Національний лісотехнічний університет України
(повне найменування вищого навчального закладу)

Навчально-науковий інститут комп'ютерних наук
та інформаційних технологій
(повне найменування інституту, назва факультету (відділення))

Кафедра комп'ютерних наук
(повна назва кафедри (предметної, циклової комісії))

**Магістерська кваліфікаційна робота**

другий (магістерський)
(рівень вищої освіти)

на тему **РОЗРОБЛЕННЯ ВЕБПЛАТФОРМИ ДЛЯ ПОПУЛЯРИЗАЦІЇ АКТИВНОГО ВІДПОЧИНКУ**

Виконав: студент 6 курсу групи КН-63м
спеціальності
122 "Комп'ютерні науки"
(шифр і назва напряму підготовки, спеціальності)

Устименко В.М.
(прізвище та ініціали)

Керівник Процах Н.П.
(прізвище та ініціали)

Рецензент_____
(прізвище та ініціали)

Львів – 2024

Національний лісотехнічний університет України
(повне найменування вищого навчального закладу)

ННІ комп'ютерних наук та інформаційних технологій
Кафедра комп'ютерних наук
Рівень вищої освіти другий (магістерський)
Спеціальність 122 "Комп'ютерні науки"
(шифр і назва)

**ЗАТВЕРДЖУЮ**
**Завідувач кафедри**
_____ Борецька І.Б.
"05" січня 2024 року

# З А В Д А Н Н Я
## НА ДИПЛОМНУ РОБОТУ СТУДЕНТУ

*Устименко Валентин Михайлович*
(прізвище, ім'я, по батькові)

1. Тема роботи РОЗРОБЛЕННЯ ВЕБПЛАТФОРМИ ДЛЯ ПОПУЛЯРИЗАЦІЇ АКТИВНОГО ВІДПОЧИНКУ

керівник роботи *Процах Наталія Петрівна, професор, доктор фізико-математичних наук*,
( прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від "13" лютого 2023 року
№ С-49

2. Термін подання студентом роботи «05» січня 2024 року

3. Вихідні дані до роботи Розробити веб застосунок який даватиме змогу дізнаватися про актуальні не офіційні і офіційні події у місті для сприяння активного проведення вільного часу мешканцями міста. Для реалізачій поставленої цілі використати мову програмування Python та Django фреймворк і з використанням Google API.

4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)
Стан проблемної області
Інформаційне забезпечення
Математичне забезпечення
Програмне забезпечення
Розроблення стартап-проекту
Висновки

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Матеріали для доповіді
Презентація

# КАЛЕНДАРНИЙ ПЛАН

| № з/п | Назва етапів дипломної роботи | Строк виконання етапів роботи | Примітка |
|---|---|---|---|
| 1 | Огляд літератури. Збір матеріалів | 20.02.23- 20.04.23 | **Виконано** |
| 2 | Постановка задачі і її формалізація | 20.04.23- 25.04.23 | **Виконано** |
| 3 | Дослідження стану предметної області | 25.04.23- 01.05.23 | **Виконано** |
| 4 | Підготовка Індивідуального Дослідного Завдання | 01.05.23-15.05.2023 | **Виконано** |
| 5 | Вибір інструментів для реалізації роботи | 15.05.23-01.06.2023 | **Виконано** |
| 6 | Розробка архітектури в рамках виконання Комплексного курсового проекту | 01.06.23-15.06.2023 | **Виконано** |
| 7 | Збір інформації щодо стратегій просування продукту на ринку для реалізації стартапу бета версії проекту | 15.06.23-01.07.2023 | **Виконано** |
| 8 | Реалізація алгоритмів і основних частин програми | 01.07.23-15.08.23 | **Виконано** |
| 9 | Ознайомлення з процесом розробки програмного забезпечення | 15.08.23-10.09.23 | **Виконано** |
| 10 | Відлагодження і тестування системи | 10.09.23-01.10.23 | **Виконано** |
| 11 | Впровадження та випуск альфа версії | 01.10.23-01.11.23 | **Виконано** |
| 12 | Оформлення пояснювально записки до дипломного проекту | 01.11.23-05.01.24 | **Виконано** |
| | | | |
| | | | |
| | | | |

**Студент** _____ **Устименко В.М.**
( підпис ) (прізвище та ініціали)

**Керівник роботи** _____ **Процах Н.П.**
( підпис ) (прізвище та ініціали)

# АНОТАЦІЯ

Дипломна робота містить 74 сторінки пояснювальної записки, 29 рисунків, 5 таблиць, 2 додатки, 28 джерел.

