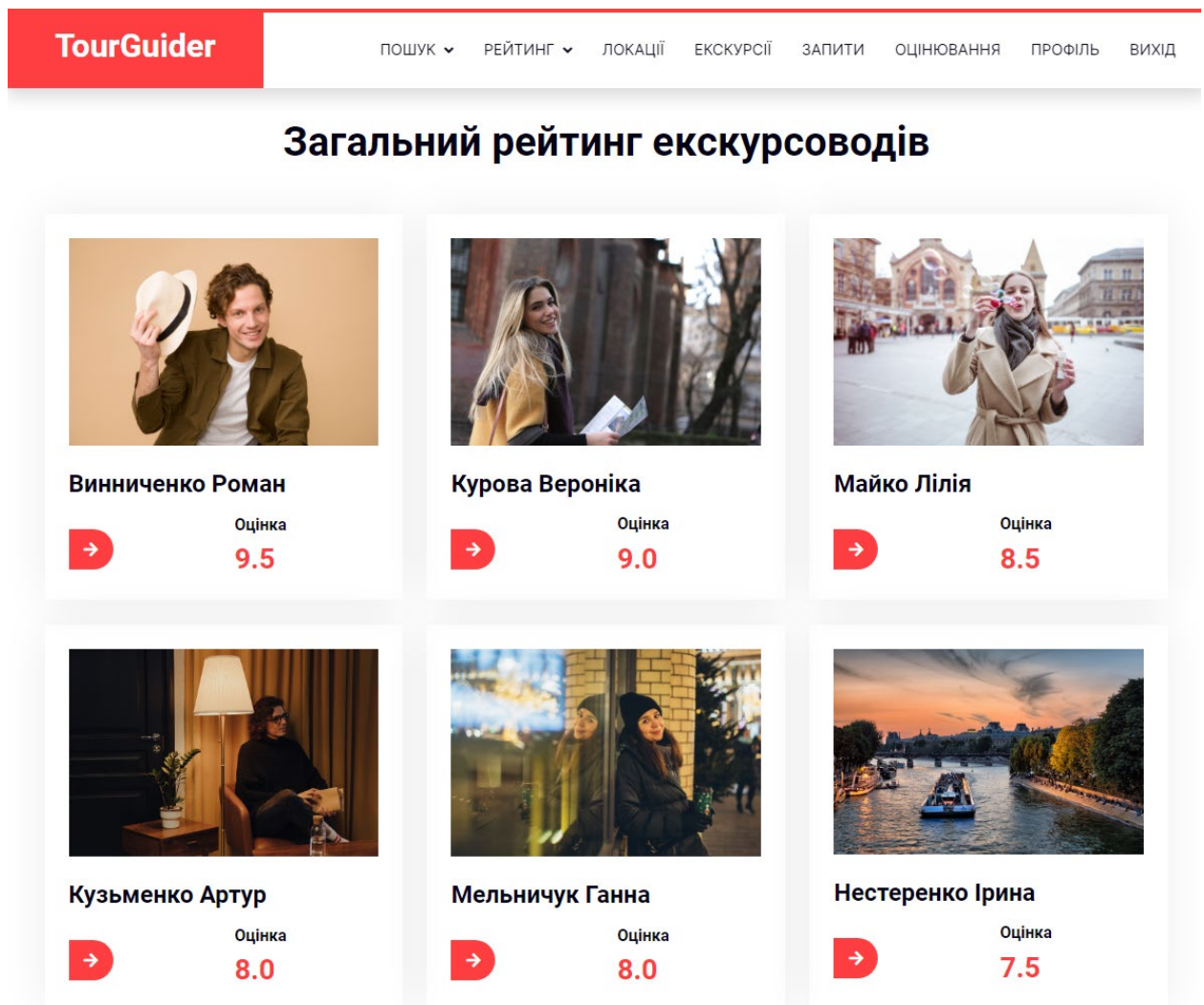


Рисунок 4.1 – Рейтинг екскурсоводів

Загалом всім користувачам у меню сторінок доступний блок “Рейтинг”, який включає рейтинг екскурсій загалом, рейтинг за містом, за країною, за місцем та рейтинг екскурсіводів загалом, рейтинг за містом, за країною, за місцем. Для всіх випадків окрім першого спочатку задається відповідно країна або країна і місто або країна, місто і місце, а далі показується за ними рейтинг відповідно екскурсій або екскурсіводів.

Сторінка рейтингу представлена першими 50 позиціями, між якими можна перемикатися через номери сторінок внизу. Якщо екскурсіводів менше у програмі, то відображається відповідно менша кількість позицій. За кожним показується фотографія екскурсівода, його ім'я та середня оцінка за підтвердженими оцінюваннями, тобто користувачами, які виконували бронювання екскурсій у цього екскурсівода. Також показується посилання для відкриття сторінки профілю цього екскурсівода.

Так само у меню всі користувачам доступний блок “Пошук”, який включає пошук екскурсій, пошук екскурсій для бронювання (з вказанням дат і максимальної вартості), пошук екскурсіводів, пошук запитів відвідувачів на персональні екскурсії.

У випадку пошуку екскурсій (рис. 4.2) видається набір результатів, кожен з яких містить назву, фотографію, ім'я екскурсівода, орієнтовну тривалість екскурсії, середню оцінку, виставлену відвідувачами, які бронювали екскурсію через вебдодаток (підтверджені). Натискаючи на посилання зі стрілкою внизу, можна відкрити сторінку з інформацією про екскурсію. Подібним варіантом доступу до цієї сторінки є рейтинг екскурсій.

Ще одним пунктом меню, який доступний всім користувачам, є “Локації”, який призводить до отримання сторінки з переліком всіх країн. З цієї сторінки можна обрати країну і відкрити сторінку з інформацією про неї та переліком міст. Обравши місто, можна відкрити сторінку з містом та місцями у цьому місці. Зі сторінки міста можна перейти до рейтингу екскурсіводів або екскурсій за цим місцем або містом або до рейтингу екскурсій.

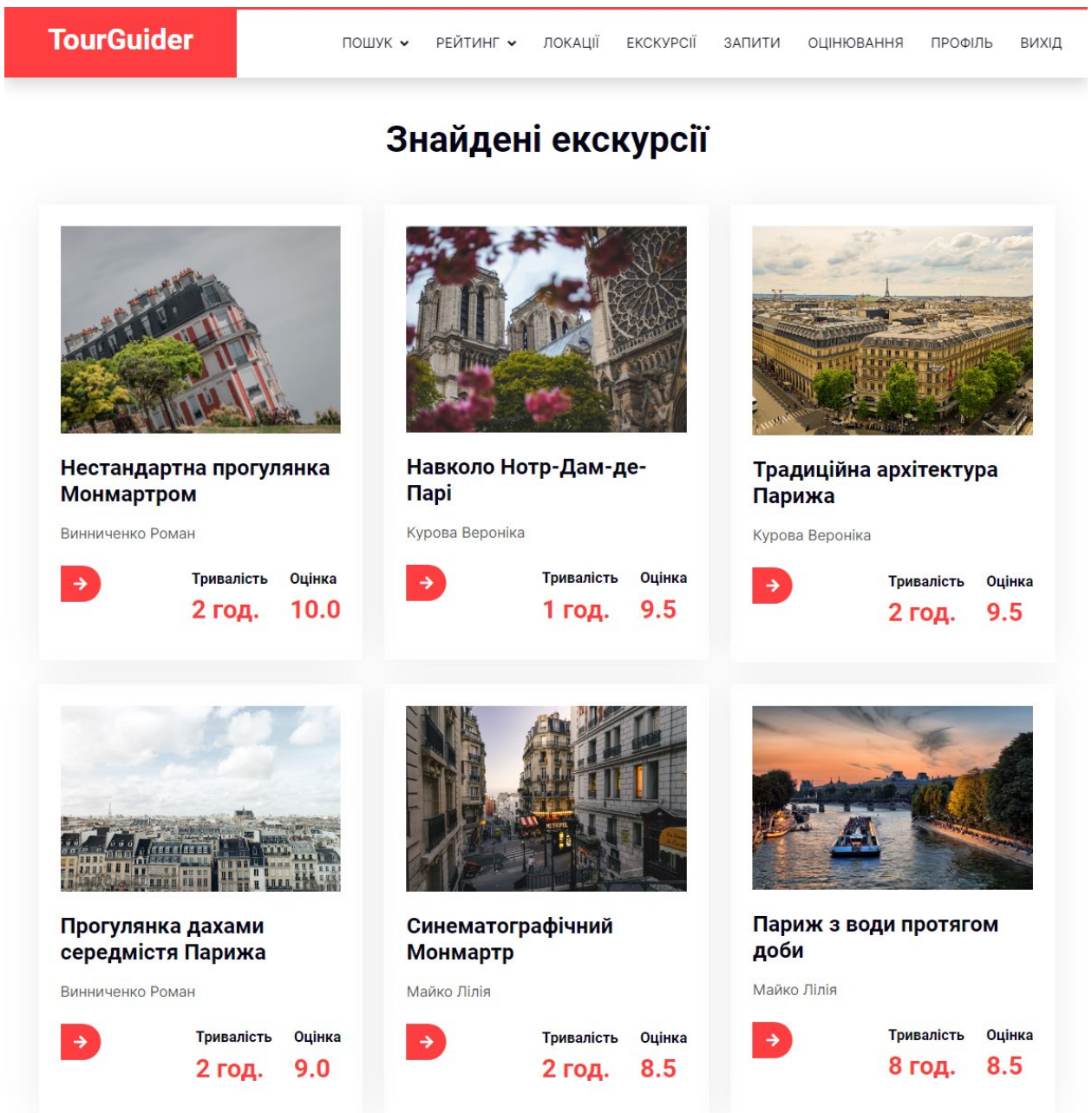


Рисунок 4.2 – Результат пошуку екскурсій

Після переміщення до сторінки з параметрами екскурсії (рис. 4.3) наверху показується фотографія та можливість переходу до інших фотографій посиланнями вліво та вправо, назва екскурсії, її опис, інформація про типи екскурсій (визначені в інтерфейсі як напрямки), наявність тільки одного міста чи декількох у екскурсії (монотур), середній бал за підтвердженими оцінюваннями та за всіма, приблизна тривалість екскурсії, місце початку екскурсії, кількість відвідувачів максимально, можливість участі організованих груп, дітей та людей з інвалідністю.

TourGuider

ПОШУК

РЕЙТИНГ

ЛОКАЦІЇ

ЕКСКУРСІЇ

ЗАПИТИ

ОЦІНЮВАННЯ

ПРОФІЛЬ

ВИХІД



ЕКСКУРСІЯ

Серце Копенгагена

Екскурсія відбувається по самому центру Копенгагена, але ми зазирнемо в дуже незвичні місця, на які рідко звертають увагу, хоча звичайно і не пропустимо все те, що на листівках Копенгагена. Тож Нюхавн – центр життя.

✓ Історія, культура Напрямки	✓ Монотур Тип
✓ 9.5 Підтверджена оцінка	✓ 8.5 Загальна оцінка
✓ 2 год. Тривалість	✓ Копенгаген, міст Нюхавн Місце зустрічі
✓ 10 Відеюдавачів	✓ Без груп Участь
✓ 3 дітьми можна Діти	✓ Безбар'єрний доступ Люди з інвалідністю

Замовити

Рисунок 4.3 – Сторінка екскурсії

Посилання “Замовити” дозволяє перейти до 3 останніх дат цієї екскурсії. Можна відкрити сторінку зі всіма датами. Також одразу потрібну дату екскурсії можна отримати через пошук екскурсій для бронювання. Нижче на сторінці приведено інформацію про план екскурсії у формі опису, важливі особливості екскурсії, вимоги для участі в екскурсії, додаткові витрати під час екскурсії, необхідність мати готівку з собою, обмеження щодо екскурсії, наявність переїздів під час проведення екскурсій, рівень фізичної активності під час даної екскурсії.

Окремо нижче містяться відтворені за посиланнями відео екскурсії, якщо вони були додані. Ще нижче знаходиться перелік мов, дані про вартість екскурсії, перелік міст і місць екскурсії, варіанти екскурсії, якщо вони

виставлені для голосування. Варіанти можна відкрити всі повністю на окремій сторінці і проголосувати за той, який відвідувач вважає за потрібне. Нижче розташовані 3 останні дати з можливістю відкрити сторінку зі всіма датами. Нижче розташовано два блоки, між яким можна перемикатися, відображаючи або коментарі, або оцінювання цієї екскурсії.

Меню “Екскурсії” дозволяє перейти до заброньованих авторизованим користувачем екскурсій. Цей пункт не показується неавторизованому користувачу. Всі створені власні запити на проведення персональної екскурсії для відвідувача доступні через меню “Запити”. Зі сторінки можна створити новий запит. Меню “Оцінювання” дозволяє переглянути всі виконані власноруч оцінювання цим відвідувачем.

Для екскурсовода перелік пунктів меню дещо інший. Пункт “Екскурсії” стосується власних екскурсій цього екскурсовода, звідки можна створювати нові екскурсії, переходити до переліку дат екскурсії, а звідти до переліку учасників екскурсії. Для екскурсовода важливим є пошук запитів відвідувачів (рис. 4.4).

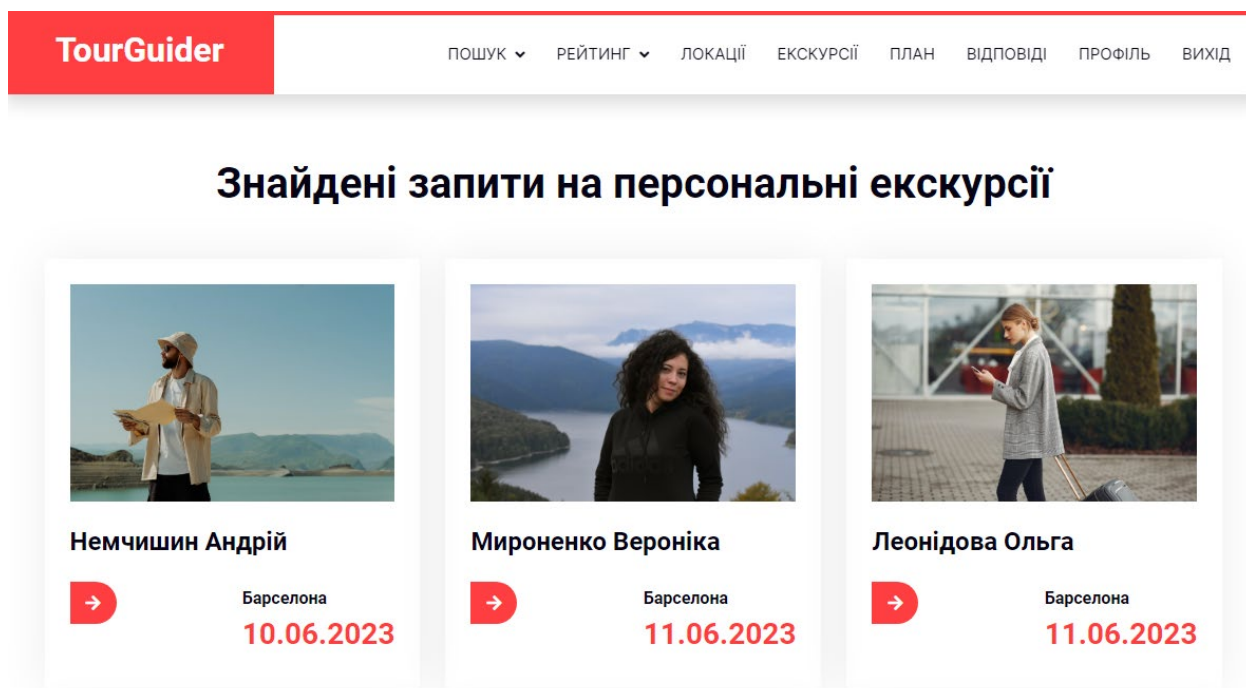


Рисунок 4.4 – Результати пошуку запитів відвідувачів на персональні екскурсії

Сторінка самої екскурсії для екскурсовода, який її створив, містить кнопку редагування, це ж містять і окремі позиції, включаючи дати проведення та цінові пропозиції. Цінові пропозиції за кожною містять певну інформацію та можливість перейти для редагування цієї пропозиції через стрілочку (рис. 4.5). Для відвідувача це посилання перенаправляє на сторінку бронювання.

The screenshot shows the 'TourGuider' website interface. At the top is a navigation bar with the logo and links: ПОШУК, РЕЙТИНГ, ЛОКАЦІЇ, ЕКСКУРСІЇ, ПЛАН, ВІДПОВІДІ, ПРОФІЛЬ, ВИХІД. Below the navigation bar, the page title is 'ЕКСКУРСІЯ: ПАРКИ КИЄВА' and the main heading is 'Цінові групи квитків'. There are three pricing cards:

Базовий тариф	Подарунковий тариф	Тариф для груп
€300 / місце	€500 / місце	€250 / місце
✓ 20 місць - Включено: ✓ емоції ✓ історії	✓ 5 місць - Включено: ✓ емоції ✓ історії ✓ подарунки ✓ плащ від дощу	✓ 10 місць - Включено: ✓ емоції ✓ історії ✓ від 5 людей
→	→	→

Рисунок 4.5 – Цінові пропозиції квитків на екскурсію

Меню “План” містить за графіком (датами) розподілені екскурсії екскурсовода. Пункт “Відповіді” містить перелік відповідей цього екскурсовода на запити, які були створені відвідувачами. Звідси можна побачити, чи були підтверджені, чи скасовані відповіді екскурсовода відвідувачем.

4.4 Висновки за розділом 4

Реалізацію всіх функцій програми перевірено через отримані результати, включаючи як коректне відображення в різних браузерах, так і перевірку результатів у БД та відображених відповідним чином в інтерфейсі, як очікувалось. Перевірку пройдено успішно.

ВИСНОВКИ

За результатами виконання роботи розроблено ПЗ пошуку і рейтингування екскурсоводів для підвищення зручності роботи екскурсоводів та відвідувачів екскурсій стосовно взаємодії між ними.

У першому розділі проаналізовано ПЗ що може застосовуватися для пошуку і рейтингування екскурсоводів, включаючи зокрема TourHQ, GetYourGuide, Showaround. Усі вони підтримують можливості бронювання, але відрізняються в направленості пошуку, в підходах до рейтингування. Ці підходи було вирішено об'єднати, розширити і розвинути у програмі, яка розроблялась.

У другому розділі забезпечено виконання наступних завдань: визначено мову і засоби розробки, що охоплюють мову python, систему для роботи з БД MySQL, базові мови веброботи, включаючи HTML, CSS, javascript, визначено вимоги до ПЗ і представлено діаграми з визначенням сценаріїв роботи користувачів, визначено структуру роботи з даними за допомогою фактично логічної схеми БД та її опису.