В даній роботі розроблено програмне та алгоритмічне забезпечення для створення платформи для популяризації активного відпочинку. Також було розроблено стартап проект, для просування даної платформи.

Програмне забезпечення розроблене з використанням мови програмування Python, фреймворку Django, бібліотеки Django-Auth, Google Maps API, та PostgreSQL базою даних.

**Ключові слова:** Python, PostgreSQL, Google Maps API, активний відпочинок, Social Media Marketing, Django.

# ABSTRACT

The thesis contains 74 pages of explanatory note, 29 figures, 5 tables, 2 appendices, 28 sources.

In this work, software and algorithmic support was developed for creating a platform for popularizing the active. A startup project was also developed to promote this platform.

The software will be developed using the Python programming language, the Django framework, the Django-Auth library, the Google Maps API, and the PostgreSQL database.

**Keywords:** Python, PostgreSQL, Google Maps API, active leisure, Social Media Marketing, Django.

## *Технічне завдання*

Метою магістерської роботи є розробка  програмного забезпечення для популяризації активного відпочинку. Завдання:

- Проаналізувати сучасний стану ринку активного відпочинку та подій.
- Вивчити існуючі платформи та їх можливості.
- Аналіз психологічних та соціокультурних аспектів активного відпочинку.
- Розробка концепції інформаційної архітектури платформи.
- Визначення методів інформаційної взаємодії з користувачами.
- Розробка системи збору та аналізу даних для персоналізації інформації.
- Розробка алгоритмів рекомендацій для користувачів на основі їхніх інтересів.
- Моделювання взаємодії користувачів з подіями та рейтингів.
- Вибір та обгрунтування використаних технологій (Python, Django, і т.д.).
- Розробка функціоналу веб-платформи для відображення подій та їхнього фільтрування.
- Реалізація адміністративного інтерфейсу для модераторів та організаторів.
- Розробка бізнес-плану для стартапу.
- Маркетингове дослідження та стратегія популяризації платформи.
- Розробка механізмів привертання інвестицій та співпраці з партнерами.

**Зміст**

## Вступ

Сучасне суспільство стикається з рядом викликів, пов'язаних із завантаженістю роботою, недостатнім фізичним рухом та стресовими ситуаціями, які стали невід'ємною частиною нашого повсякденного життя. Додатковою проблемою є розвиток галузі послуг, що сприяє зменшенню потреби у продукуванні та затраті енергії для реалізації потреб. Все це призводить до зниження рівня фізичної активності та загрози для нашого здоров'я та благополуччя.

***Актуальність дослідження.*** Україна також стикається зі зростанням кількості тимчасово переміщених осіб. За даними Міністерства соціальної політики, станом на початок 2023 року, в Україні офіційно зареєстровано 4,9 [1] мільйона внутрішньо переміщених осіб. Для цих людей утруднено пошук товариства однодумців через відсутність багатьох контактів та обмежені знання про місцевість.

Популяція людей пенсійного та передпенсійного віку також зростає. За даними "Великої Української Енциклопедії", станом на 2022 рік, в Україні налічувалося понад 12 мільйонів людей старше 55 років. Ця категорія населення часто має обмежений доступ до можливостей активного відпочинку та недостатню комп'ютерну грамотність.

Україна, на жаль, немає достатньої кількості цільових програм для популяризації саме активного відпочинку, що обмежує можливості для залучення широких верств населення. Крім того, низький рівень комп'ютерної освіченості, особливо серед літнього населення, ускладнює використання інформаційних технологій та інтернет-платформ для пошуку подій активного відпочинку.

Базуючись на потребах людей, дана робота веб платформі для популяризації активного відпочинку. Ця система, має позитивно вплинути на економіку і розвиток туристичної галузі держави, а також покращити і полегшити життя людей у сфері туризму.

***Об'єктом дослідження*** в даній роботі є платформи для пошуку подій активного відпочинку у межах визначеної локації (області, міста, району міста і т.д.)