У третьому розділі описано реалізацію програми за допомогою визначення параметрів, алгоритмів роботи основних функцій. У четвертому розділі реалізацію всіх функцій програми перевірено через отримані результати, включаючи як коректне відображення в різних браузерах, так і перевірку результатів у БД та відображених відповідним чином в інтерфейсі, як очікувалось. Перевірку пройдено успішно.

Програма підтримує два типи користувачів: відвідувачів екскурсій та екскурсоводів. Кожного з них вона забезпечує потрібними засобами, включаючи двосторонню ініціативу, тобто і екскурсоводи можуть пропонувати екскурсії, і відвідувачі можуть виставляти свої запити з пропозицією проведення для них персональної екскурсії.

Загалом програма надає екскурсоводу можливість формувати представлення своїх екскурсій, визначати дати проведення екскурсій,

керувати внесеними даними, забезпечувати його можливістю розподілу місць на екскурсії через систему. Також програма надає екскурсоводу можливість порівнювати враження різних відвідувачів від різних власних екскурсій, переглядати різноманітні рейтинги і відповідну позицію власної пропозиції у них.

Відвідувачу програма надає можливість шукати, ознайомлюватися з варіантами існуючих екскурсій, обирати з них ті, які він вважає за потрібне, бронювати на них місця, створювати власні запити з проханням організувати для них персональну екскурсію у певному місті.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. The Most Surprising Tour Guide Industry Statistics And Trends in 2023 [Electronic resource]. – Access mode : <https://blog.gitnux.com/tour-guide-industry-statistics/>.
2. Explore the world with tourHQ [Electronic resource]. – Access mode : <https://www.tourhq.com/>.
3. ToursByLocals – Private Tours By Local Guides [Electronic resource]. – Access mode : <https://www.toursbylocals.com/>.
4. Find Local Tour Guides and Private Tours with Private Guide World at www.pg.world [Electronic resource] : The Exhaustive List of Local Tour Guides' Excursions to choose. – Access mode : <https://pg.world/>.
5. Tours by locals. Meet your local guide with showaround.com [Electronic resource]. – Access mode : <https://www.showaround.com/>.
6. Book Things To Do, Attractions, and Tours. GetYourGuide [Electronic resource]. – Access mode : <https://www.getyourguide.com/>.
7. GuidesFinder is your best chance to find a local tour guide by yourself [Electronic resource]. – Access mode : <https://guidesfinder.com/>.
8. Ernesti, J. Python 3 [Text] : The Comprehensive Guide to Hands-On Python Programming / Johannes Ernesti, Peter Kaiser. – Bonn : Rheinwerk Computing, 2022. – 1078 p.
9. Васильєв, О. Програмування мовою Python [Текст] / О. Васильєв. – Тернопіль : Навчальна книга – Богдан, 2019. – 504 с.
10. Logistica Free Website Template. Free CSS Templates. Free CSS [Electronic resource]. – Access mode : <https://www.free-css.com/free-css-templates/page292/logistica>.

ДОДАТОК А
Технічне завдання

Вступ

Туристична галузь базується не тільки на роботі туристичних агентств, але і індивідуальних працівників, зокрема екскурсоводів. Екскурсія – один з найкращих способів дізнатися про місто більше або познайомитись з ним. Екскурсії відбуваються на різну тематику, але загалом проведення екскурсій місцевими ентузіастами зокрема останнім часом активно привертає увагу багатьох туристів. Нерідко такі екскурсії містять більш особливу інформацію, є більш персоналізованими, надають новий погляд, який у масових екскурсіях туристичних агентств, які постійно повторюються за одним і тим самим сценарієм, не представлений. Але за такого підходу виникає проблема: відвідувачі екскурсій є окремими особами, екскурсоводи також ніяким чином не об'єднані, тому як в багатьох сучасних рішеннях потрібна програмна система, яка дозволить об'єднати їх разом, щоб екскурсоводи могли сформулювати пропозицію, а відвідувачі – знайти ту пропозицію, яка найкраще відповідає їхнім інтересам. Саме таке ПЗ, до того ж розширене за рахунок виділення в подальшому інших особливостей, потрібно створити в роботі.

A.1 Підстави для розробки

Завдання на дипломну кваліфікаційну роботу сформульовано у відповідності з наказом № 87 від 4 квітня 2023 р. за Національним університетом «Запорізька політехніка», яким визначено тему «Програмне забезпечення пошуку і рейтингування екскурсоводів».

A.2 Призначення розробки

Програма має повноцінно підтримувати два типи користувачів: відвідувачів екскурсій та екскурсоводів. Кожного з них вона має забезпечувати потрібними засобами, включаючи двосторонню ініціативу,

тобто щонайменше розглянувши в роботі можливість ініціативи з проведення екскурсій не тільки самим екскурсоводом. Загалом програма має надавати екскурсоводу можливість формувати представлення своїх екскурсій у програмі, визначати дати проведення екскурсій, керувати внесеними даними, забезпечувати його можливістю розподілу місць на екскурсії через систему. Також програма має надавати екскурсоводу можливість порівнювати враження різних відвідувачів від різних власних екскурсій, переглядати різноманітні рейтинги і відповідну позицію власної пропозиції у них. Відвідувачу програма має надавати можливість шукати, ознайомлюватися з варіантами існуючих екскурсій, обирати з них ті, які він вважає за потрібне, бронювати на них місця.

A.3 Основні вимоги до програми

A.3.1 Вимоги до функціональних характеристик

Функціональні характеристики програми мають бути забезпечені за таким переліком:

- пошук екскурсій;
- пошук екскурсій для бронювання;
- пошук екскурсоводів;
- пошук запитів відвідувачів на персональні екскурсії;
- створення екскурсії;
- редагування екскурсії;
- визначення дати проведення екскурсії;
- редагування або вилучення дати проведення екскурсії;
- авторизація і реєстрація;
- оцінювання екскурсовода;
- оцінювання екскурсії;
- перегляд рейтингу екскурсоводів загалом, за містом, за країною, за місцем;

- перегляд рейтингу екскурсій загалом, за містом, за країною, за місцем;
- перегляд профілю екскурсовода;
- перегляд профілю відвідувача;
- редагування персональних даних користувача;
- перегляд екскурсій, які проводить екскурсовод;
- бронювання місця на екскурсії;
- перегляд всіх учасників екскурсії екскурсоводом;
- перегляд заброньованих екскурсій;
- коментування екскурсії;
- голосування за варіант екскурсії;
- перегляд варіантів екскурсії;
- створення варіанта екскурсії;
- редагування варіанта екскурсії;
- подання відповіді на запит відвідувача на персональну екскурсію;
- затвердження або скасування відповіді на запит відвідувача на персональну екскурсію;
- перегляд запитів відвідувача на персональну екскурсію;
- перегляд власних відповідей на запити відвідувачів на персональні екскурсії;
- перегляд плану проведення екскурсії;
- перегляд плану проведення всіх екскурсій екскурсоводом;
- визначення кількості місць на екскурсії;
- перегляд власних оцінювань;
- перегляд країн;
- перегляд міст за країнами;
- перегляд місць за містом;
- перегляд параметрів виконуваної екскурсії;
- перегляд запиту відвідувача на персональну екскурсію.

A.3.2 Вимоги до складу та параметрів технічних засобів

Апаратні вимоги встановлюються для серверної сторони, адже клієнтська тільки забезпечує доступ через браузер, тому можливості запуску браузера на комп'ютері та наявності доступу до мережі достатньо для роботи. Апаратні вимоги до сервера включають процесор не менше ніж Dual Intel Xeon 3,2 ГГц, оперативну пам'ять обсягом не менше 4 Гб, жорсткий диск з вільним доступом до 36 Гб.

A.4 Порядок контролю та приймання

На виконання і захист дипломної кваліфікаційної роботи відводиться строк у 8 тижнів. Розподіл етапів роботи за тижнями визначений у календарному плані виданого завдання на роботу.

ДОДАТОК Б
Текст програми

Б.1 Текст файла `displayviews.py`

```

from .models import Excursion
from django.shortcuts import render
from .models import City
from .models import Country
from .models import Place
from .models import ExcursionLanguage
from .models import ExcursionCity
from .models import ExcursionPlace
from .models import Language

def find_excursions_view(request, city = None, country = None, place =
None, rating_from = 1, duration_from = None, duration_to = None, mono = None,
transportation = None, activitylevel_from = None, activity_to = None, group =
None, participants = 1, kids = None, pay = None, disabled = None, excursiontypes
= None, languages = None, live = None, cities = None, cities_time = None,
city_and = False, places = None, places_time = None):
    excursions = Excursion.objects.all()
    if city:
        excursions = excursions.filter(cities__id__exact = city)
    if country:
        excursions = excursions.filter(cities__country__id__exact = country)
    if place:
        excursions = excursions.filter(places__id__exact = place)
    if rating_from:
        excursions = excursions.filter(verifiedrating__gte = rating_from)
    if duration_from:
        excursions = excursions.filter(duration__gte = duration_from)
    if duration_to:

```

```
excursions = excursions.filter(duration__lte = duration_to)
if mono == 1:
    excursions = excursions.filter(monotour = True)
elif mono == 2:
    excursions = excursions.filter(monotour = False)
if transportation == 1:
    excursions = excursions.filter(transportation = True)
elif transportation == 2:
    excursions = excursions.filter(transportation = False)
if activitylevel_from:
    excursions = excursions.filter(activitylevel__gte = activitylevel_from)
if activity_to:
    excursions = excursions.filter(activitylevel__lte = activity_to)
if group == 1:
    excursions = excursions.filter(groups = True)
elif group == 2:
    excursions = excursions.filter(groups = False)
if participants:
    excursions = excursions.filter(participants__gte = participants)
if kids == 1:
    excursions = excursions.filter(kids = True)
elif kids == 2:
    excursions = excursions.filter(kids = False)
if pay == 1:
    excursions = excursions.filter(pay = True)
elif pay == 2:
    excursions = excursions.filter(pay = False)
if disabled == 1:
    excursions = excursions.filter(disabled = True)
elif disabled == 2:
```

```

        excursions = excursions.filter(disabled = False)
    if excursiontypes:
        excursions = excursions.filter(excursiontypes__id__in =
excursiontypes)
    language_selected_excursions = []
    if languages:
        for excursion in excursions:
            language_excursion = ExcursionLanguage.objects.filter(excursion
= excursion, language__in = languages)
            if language_excursion:
                language_selected_excursions.append(excursion)
    city_selected_excursions = []
    if cities:
        for excursion in excursions:
            i = 0
            found = True
            for city in cities:
                city_excursion = ExcursionCity.objects.filter(excursion =
excursion, city__in = cities, duration__gte = cities_time[i])
                if city_excursion and not city_and:
                    city_selected_excursions.append(excursion)
                    break
                elif not city_excursion:
                    found = False
                    break
            if found:
                city_selected_excursions.append(excursion)
            i = i + 1
    place_selected_excursions = []
    if places:

```

```

    for excursion in city_selected_excursions:
        i = 0
        found = True
        for place in places:
            place_excursion = ExcursionPlace.objects.filter(excursion =
excursion, place__in = places, duration__gte = places_time[i])
            if place_excursion and not city_and:
                place_selected_excursions.append(excursion)
                break
            elif not place_excursion:
                found = False
                break
            if found:
                place_selected_excursions.append(excursion)
        i = i + 1
    excursions = place_selected_excursions.order_by("-verifiedrating")
    cities = City.objects.select_related("country").all()
    places =
Place.objects.select_related("city").select_related("city__country").all()
    languages = Language.objects.all()
    countries = Country.objects.all()
    return render(request, "tourguider/excfind.html", {"excursions":
excursions, "cities": cities, "places": places, "languages": languages, "countries":
countries })

```

```

from .models import ExcursionDate
from .models import ExcursionPricegroup

```

```

def find_booked_excursions_view(request, excursiondate, city = None,
country = None, place = None, rating_from, duration_from, duration_to, mono =

```

```

None, transportation = None, activitylevel_from = None, activity_to = None, group
= None, participants = 1, kids = None, pay = None, disabled = None,
excursiontypes = None, languages = None, live = None, cities = None, cities_time
= None, city_and = False, places = None, places_time = None, time_begin = None,
time_end = None, price = None, tickets = 1):

```

```

    excursions = Excursion.objects.all()

```

```

    if city:

```

```

        excursions = excursions.filter(cities__id__exact = city)

```

```

    if country:

```

```

        excursions = excursions.filter(cities__country__id__exact = country)

```

```

    if place:

```

```

        excursions = excursions.filter(places__id__exact = place)

```

```

    if rating_from:

```

```

        excursions = excursions.filter(verifiedrating__gte = rating_from)

```

```

    if duration_from:

```

```

        excursions = excursions.filter(duration__gte = duration_from)

```

```

    if duration_to:

```

```

        excursions = excursions.filter(duration__lte = duration_to)

```

```

    if mono == 1:

```

```

        excursions = excursions.filter(monotour = True)

```

```

    elif mono == 2:

```

```

        excursions = excursions.filter(monotour = False)

```

```

    if transportation == 1:

```

```

        excursions = excursions.filter(transportation = True)

```

```

    elif transportation == 2:

```

```

        excursions = excursions.filter(transportation = False)

```

```

    if activitylevel_from:

```

```

        excursions = excursions.filter(activitylevel__gte = activitylevel_from)

```

```

    if activity_to:

```

```

        excursions = excursions.filter(activitylevel__lte = activity_to)

```

```

if group == 1:
    excursions = excursions.filter(groups = True)
elif group == 2:
    excursions = excursions.filter(groups = False)
if participants:
    excursions = excursions.filter(participants__gte = participants)
if kids == 1:
    excursions = excursions.filter(kids = True)
elif kids == 2:
    excursions = excursions.filter(kids = False)
if pay == 1:
    excursions = excursions.filter(pay = True)
elif pay == 2:
    excursions = excursions.filter(pay = False)
if disabled == 1:
    excursions = excursions.filter(disabled = True)
elif disabled == 2:
    excursions = excursions.filter(disabled = False)
if excursiontypes:
    excursions = excursions.filter(excursiontypes__id__in =
excursiontypes)
language_selected_excursions = []
if languages:
    for excursion in excursions:
        language_excursion = ExcursionLanguage.objects.filter(excursion
= excursion, language__in = languages)
        if language_excursion:
            language_selected_excursions.append(excursion)
city_selected_excursions = []
if cities:

```

```

for excursion in excursions:
    i = 0
    found = True
    for city in cities:
        city_excursion = ExcursionCity.objects.filter(excursion =
excursion, city__in = cities, duration__gte = cities_time[i])
        if city_excursion and not city_and:
            city_selected_excursions.append(excursion)
            break
        elif not city_excursion:
            found = False
            break
        if found:
            city_selected_excursions.append(excursion)
    i = i + 1
place_selected_excursions = []
if places:
    for excursion in city_selected_excursions:
        i = 0
        found = True
        for place in places:
            place_excursion = ExcursionPlace.objects.filter(excursion =
excursion, place__in = places, duration__gte = places_time[i])
            if place_excursion and not city_and:
                place_selected_excursions.append(excursion)
                break
            elif not place_excursion:
                found = False
                break
            if found:

```

```

        place_selected_excursions.append(excursion)
        i = i + 1
    excursion_dates_results = []
    for excursion in place_selected_excursions:
        excursion_dates =
ExcursionDate.objects.prefetch_related("excursionlanguage").prefetch_related("ex
cursionlanguage__excursion").prefetch_related("excursionlanguage__language").fi
lter(excursionlanguage__excursion = excursion, excursionlanguage__language__in
= languages, date = excursiondate, timestart__gte = time_begin, timestart__lte =
time_end, state = True)
        for excursion_date in excursion_dates:
            excursion_end = excursion.timestart +
excursion_date.excursionlanguage.excursion.duration
            if excursion_end >= time_begin and excursion_end <= time_end:
                excursion_dates_results.append(excursion_date)
    for excursion in excursion_dates_results:
        groups = ExcursionPricegroup.objects.filter(excursion =
excursion.excursionlanguage.excursion, price__lte = price)
        group_ready = False
        for group in groups:
            bookings = Booking.objects.filter(excursiondate = excursion, price
= group.price, state = True)
            sum = 0
            for booking in bookings:
                sum = sum + booking.participants
            available = group.participants - sum
            if available >= tickets:
                group_ready = True
        if group_ready:
            excursion_dates_result.append(excursion)

```

```

        excursion_dates_result.sort(reverse=True, key = lambda x:
x.excursionlanguage.excursion.verifiedrating)
        cities = City.objects.select_related("country").all()
        places =
Place.objects.select_related("city").select_related("city__country").all()
        languages = Language.objects.all()
        countries = Country.objects.all()
        return render(request, "tourguider/excbookedfind.html", {"excursions":
excursion_dates_result, "cities": cities, "places": places, "languages": languages,
"countries": countries })

```

```

from .models import Tourguide

```

```

def find_tourguides_view(request, city = None, country = None, place =
None, experience = None, rating_from = None, excursiontypes = None, languages
= None):
    tourguides = Tourguide.objects.all()
    if city:
        city_tour_guides =
ExcursionCity.objects.select_related("excursion").select_related("excursion__tour
guide").filter(city__id = city).distinct("excursion__tourguide")
        tourguides = tourguides.filter(id__in = city_tour_guides)
    if country:
        country_tour_guides =
ExcursionCity.objects.select_related("excursion").select_related("excursion__tour
guide").select_related("city__country").filter(city__country__id =
country).distinct("excursion__tourguide")
        tourguides = tourguides.filter(id__in = country_tour_guides)
    if place:

```

```

        place_tour_guides =
ExcursionPlace.objects.select_related("excursion").select_related("excursion__tour
guide").filter(place__id = place).distinct("excursion__tourguide")
        tourguides = tourguides.filter(id__in = place_tour_guides)
        if languages:
            language_tour_guides =
ExcursionLanguage.objects.select_related("excursion").select_related("excursion__
_tourguide").select_related("language").filter(language__in =
languages).distinct("excursion__tourguide")
            tourguides = tourguides.filter(id__in = language_tour_guides)
        if experience:
            search_year = datetime.datetime.now() - datetime.timedelta(years =
experience)
            tourguides = tourguides.filter(experience__lte = search_year)
        if rating_from:
            tourguides = tourguides.filter(verifiedrating__gte = rating_from)
        if excursiontypes:
            type_tour_guides =
Excursion.objects.select_related("tourguide").prefetch_related("types").filter(types
__in = excursiontypes).distinct("excursion__tourguide")
            tourguides = tourguides.filter(id__in = type_tour_guides)
        tourguides = tourguides.order_by("-verifiedrating")
        languages = Language.objects.all()
        countries = Country.objects.all()
        cities = City.objects.select_related("country").all()
        places =
Place.objects.select_related("city").select_related("city__country").all()
        return render(request, "tourguider/guidefind.html", {"guides":
tourguides, "cities": cities, "places": places, "countries": countries, "languages":
languages })