***Предметом дослідження*** є доцільність створення такої платформи та її вплив на популяризацію активного відпочинку. Використання серед таргетованої аудиторії.

***Метою роботи*** є проектування функціональної та зручної у використанні платформи, яка спонукає людей до зайняття активним відпочинком, а також допомагає їм відкривати нові можливості для відпочинку та пригод.

***Наукова новизна*** полягає у розробці та впровадженні платформи, спеціально створеної для популяризації активного відпочинку. Ця платформа має на меті надати користувачам комплексну інформацію про різноманітні активності, сприяти їх пошуку та фільтрації, а також дозволити користувачам оцінювати та залишати відгуки про активності.

***Практичне значення одержаних результатів.***Здійснюючи ці завдання, проект має на меті сприяти популяризації активного відпочинку та надавати користувачам цінний ресурс для відкриття та участі у різноманітних рекреаційних активностях.

## Розділ 1.Стан предметної області

### 1.1Аналіз сфери активного відпочинку

Сучасні люди все більше усвідомлюють важливість здорового способу життя, фізичної активності та відпочинку на природі. Популяризація активного відпочинку сприяє задоволенню цих потреб і надає людям можливість знаходити та використовувати різноманітні активності для покращення фізичного та психологічного здоров'я.

Туризм і активний відпочинок стали важливим сегментом економіки багатьох країн. Галузь Туристичної індустрії зростає, разом з тим в дрібніших масштабах - області/району/міста - це дієвий інструмент для привернення уваги людей та меценатів що сприятиме економічному зростанню.

Віртуальні спільноти та соціальні медіа стали важливими інструментами спілкування та обміну інформацією. Платформи для популяризації активного відпочинку надають зручний спосіб поширювати інформацію про різноманітні активності, об'єднувати спільноти за інтересами та сприяти обміну досвідом.

Україна також стикається зі зростанням кількості тимчасово переміщених осіб. За даними Міністерства соціальної політики, станом на початок 2023 року, в Україні офіційно зареєстровано 4,9 мільйона внутрішньо переміщених осіб. Для цих людей утруднено пошук товариства однодумців через відсутність багатьох контактів та обмежені знання про місцевість.

Популяція людей пенсійного та передпенсійного віку також зростає. За даними "Великої Української Енциклопедії", станом на 2022 рік, в Україні налічувалося понад 12 мільйонів людей старше 55 років. Ця категорія населення часто має обмежений доступ до можливостей активного відпочинку та недостатню комп'ютерну грамотність.

Україна, на жаль, немає достатньої кількості цільових програм для популяризації саме активного відпочинку, що обмежує можливості для залучення широких верств населення. Крім того, низький рівень комп'ютерної освіченості,

особливо серед літнього населення, ускладнює використання інформаційних технологій та інтернет-платформ для пошуку подій активного відпочинку.

Задовольнити всі потреби туризму в одному застосунку дуже важко, саме тому зазвичай використовується підхід «Розділяй і володарю», що дозволяє обмежитися певною під-сферою: готелі, екскурсії, квитки на літак чи автобус, тощо. Саме такий підхід допомагає зосередити увагу на конкретній проблемі і створити систему, яка б задовольняла потреби більшості населення.

Сферу активного відпочинку, можна вважати під-сферою туризму. Туристична сфера складається з багатьох складових під-сфер, в яких працює неймовірно-велика кількість людей. Для наочності, проаналізуємо деякі компоненти туристичної сфери і виділимо ту частину, для якої буде створена інформаційна система, для подальшого розвитку і популяризації активного відпочинку [2].

Туризм складається з деяких наступних складових:

- Транспортні сполучення між місцями:
    - Літаки.
    - Потяги.
    - Автобуси.
    - Кораблі.
    - Автостоп.
    - Тощо.
- Місця проживання:
    - Готелі.
    - Хостели.
    - Кемпінги.
- Екскурсії різних типів.
- Заклади харчування.
- Активний відпочинок:
    - Кемпінг.