```

```

def find_persrequests_view(request, city = None, country = None,
date_from = None, date_to = None, duration_from = None, duration_to = None,
language = None):
    personal_requests =
Personalrequest.objects.select_related("city").select_related("language").select_rel
ated("city__country").filter(state = True)
    if city:
        personal_requests = personal_requests.filter(city__id = city)
    if country:
        personal_requests = personal_requests.filter(city__country__id =
country)
    if date_from:
        personal_requests = personal_requests.filter(date__gte = date_from)
    if date_to:
        personal_requests = personal_requests.filter(date__lte = date_to)
    if duration_from:
        personal_requests = personal_requests.filter(duration__gte =
duration_from)
    if duration_to:
        personal_requests = personal_requests.filter(duration__gte =
duration_to)
    if language:
        personal_requests = personal_requests.filter(language__id = language)
    personal_requests = personal_requests.order_by("-creation")
    languages = Language.objects.all()
    countries = Country.objects.all()
    cities = City.objects.select_related("country").all()
    return render(request, "tourguider/requestfind.html", {"requests":
personal_requests, "cities": cities, "countries": countries, "languages": languages })

```

```
def tourguides_rating_view(request):
    tourguides = Tourguide.objects.select_related("city").all().order_by("-verifiedrating")[:50]
    return render(request, "tourguider/guiderating.html", {"tourguides": tourguides })
```

```
def city_tourguides_rating_view(request, city):
    tourguides = Tourguide.objects.select_related("city").filter(city__id = city).order_by("-verifiedrating")[:50]
    countries = Country.objects.all()
    cities = City.objects.select_related("country").all()
    return render(request, "tourguider/cityguiderating.html", {"tourguides": tourguides, "cities": cities, "countries": countries })
```

```
def city_excurses_tourguides_rating_view(request, city):
    excursions =
Excursion.objects.select_related("tourguide").prefetch_related("cities").filter(cities__id__exact = city).order_by("-verifiedrating")
    tourguides = []
    for excursion in excursions:
        if excursion.tourguide not in tourguides:
            tourguides.append(excursion.tourguide)
    countries = Country.objects.all()
    cities = City.objects.select_related("country").all()
    return render(request, "tourguider/cityaltguiderating.html", {"tourguides": tourguides, "cities": cities, "countries": countries })
```

```
def country_tourguides_rating_view(request, country):
```

```

    tourguides =
Tourguide.objects.select_related("city").select_related("city__country").filter(city__country__id = country).order_by("-verifiedrating")[:50]
    countries = Country.objects.all()
    return render(request, "tourguider/countryguiderating.html",
{"tourguides": tourguides, "countries": countries })

```

```

def place_tourguides_rating_view(request, place):
    excursions =
Excursion.objects.select_related("tourguide").prefetch_related("cities").filter(place__id__exact = place).order_by("-verifiedrating")
    tourguides = []
    for excursion in excursions:
        if excursion.tourguide not in tourguides:
            tourguides.append(excursion.tourguide)
    countries = Country.objects.all()
    cities = City.objects.select_related("country").all()
    places =
Place.objects.select_related("city").select_related("city__country").all()
    return render(request, "tourguider/cityaltguiderating.html",
{"tourguides": tourguides, "cities": cities, "countries": countries, "places": places
})

```

```

def excursion_rating_view(request):
    excursions =
Excursion.objects.select_related("tourguides").prefetch_related("cities").prefetch_r
elated("places").all().order_by("-verifiedrating")[:50]
    return render(request, "tourguider/excrating.html", {"excursions":
excursions })

```

```

def city_excursion_rating_view(request, city):
    excursions =
Excursion.objects.select_related("tourguides").prefetch_related("cities").prefetch_r
elated("places").filter(cities__id__exact = city).order_by("-verifiedrating")[:50]
    countries = Country.objects.all()
    cities = City.objects.select_related("country").all()
    return render(request, "tourguider/cityexcrationg.html", {"tourguides":
tourguides, "cities": cities, "countries": countries })

```

```

def country_excursion_rating_view(request, country):
    excursions =
Excursion.objects.select_related("tourguides").prefetch_related("cities").prefetch_r
elated("places").filter(cities__country__id__exact = country).order_by("-
verifiedrating")[:50]
    countries = Country.objects.all()
    return render(request, "tourguider/countryexcrationg.html", {"tourguides":
tourguides, "countries": countries })

```

```

def city_places_view(request, city):
    city = City.objects.select_related("country").filter(pk = city).get()
    places =
Place.objects.select_related("city").select_related("city__country").filter(city =
city).order_by("name")
    return render(request, "tourguider/cityplaces.html", {"places": places,
"city": city })

```

```

def countries_view(request):
    countries = Country.objects.all()
    return render(request, "tourguider/countries.html", {"countries":
countries })

```

```
def country_cities_view(request, country):
    country = Country.objects.filter(pk = country).get()
    cities = City.objects.select_related("country").filter(city__country =
country).order_by("name")
    return render(request, "tourguider/countrycities.html", { "cities": cities,
"country": country })
```

```
from .models import Visitor
```

```
def ratings_view(request):
    visitor = Visitor.objects.filter(user = request.user)
    guide_ratings =
TourguideRate.objects.select_related("tourguide").filter(visitor =
visitor).order_by("-creation")
    ratings =
ExcursionRate.objects.select_related("excursion").select_related("excursion__tour
guide").prefetch_related("excursion__cities").prefetch_related("excursion__places
").filter(visitor = visitor).order_by("-creation")
    return render(request, "tourguider/ratings.html", { "ratings": ratings,
"guide_ratings": guide_ratings })
```

```
from .models import TourguideRate
```

```
from .models import ExcursionRate
```

```
def tourguide_view(request, tourguide = None):
    if tourguide:
        tourguide = Tourguide.objects.select_related("city").filter(pk =
tourguide).get()
    else:
```

```

        tourguide = Tourguide.objects.select_related("city").filter(user =
request.user).get()

        excursions_present = Excursion.objects.filter(tourguide =
tourguide).order_by("-verifiedrating")[3]

        rates = TourguideRate.objects.filter(tourguide = tourguide).order_by("-
creation")

        return render(request, "tourguider/tourguide.html", {"excursions":
excursions_present, "rates": rates, "tourguide": tourguide })

```

```

def visitor_view(request, visitor = None):
    if visitor:
        visitor = Visitor.objects.select_related("city").filter(pk = visitor).get()
    else:
        visitor = Visitor.objects.select_related("city").filter(user =
request.user).get()

        his_guide_rates = TourguideRate.objects.filter(visitor =
visitor).order_by("-creation")

        his_excursion_rates = ExcursionRate.objects.filter(visitor =
visitor).order_by("-creation")

        requests =
Personalrequest.objects.select_related("city").selelect_related("city__country").filt
er(visitor = visitor).order_by("-date")

        return render(request, "tourguider/tourguide.html", {"visitor": visitor,
"rates": his_guide_rates, "excrates": his_excursion_rates, "requests": requests })

```

```

def tourguide_excursions_view(request, tourguide = None):
    if tourguide:
        tourguide = Tourguide.objects.select_related("city").filter(pk =
tourguide).get()
    else:

```

```

        tourguide = Tourguide.objects.select_related("city").filter(user =
request.user).get()
        excursions = Excursion.objects.filter(tourguide = tourguide).order_by("-
verifiedrating")
        return render(request, "tourguider/tourguideexcursions.html",
{"excursions": excursions })

```

```

from .models import Comment
from .models import Excursionvariant
from .models import ExcursionPhoto
from .models import ExcursionVideo

def excursion_view(request, excursion):
    excursion =
Excursion.objects.select_related("tourguide").prefetch_related("types").filter(pk =
excursion).get()
    live_languages = ExcursionLanguage.objects.filter(excursion =
excursion, live = True)
    languages = ExcursionLanguage.objects.filter(excursion = excursion,
live = False)
    prices = ExcursionPricegroup.objects.filter(excursion =
excursion).order_by("price")
    comments = Comment.objects.filter(excursion = excursion).order_by("-
publication")
    cities = ExcursionCity.objects.select_related("city").filter(excursion =
excursion).order_by("position")
    places = ExcursionPlace.objects.select_related("city").filter(excursion =
excursion).order_by("position")
    places_result = []
    cities_num = len(cities)

```

```

places_num = len(places)
i = 0
j = 0
while i < cities_num and j < places_num:
    if i != cities_num and j != places_num:
        if cities[i].position < places[j].position:
            places_result.append(cities[i])
        else:
            places_result.append(places[j])
    elif i == cities_num and j != places_num:
        places_result.append(places[j])
    elif i != cities_num and j == places_num:
        places_result.append(cities[i])
variants = Excursionvariant.objects.filter(excursion = excursion)
if variants:
    vote = True
else:
    vote = False
rates = ExcursionRate.objects.filter(excursion = excursion).order_by("-
creation")
dates = ExcursionDate.objects.filter(excursionlanguage__excursion =
excursion).order_by("date")[3]
photos = ExcursionPhoto.objects.filter(excursion = excursion)
videos = ExcursionVideo.objects.filter(excursion = excursion)
return render(request, "tourguider/defexcursion.html", {"excursion":
excursion, "live_languages": live_languages, "languages": languages, "prices":
prices, "comments": comments, "places": places_result, "vote": vote, "rates": rates,
"dates": dates, "photos": photos, "videos": videos })

def excursion_dates_view(request, excursion):