- Хайкінг.
- Сапи.
- Спортивна Рибалка.
- Віндсерфінг.
- Серфінг.
- Стрибки з парашутом.
- Кемпінг.
- Скелелазіння.
- Альпінізм.
- Рафтинг.
- Прогулянка на повітряній кулі.
- Прогулянки на літаком.
- Прогулянки по бездоріжжю на квадрациклі.
- Байдарки.
- Велосипедний тур.
- Кінні прогулянки.
- Лижі, сноуборд.
- Кайтбординг.
- Кайтсерфинг.
- Вейксерфінг.
- Вейкбординг.
- Ліплення з глини.
- Відвідування цікавих і визначних місць.
- Тощо.

Перелічивши різні варіанти активного відпочинку, можна побачити, що сфера доволі об’ємна і включає в себе багато способів проводження часу.

Також важливим фактором є те, що з кожним роком з’являються нові й нові варіанти, як можна активно відпочивати.

Проаналізувавши сферу активного відпочинку, можна виділити необхідні складові для створення мінімально-функціонуючої інформаційної технології (рис. 1.1).


Рисунок 1.1 – Складові інформаційної технології активного відпочинку

Кожен з компонентів відіграє важливу роль для успішногофункціонування системи, також, варто зауважити, що кількість компонентів може збільшуватись, зважаючи на потреби користувачів.

Організатори відпочинку мають великий вплив на систему, тому, для них має бути створений зручний і зрозумілий інтерфейс для взаємодії. Споживачі, також, мають мати зручний і доступний інтерфейс для того щоб застосунок був дійсно корисним і конкурентно-спроможним в порівнянні з іншими рішеннями.

Важливим елементом є взаємодія між клієнтом і користувачем, яка може бути реалізована різними способами і технологіями, починаючи від простого надання клієнтові або організатору контактних даних, закінчуючи власною системою листування.

Крім різного типу клієнтських інтерфейсів існують, також, такі важливі елементи, як менеджмент і обробка інформації, захист особистих даних,адаптивність і масштабованість, а також білінг. Всі процеси мають відбуватися швидко. Зазвичай користувачі не люблять чекати поки все завантажиться. Крім того, якщо застосунок не матиме захисту то їм буде просто

неможливо користуватися бо ніхто не хоче ділитися особистими даними з усіма людьми, більш того, захист особистої інформації регулюється законом.

Адаптивність і масштабованість дозволяє застосунку розроблювати нові функції і не впливати на вже існуючі функціональності. Таки чином, можна, дуже легко створювати експериментальні модулі, виявляти потреби користувачів і впроваджувати нові ідеї.

Білінг є також важливим компонентом в життєвому циклі застосунку, але даний модуль не є найважливішим і при розробці перших версій продукту може повністю буди відсутнім.

## 1.2Аналіз конкурентних систем

Незважаючи на розвиток інформаційних систем, ще існує багато сфер людської життєдіяльності, які взагалі не мають інформаційних систем оптимізації або потребують покращення і розширення. Таким чином, проаналізувавши конкурентів, можна виділити найпопулярніші, схожі за деякими принципами системи у сфері туризму, які мають схожі функціональності з технологією яку представляє дана робота.

Існує кілька популярних платформ для популяризації активного відпочинку, серед яких можна виділити:

– TripAdvisor: Ця платформа дозволяє користувачам шукати, порівнювати та бронювати різноманітні рекреаційні активності, такі як тури, екскурсії, спортивні заходи та інші.

– AirbnbExperiences: Airbnb, відомий своїми послугами бронювання помешкань, також пропонує розділ Experiences, де користувачі можуть знайти та бронювати активності, проводити майстер-класи, екскурсії та інші події, які організовують місцеві жителі.

– Meetup: Ця платформа спрямована на об'єднання людей зі схожими інтересами. Тут можна знайти різноманітні групи та події, пов'язані з активним відпочинком, спортом, пригодами та іншими активностями.

– Eventbrite: Eventbrite є платформою для організації та реєстрації на різноманітні події. Вона включає в себе події активного відпочинку, такі як марафони, фестивалі, спортивні змагання та інші.

– AllTrails: Ця платформа спеціалізується на пошуку та навігації по маршрутах для пішохідних прогулянок, велосипедних поїздок та інших активностей на відкритому повітрі. Користувачі можуть знайти маршрути, переглядати відгуки, фотографії та ділитися своїми власними маршрутами.

– Strava: Ця платформа спеціалізується на активностях, пов'язаних зі спортом та фітнесом. Вона дозволяє користувачам записувати свої тренування, ділитися ними з іншими, створювати виклики та змагання, а також приєднуватися до груп та спілкуватися з іншими спортсменами.