```

```

        excursion = Excursion.objects.filter(pk = excursion).get()
        dates =
ExcursionDate.objects.prefetch_related("excursionlanguage__excursion").prefetch
_related("excursionlanguage__language").filter(excursionlanguage__excursion =
excursion).order_by("date")
        return render(request, "tourguider/excursiondates.html", {"dates": dates
    })

```

```

def toureguide_dates_view(request):
    tourguide = Tourguide.objects.get(user = request.user)
    dates =
ExcursionDate.objects.prefetch_related("excursionlanguage__excursion").prefetch
_related("excursionlanguage__language").filter(excursionlanguage__excursion__t
ourguide = tourguide).order_by("date")
    return render(request, "tourguider/tourguideplan.html", {"dates": dates })

```

```

def excursion_participants_view(request, excursiondate):
    tourguide = Tourguide.objects.get(user = request.user)
    excursiondate =
ExcursionDate.objects.select_related("excursionlanguage").select_related("excursi
onlanguage__excursion").select_related("excursionlanguage__excursion__tourgui
de").filter(pk = excursiondate).get(pk = excursiondate)
    if excursiondate.excursionlanguage.excursion.tourguide == tourguide:
        participants =
Booking.objects.select_related("visitor").prefetch_related("bookinggroups").filter(
excursiondate = excursiondate, state = True).order_by("creation")
    return render(request, "tourguider/participants.html", {"participants":
participants })

```

```

def visitor_excursions_view(request):

```

```

        visitor = Visitor.objects.get(user = request.user)
        bookings =
Booking.objects.select_related("visitor").prefetch_related("bookinggroups").select
_related("excursiondate").select_related("excursiondate__excursionlanguage").sele
ct_related("excursiondate__excursionlanguage__excursion").select_related("excur
siondate__excursionlanguage__excursion__tourguide").filter(visitor =
visitor).order_by("-excursiondate")
        return render(request, "tourguider/visitorexcursions.html", {"bookings":
bookings })

```

```

from .models import Personalrequest

```

```

def visitor_requests_view(request):
    visitor = Visitor.objects.get(user = request.user)
    requests =
Personalrequest.objects.select_related("city").select_related("city__country").filt
er(visitor = visitor).order_by("-date")
    return render(request, "tourguider/visitorrequests.html", {"requests":
requests })

```

```

def tourguide_requests_view(request):
    tourguide = Tourguide.objects.get(user = request.user)
    requests =
PersonalrequestAnswer.objects.select_related("personalrequest").select_related("
personalrequest__city").select_related("personalrequest__visitor").select_relate
d("personalrequest__city__country").filter(tourguide = tourguide).order_by("-
personalrequest__date")
    return render(request, "tourguider/guiderequests.html", {"requests":
requests })

```

```

def visitor_request_view(request, personalrequest):
    visitor = Visitor.objects.get(user = request.user)
    personalrequest =
Personalrequest.objects.select_related("city").select_related("visitor").select_rela
ted("city__country").filter(pk = personalrequest).get()
    answers = []
    if personalrequest.visitor == visitor:
        answers =
PersonalrequestAnswer.objects.select_related("tourguide").filter(personalrequest =
personalrequest).order_by("-creation")
    return render(request, "tourguider/request.html", {"persrequest":
personalrequest, "answers": answers })

```

```

def excursion_variants_view(request, excursion):
    excursion =
Excursion.objects.select_related("tourguide").prefetch_related("types").filter(pk =
excursion).get()
    variants =
Excursionvariant.objects.prefetch_related("cities").prefetch_related("places").prefe
tch_related("basecities").prefetch_related("baseplaces").filter(excursion =
excursion)
    return render(request, "tourguider/variants.html", {"variants": variants,
"excursion": excursion })

```

Б.2 Текст файла guidechangeviews.py

```

from .models import Excursion
from .models import Tourguide
from .forms import ExcursionForm

```

```

def new_excursion_view(request):
    guide = Tourguide.objects.filter(user = request.user)
    if guide:
        if request.method == "POST":
            excursion_form = ExcursionForm(request.POST)
            if excursion_form.is_valid():
                excursion_object = excursion_form.save(commit = False)
                excursion_object.tourguide = guide
                excursion_object.save()
                excursion_object.save_m2m()
                return redirect("excursionview", excursion_object.pk)
            else:
                excursion_form = ExcursionForm()
        else:
            excursion_form = None
    return render(request, "tourguider/excursion.html", {"excursion_form":
excursion_form})

```

```

from .models import Tourguide

```

```

def edit_excursion_view(request, excursion):
    guide = Tourguide.objects.filter(user = request.user)
    excursion = Excursion.objects.filter(pk = excursion).get()
    if guide and excursion and excursion.tourguide == guide:
        if request.method == "POST":
            excursion_form = ExcursionForm(request.POST, instance =
excursion)
            if excursion_form.is_valid():
                excursion_object = excursion_form.save(commit = False)
                excursion_object.save()

```

```

        excursion_object.save_m2m()
        return redirect("excursionview", excursion_object.pk)
    else:
        excursion_form = ExcursionForm(None, instance = excursion)
    else:
        excursion_form = None
    return render(request, "tourguider/excursion.html", {"excursion_form":
excursion_form})

```

```

from .forms import ExcursionDateForm
from .models import Language

```

```

def new_excursiondata_view(request, excursion, language):
    guide = Tourguide.objects.filter(user = request.user)
    excursion = Excursion.objects.filter(excursion = excursion)
    language = Language.objects.filter(language = language)
    language = ExcursionLanguage.objects.filter(excursion = excursion,
language = language)
    if language and excursion.tourguide == guide:
        if request.method == "POST":
            excursiondate_form = ExcursionDateForm(request.POST)
            if excursiondate_form.is_valid():
                excursiondate_object = excursiondate_form.save(commit =
False)

                excursiondate_object.excursionlanguage = language
                excursiondate_object.save()
                excursiondate_object.save_m2m()
                return redirect("excursiondatesview", excursion.pk)
            else:
                excursiondate_form = ExcursionDateForm()

```

```

else:
    excursiondate_form = None
    return render(request, "tourguider/excursiondate.html",
{"excursiondate_form": excursiondate_form})

from .models import ExcursionDate

def edit_excursiondate_view(request, excursiondate):
    guide = Tourguide.objects.filter(user = request.user)
    excursiondate =
ExcursionDate.objects.select_related("excursionlanguage").select_related("excursi
onlanguage__excursion").filter(pk = excursiondate).get()
    if guide and excursion and
excursiondate.excursionlanguage.excursion.tourguide == guide:
        if request.method == "POST":
            excursiondate_form = ExcursionDateForm(request.POST, instance
= excursiondate)
            if excursiondate_form.is_valid():
                excursiondate_object = excursiondate_form.save(commit =
False)
                excursiondate_object.save()
                excursiondate_object.save_m2m()
                return redirect("excursionview",
excursiondate.excursionlanguage.excursion.pk)
            else:
                excursiondate_form = ExcursionDateForm(None, instance =
excursiondate)
        else:
            excursiondate_form = None

```

```

        return render(request,
            "tourguider/excursiondate.html", {"excursiondate_form": excursiondate_form})

    def delete_excursiondate_view(request, excursiondate):
        guide = Tourguide.objects.filter(user = request.user)
        excursiondate =
ExcursionDate.objects.select_related("excursionlanguage").select_related("excursi
onlanguage__excursion").filter(pk = excursiondate).get()
        if guide and excursion and
excursiondate.excursionlanguage.excursion.tourguide == guide:
            excursion = excursiondate.excursionlanguage.excursion
            excursiondate.delete()
            return redirect("excursionview", excursion.pk)

from .forms import ExcursionvariantForm

def new_excursionvariant_view(request):
    guide = Tourguide.objects.filter(user = request.user)
    excursion = Excursion.objects.filter(excursion = excursion)
    if excursion and excursion.tourguide == guide:
        if request.method == "POST":
            excursionvariant_form = ExcursionvariantForm(request.POST)
            if excursionvariant_form.is_valid():
                excursionvariant_object = excursionvariant_form.save(commit =
False)

                excursionvariant_object.excursion = excursion
                excursionvariant_object.save()
                excursionvariant_object.save_m2m()
                return redirect("excursionview", excursion.pk)
        else:

```

```

        excursionvariant_form = ExcursionvariantForm()
    else:
        excursionvariant_form = None
    return render(request, "tourguider/excursionvariant.html",
{"excursionvariant_form": excursionvariant_form})

def edit_excursionvariant_view(request, excursionvariant):
    guide = Tourguide.objects.filter(user = request.user)
    excursionvariant =
Excursionvariant.objects.select_related("excursion").select_related("excursion__to
urguide").filter(pk = excursionvariant).get()
    if guide and excursionvariant and excursionvariant.excursion.tourguide
== guide:
        if request.method == "POST":
            excursionvariant_form = ExcursionvariantForm(request.POST,
instance = excursionvariant)
            if excursionvariant_form.is_valid():
                excursionvariant_object = excursionvariant_form.save(commit =
False)

                excursionvariant_object.save()
                excursionvariant_object.save_m2m()
                return redirect("excursionview", excursionvariant.excursion.pk)
            else:
                excursionvariant_form = ExcursionvariantForm(None, instance =
excursionvariant)
        else:
            excursionvariant_form = None
    return render(request,
"tourguider/excursionvariant.html", {"excursionvariant_form":
excursionvariant_form})