Ці платформи надають користувачам можливість знайти різноманітні активності, оцінювати їх, спілкуватися з іншими учасниками та забезпечити інформування щодо певних спортивних подій [3].

Переваги існуючих платформ:

– Досвід на ринку.

– Існуюча аудиторія.

Недоліки існуючих платформ

– Обмежена спеціалізація (туризм, масові заходи тощо).

– Цільова аудиторія (молодь, люди середнього віку).

– Складний інтерфейс.

– Відсутність відображення івентів на мапі.

– Відсутність співпраці з локальними державними інституціями.

Також існує велика кількість підприємців, які мають власний сайт і на якому вони пропонують свої послуги, але це лише декілька десятків пропозицій без реальних відгуків і оцінок, а також без можливості порівняти пропозиції конкурентів.

Незважаючи на достатню кількість різних систем для пошуку відпочинку, жодна з них не зосереджена саме на активного відпочинку і немає достатньої

популярності і авторитетності в Україні. Таким чином, можна зробити висновок до проведеного аналізу конкурентів, що систему, саме для пошуку активного відпочинку, яка б інтенсивно розвивала цю сферу, поки не створили або не розвинули до достатньо популярного рівня.

## 1.3Проблематика дослідження

У сучасному світі, можна знайти інформацію майже про все, починаючи від вирощування огірків, закінчуючи будівництвом ракет, але проблема полягає в тому, що на знаходження потрібної інформації може знадобитися дуже багато часу і сил. Враховуючи той факт, що мозок людини влаштований таким чином, щоб заощаджувати енергію, зазвичай, багатьом не вистачає терпіння знайти необхідну інформацію, і саме це стає причиною прийняття не найкращого рішення або вибору продукції чи послуг. Таким чином, однією з проблем, яку має вирішувати розроблена система є надання користувачу широкого вибору, який підкріплюється зворотнім зв'язком інший користувачів, що дозволяє іншим туристам краще зрозуміти рівень сервісу, який надається організатором і прийняти вигідне рішення і гарно провести час.

Зважаючи на той фактор, що існує достатня кількість підприємців, які працюють у сфері туризму, майже кожен з них має власний сайт або сторінку в соцмережах, де вони продають свої послуги активного відпочинку, такий підхід ведення бізнесу у даній сфері є складним, як для існуючих так і нових організаторів івентів бо потребує значного інвестування часу. Підприємці мають слідкувати за багатьма чинниками, такими як: актуальність контенту, привабливість інформації про відпочинок і вигляд сайту в цілому, а також інколи є потреба у кваліфікованій допомозі з боку фахівців зі сфери інформаційних технологій, що спричиняє додаткові витрати часу і коштів. Враховуючи підприємців, які тільки починають ріст у сфері туризму, їм дуже важко конкурувати з існуючими підприємцями, хоч їх сервіс може бути набагато кращим. Таким чином, другу проблему, яку має вирішувати система є зручність

ведення бізнесу у даній сфері, за рахунок зручного і розширюваного інтерфейсу, а також «здорової» конкуренції між різними за досвідом і сервісом підприємцями. Організатори можуть надавати детальну інформацію, а також, розширювати або редагувати її, що відрізняє дану систему від статичних форм представлення інформації іншими локальними сайтами підприємців.

Інколи, буває так, що людина сама не знає чого хоче, а коли невідомо, що шукати то дуже складно це знайти. Саме так буває, коли люди хочуть спробувати, щось нове, але не знають де і як це знайти. Таким чином, розроблювана інформаційна технологія, направлена на ознайомлення і пропонування клієнтам нових і різноманітних видів відпочинку, що має підвищувати їх інтерес до активного проводження часу і задовольняти потреби у оригінальному відпочинку.

З кожним роком кількість інтернет-ресурсів неперервно збільшується, і станом на початок 2023 року ця кількість перевищує два мільярди. Разом з цим, зростає і кількість користувачів інтернету, що за даними Google складає понад п'ять мільярдів, з яких близько 80% активно використовують соціальні мережі та месенджери [4]. Враховуючи це, сучасна пошукова оптимізація може бути розділена на два основних напрямки: роботу над внутрішньою структурою ресурсу, зокрема, наповнення його якісним контентом та покращення його зручності для користувача [5], і роботу з зовнішніми факторами, яка включає створення цільових посилальних зв'язків за допомогою авторитетних ресурсів, використання комерційних методів просування, зокрема контекстної реклами, а також використання соціальних медіа [6].