```

```

from .forms import PersonalrequestAnswerForm
from .models import Personalrequest

def new_persrequestanswer_view(request, persrequest):
    guide = Tourguide.objects.filter(user = request.user)
    personal_request = Personalrequest.objects.filter(pk = persrequest)
    if guide:
        if request.method == "POST":
            persreqanswer_form = PersonalrequestAnswerForm(request.POST)
            if persreqanswer_form.is_valid():
                answer_object = persreqanswer_form.save(commit = False)
                answer_object.personalrequest = personal_request
                answer_object.save()
                return redirect("persreqanswersview")
            else:
                persreqanswer_form = PersonalrequestAnswerForm()
        else:
            persreqanswer_form = None
    return render(request, "tourguider/grequestanswer.html",
{"persreqanswer_form": persreqanswer_form})

from .forms import ExcursionPricegroupForm

def new_price_view(request, excursion):
    guide = Tourguide.objects.filter(user = request.user)
    excursion = Excursion.objects.select_related("tourguide").filter(pk =
excursion)
    if guide and excursion.tourguide == guide:
        if request.method == "POST":
            price_form = ExcursionPricegroupForm(request.POST)

```

```

        if price_form.is_valid():
            price_object = price_form.save(commit = False)
            price_object.excursion = excursion
            price_object.save()
            return redirect("excursionview", excursion.pk)
        else:
            price_form = ExcursionPricegroupForm()
    else:
        price_form = None
    return render(request, "tourguider/pricegroup.html", {"price_form":
price_form})

def edit_price_view(request, excursionpricegroup):
    guide = Tourguide.objects.filter(user = request.user)
    excursionpricegroup =
ExcursionPricegroup.objects.select_related("excursion").select_related("excursion
__tourguide").filter(pk = excursionpricegroup).get()
    if guide and excursionpricegroup and
excursionpricegroup.excursion.tourguide == guide:
        if request.method == "POST":
            price_form = ExcursionPricegroupForm(request.POST, instance =
excursionpricegroup)
            if price_form.is_valid():
                price_form.save()
                return redirect("excursionview",
excursionpricegroup.excursion.pk)
            else:
                price_form = ExcursionPricegroupForm(None, instance =
excursionpricegroup)
        else:

```

```
        price_form = None
    return render(request,
        "tourguider/excursionpricegroup.html", {"excursionpricegroup_form":
        excursionpricegroup_form})
```

ДОДАТОК В
Слайди презентації

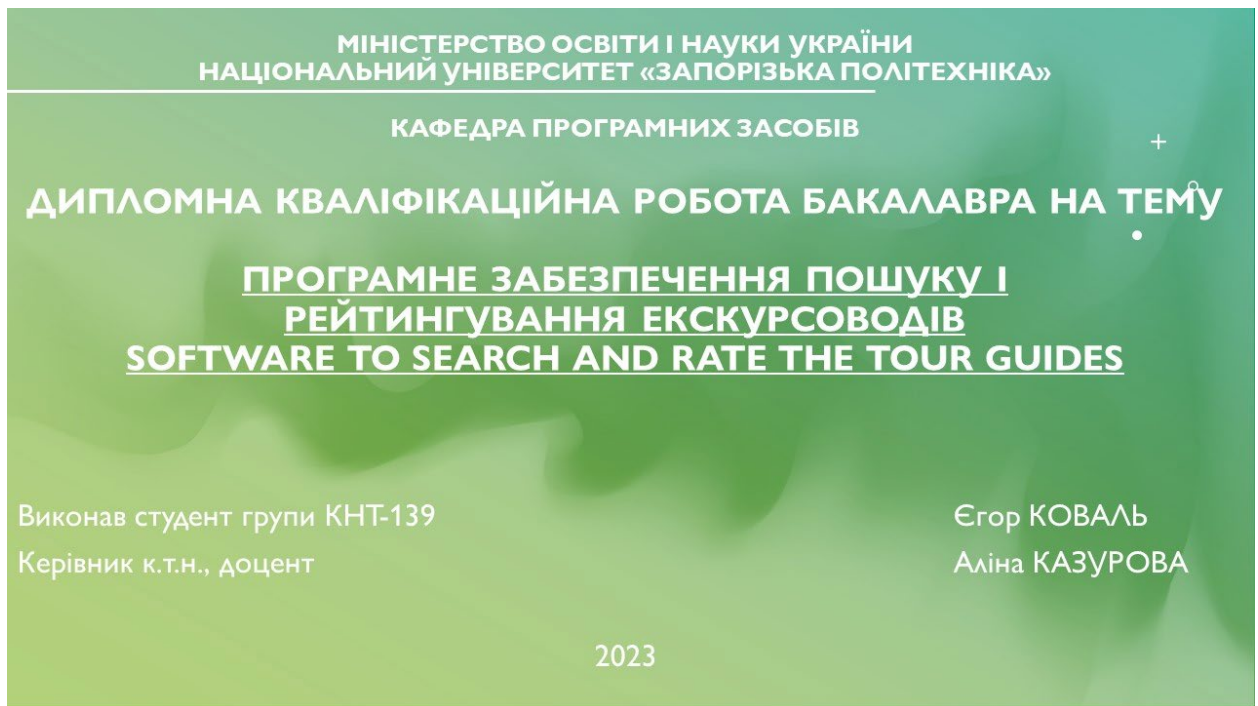


Рисунок В.1 – Початковий слайд

Об'єкт, предмет, мета роботи

- об'єкт роботи – процес підтримки проведення екскурсій;
- предмет роботи – програмне забезпечення пошуку і рейтингування екскурсоводів;
- мета роботи – розробити програмне забезпечення пошуку і рейтингування екскурсоводів для підвищення зручності роботи екскурсоводів та відвідувачів екскурсій стосовно взаємодії між ними.

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

Завдання роботи

- проаналізувати програмне забезпечення, що може застосовуватися для пошуку і рейтингування екскурсоводів;
- визначити мову і засоби розробки;
- визначити вимоги до програми;
- визначити структуру роботи з даними;
- виконати реалізацію програмного забезпечення.

3

Рисунок В.3 – Завдання роботи

Порівняння аналогів з розробкою роботи

Показник	TourHQ	GetYourGuide	Showaround	Розробка
Пошук екскурсій	Ні	Так	Ні	Так
Пошук екскурсоводів	Так	Ні	Так	Так
Бронювання екскурсій	Так	Так	Так	Так
Рейтингування екскурсій	Ні	Так	Ні	Так
Рейтингування екскурсоводів	Так	Пряме відсутнє	Так	Так
Створення оголошень про пошук екскурсовода	Ні	Ні	Ні	Так

4

Рисунок В.4 – Порівняння аналогів з розробкою роботи

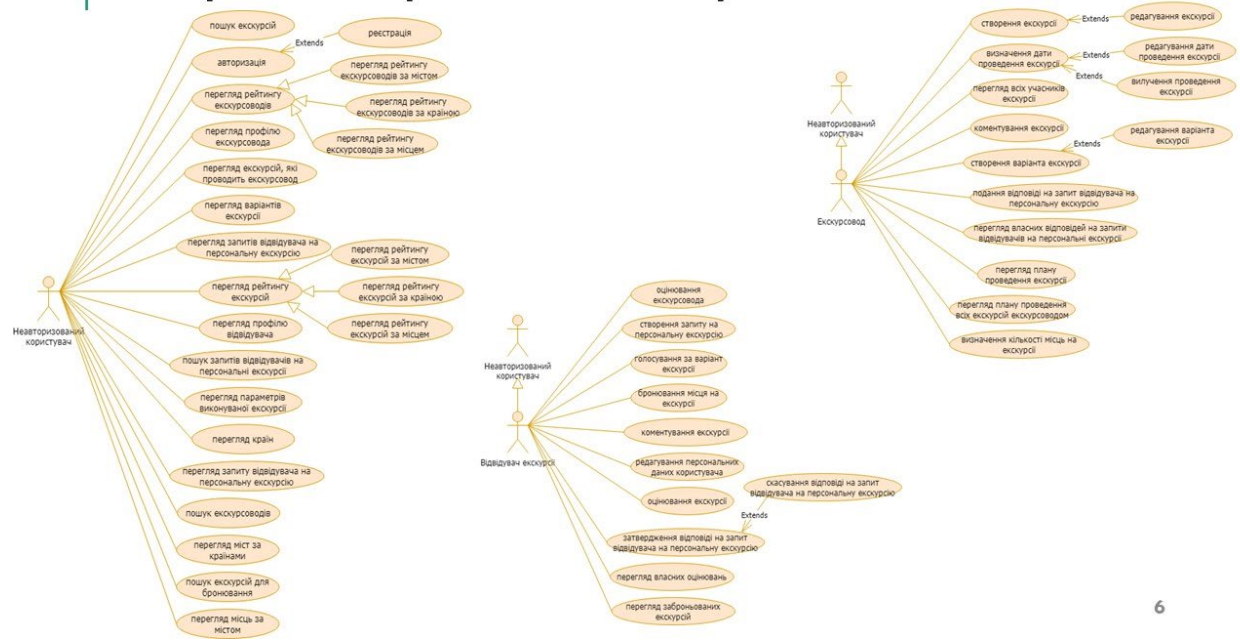
Порівняння мов веброзробки

Показник	Python	Java
Підтримувані парадигми програмування	Об'єктно-орієнтована, процедурна, функціональна	Об'єктно-орієнтована
Створення читабельного коду	Закладено в концепцію мови	Потребує додаткових зусиль, складніше
Варіант типізації	Динамічна	Сильно типізована
Продуктивність програми	Менша за java	Більша за python
Продуктивність розробки	Більша за java	Менша за python