Соціальні медіа охоплюють різноманітні онлайн-технології, які надають засоби комунікації та соціальної взаємодії з іншими користувачами, такі як спільні проекти, блоги, соціальні мережі та контент-спільноти [7]. За статистичними даними [8], близько 90% користувачів інтернету в Україні щодня використовують соціальні медіа для перегляду новин та коментарів, отримання інформації про товари та послуги, аналізу оглядів. Інтернет-спільнота може значно впливати на репутацію компанії та просування товару або послуги. Тому

популяризація комерційних інтернет-ресурсів неможлива без їхнього просування в соціальних медіа, які надають можливості впливати на рейтинг і сприяють визнаності бренду [9].

Аналіз стратегій просування (реклами) інтернет-ресурсів є важливим для розуміння того, як компанії та організації використовують інтернет для привертання уваги цільової аудиторії та просування своїх товарів, послуг або ідей. Застосування ефективних стратегій просування допомагає залучати більше користувачів та збільшувати вплив рекламодавця в онлайн-середовищі.

Одна з основних стратегій просування в інтернеті - контекстна реклама. Вона ґрунтується на відображенні рекламних оголошень, що відповідають контексту сторінки або запиту користувача. Ця стратегія базується на використанні ключових слів, тематики або пов'язаних змістом ресурсу, що дозволяє показувати рекламу, яка максимально відповідає інтересам користувачів.

Інша популярна стратегія - ретаргетинг (повторне відстеження). Вона використовується для показу реклами тим користувачам, які вже взаємодіяли з веб-сайтом чи раніше проявляли інтерес до певного продукту або послуги. Рекламні оголошення можуть з'являтися на інших сайтах, які відвідує ця аудиторія, що нагадує їм про певну пропозицію і спонукає до подальших дій.

Додатково, стратегії просування в інтернеті включають банерну рекламу, спонсоровані публікації в соціальних мережах, електронну розсилку, впливовий маркетинг, а також розміщення контенту на платформах відеострімінгу, блогах та форумах.

Аналіз стратегій просування інтернет-ресурсів дозволяє виявити ефективні підходи та інструменти, що сприяють досягненню маркетингових цілей та забезпечують максимальний вплив на цільову аудиторію в онлайн-середовищі.

Проведений аналіз показав, що в сучасному світі соціальні медіа стали невід'ємною частиною життя користувачів і призвели до появи нових підходів до просування інтернет-ресурсів [10, 11]. Якщо раніше при класичному SEO просуванні основною стратегією була оптимізація контенту для пошукових

систем, то з поширенням соціальних медіа акцент переноситься на привабливість для користувачів і включає такі елементи як створення та управління групами за інтересами, використання вподобань користувачів, мультимедійний зміст та оптимізацію для мобільних платформ [6].

Однак, новаторський підхід і постійні зміни в алгоритмах пошукових систем призвели до того, що на сьогоднішній день наукові дослідження переважно зосереджуються на аналізі семантичного наповнення інтернет-ресурсів [12–14], контекстній рекламі [15], оптимізації внутрішніх і зовнішніх факторів просування [16, 17] і алгоритмах розширених сніпетів [18]. Враховуючи це, актуальним завданням є аналіз існуючих моделей просування та розробка алгоритму для популяризації комерційних інтернет-ресурсів за допомогою соціальних медіа.

## 1.4Постановка завдання дослідження

Завданням дослідження є: розроблення веб платформи для популяризації активного відпочинку, серверна частина якої, буде базуватиметеся на мікросервісній архітектурі, а також клієнтський інтерфейс для взаємодії користувачів через браузер. Перед етапом розробки провести етап планування, який включатиме схематичну побудову архітектури, внутрішніх і зовнішніх процесів. Розробити моделі бази даних, створивши діаграми сутностей і зв’язків. Провести аналіз сучасних технологій і рішень для подальшого використання. При розробці системи, базуватися на вирішенні актуальних проблем у даній сфері.

Для подальшого дослідження необхідно розробити такі функціональності для користувачів:

- Реєстрація і авторизація двох типів користувачів, а саме туристів і гідів.
- Створенняіменеджментпропозиційактивноговідпочинку організаторами.
- Пошук по критеріях пропозицій.
- Бронювання і додавання турів до улюблених.
- Написання відгуків.

Спроєктувати і розробити гнучку мікросервісну архітектуру, а також інтерфейс для взаємодії клієнтських застосунків, за допомогою якого відбуватиметься комунікація з серверною частиною. Для цього необхідно розробити такі мікросервіси :

– Конфігурацій – для централізованого доступу і налаштування інших інтерфейсів.

– Навантаженняімоніторингуданих–длямоніторингуі розподілення навантаження.

– Шлюзу – для надавання точки вході клієнтським застосункам через один інтерфейс.

– Організаторівтурів–дляобробленняінформаціїпро організаторів турів.

– Подорожей – для оброблення інформації про подорожі.

– Туристів – для оброблення інформації про туристів.

– Коментарів – для оброблення коментарів.

– Аутентифікації – для реєстрації.

## Розділ 2. Інформаційне забезпечення

### 2.1 Вибір мови програмування

Для реалізації веб платформи для популяризації активного відпочинкубуло вирішено обрати мультипарадигмальну мову програмування, з підтримкою ООП, великим об’ємом стандартної бібліотеки та наявністю вибору серед множини веб-фреймворків. Також до мови програмування була висунута вимога інтерпретованості з метою підвищення швидкості розробки та налагодження коду розроблюваної системи.

#### 2.1.1Мова PHP

PHP – мова програмування загального призначення, яка створена переважно для веб-розробки. Код PHP може виконуватися як за допомогою інтерфейсу командного рядка, вбудованого в HTML-код, так і в поєднанні з різними системами веб-шаблонів, системами керування веб-контентом і веб-фреймворками. Це найпопулярніше рішення для веб-розробки з точки зору кількості веб-сайтів (серед яких Facebook та Вікіпедія), де використовується ця мова, та систем керування контентом, побудованих на її основі (Wordpress, Drupal та Bolt серед найбільш відомих) [19].

Мова PHP знайшла застосування і за межами контексту веб-розробки, з її допомогою розробляються:

–GUI-додатки;

–кросплатформні додатки;

–програмні системи для управління дронами [20].

Мова є інтерпретованою; код обробляється інтерпретатором PHP, що зазвичай реалізований модулем у веб-сервері. Інтерпретованість дозволяє достатньо швидко обробляти написані сценарії, в порівнянні з іншими мовами (наприклад, Perl).

Веб-сервер комбінує результати інтерпретованого та виконаного PHP- коду, який може містити будь-які типи даних, з-поміж них зображення, зі згенерованою

веб-сторінкою, що передається на сторону клієнта. На відмінувід скриптових мов програмування, користувач не має доступу до PHP-коду, тому що браузер отримує згенерований HTML-код в готовому вигляді.

Стандартний інтерпретатор мови PHP вільно розповсюджується, може бути розгорнутий на більшості веб-серверів практично на будь-якій операційній системі та платформі безкоштовно.

Синтаксис PHP подібний до синтаксису мови C, містить запозичені конструкції з Perl (наприклад, асоціативні масиви). В процесі історичного розвитку мова розширювалась та доповнювалась новими і запозиченими з інших мов можливостями без встановлення послідовних правил та конвенцій. З одного боку, це спричинило неузгодженість синтаксису PHP, але з іншого боку, мова отримала невисокий поріг входження для програмістів з різних галузей, обумовлений наявністю традиційних та знайомих конструкцій.

PHP є мультипарадигмальною мовою програмування, з підтримкою функціональної, процедурної та об'єктно-орієнтованої парадигм [21]. Починаючи лише з п’ятої версії мова підтримує ООП: три основні механізми (інкапсуляцію, наслідування та поліморфізм), інтерфейси, абстрактні класи та методи. Множинне наслідування не підтримується.

Інтерпретатор PHP забезпечує обробку скриптів зі збереженням кросплатформності розроблюваних додатків. Швидкодія додатків може бути збільшена з допомогою спеціального програмного забезпечення - акселераторів.