5

Рисунок В.5 – Порівняння мов веброзробки

Діаграми варіантів використання



6

Рисунок В.6 – Діаграми варіантів використання

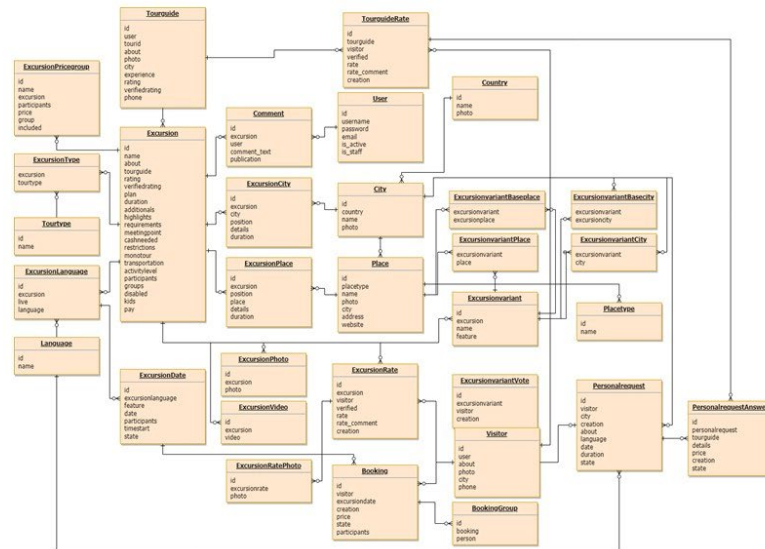


Рисунок В.7 – Проєктування структури роботи з даними

Порядок виконання пошуку екскурсоводів

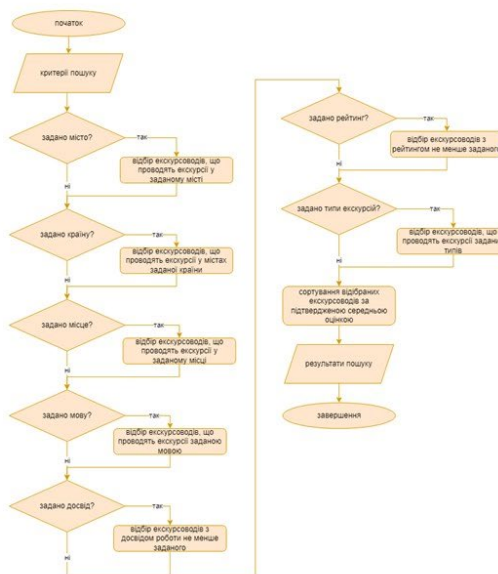
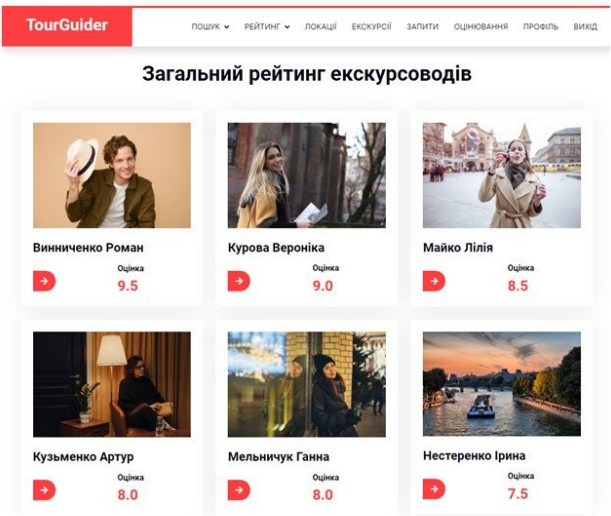


Рисунок В.8 – Порядок виконання пошуку екскурсоводів

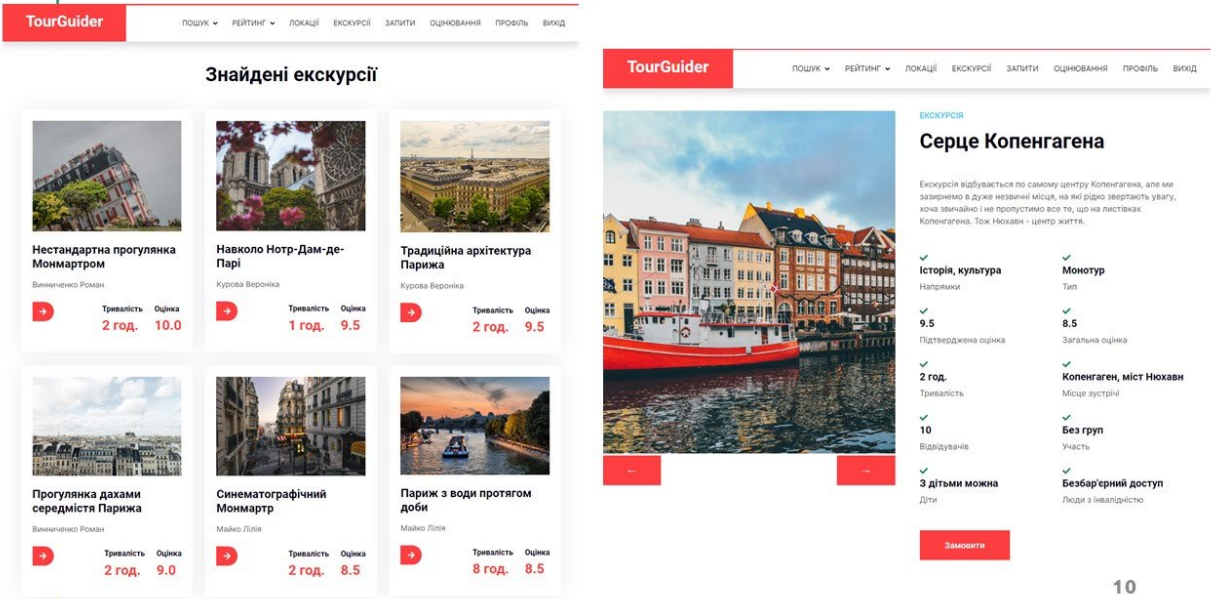
Рейтинг екскурсів у програмі



9

Рисунок В.9 – Рейтинг екскурсів у програмі

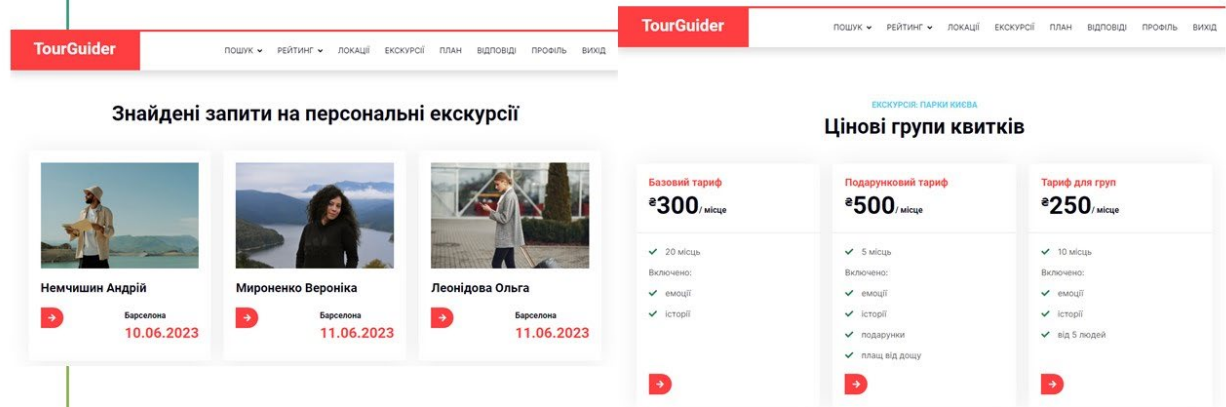
Робота з екскурсіями в програмі



10

Рисунок В.10 – Робота з екскурсіями в програмі

Сторінки екскурсовода



11

Рисунок В.11 – Сторінки екскурсовода

Висновки

- проаналізовано програмне забезпечення що може застосовуватися для пошуку і рейтингування екскурсоводів, включаючи зокрема TourHQ, GetYourGuide, Showaround: використувані у них підходи було вирішено об'єднати, розширити і розвинути у програмі;
- визначено мову і засоби розробки, що охоплюють мову python, систему для роботи з базами даних MySQL, базові мови веброзробки, включаючи HTML, CSS, javascript, визначено вимоги до програмного забезпечення і представлено діаграми з визначенням сценаріїв роботи користувачів, визначено структуру роботи з даними за допомогою логічної схеми бази даних та її опису;
- описано реалізацію програми за допомогою визначення параметрів, алгоритмів роботи основних функцій, реалізацію всіх функцій програми перевірено через отримані результати, перевірку пройдено успішно.

12

Рисунок В.12 – Висновки