Програміст не має піклуватися про розподіл і звільнення пам'яті, тому що ядро PHP надає засоби для автоматичного керування пам'яттю; вся виділена пам'ять повертається в систему після завершення роботи скрипта. Стандартна бібліотека мови PHP являє собою колекцію класів та інтерфейсів для вирішення типових проблем PHP. Бібліотека доступна розробнику за замовчуванням та загалом містить класи-ітератори дляітерацій по каталогу, масиву, дереву XML.

Найбільш поширеними PHP-фреймворками для веб-розробки є Laravel, CodeIgniter, Symfony, Zend.

### 2.1.2.Мова JavaScript

JavaScript – високорівнева інтерпретована мультипарадигмальна мова програмування з підтримкою об’єктно-орієнтованого, функціонального та імперативного стилів; реалізація стандарту ECMAScript. Широке застосування мова JavaScript знайшла в галузі веб-розробки, створенні сценаріїв веб-сторінок. Мова надає можливості для реалізації як клієнтської, так і серверної частин веб-додатків, взаємодії з користувачем на стороні клієнта, керування браузером, асинхронного обміну даними з сервером, динамічної зміни структури та зовнішнього вигляду сторінок. JavaScript має С-подібний синтаксис, динамічну типізацію, автоматичне управління пам’яттю, прототипно-орієнтований підхід до реалізації ООП [22, 23].

Типові випадки використання мови JavaScript:

–створення сценаріїв для інтерактивних веб-сторінок;

–розробка односторінкових веб-додатків (клієнтські фреймворки React, AngularJS, Vue.js);

–розробка серверної частини веб-додатків (фреймворк Node.js);

–розробка кросплатформних додатків (фреймворки Electron, NW.js);

–розробка нативних мобільних додатків (ReactNative, Cordova);

–написання сценаріїв для прикладного ПЗ.

Мова не є об’єктно-орієнтованою в класичному розумінні через використання прототипного програмування, в якому поняття класу відсутнє, а наслідування відбувається шляхом клонування прототипу – існуючого примірника об’єкту.

JavaScript надає властивості, запозичені з функціональних мов програмування, зокрема функції як об’єкти першого класу, об’єкти як списки, каррінг, анонімні функції, замикання, що надає мові додаткової гнучкості, втім не є значною перевагою в контексті розроблюваної системи.

Мова використовується в AJAX, поширеному підході до побудови інтерактивних користувацьких інтерфейсів веб-додатків, що полягає в

«фоновому» асинхронному обміні даними браузера з веб-сервером з метою пришвидшення їх взаємодії [24].

Варто відзначати відсутність у складі мови наступних складових:

–стандартна бібліотека (відсутній інтерфейс для роботи з файловою системою, управління потоками введення-виведення);

–стандартні інтерфейси до веб-серверів та баз даних;

–системауправлінняпакетамидлявідслідковуваннята автоматичного встановлення залежностей.

Для розробки серверної частини додатків мова не може застосована безпосередньо, потрібне використання спеціальних JavaScript-рушіїв [25]:

–Rhino – перетворює JavaScript-скрипти в Java класи, може працювати в інтерпретованому режимі, не має вбудованої підтримки для об’єктів браузера;

–SpiderMonkey – рушій вбудовується в інші застосунки, які надають робоче оточення для JavaScript; містить компілятор, інтерпретатор, декомпілятор, прибиральник сміття і стандартні класи;

–V8 – найбільш швидкий та потужний рушій, в якому JavaScript-код безпосередньо перетворюється в асемблер цільового процесора, що дозволяє обійти за швидкістю інші засоби.

На основі V8 побудована популярна серверна платформа Node.JS. Саме впровадження Node.JS перетворило мову JavaScript на мову загального використання з можливістю виконувати JavaScript-скрипти на сервері та відправляти користувачам результати їх виконання.

Окрім рушія, Node.JS постачає вбудований сервер та базовий набір бібліотек, а також надає можливість асинхронної роботи з файлами та мережевими пристроями.

Наразі мова JavaScript є однією з найпопулярніших в інтернеті, підтримує відповідність сучасним стандартам веб-програмування та надає засоби для серверної розробки, проте не всі її інструменти є достатньо зрілими та протестованими і не завжди мають вичерпну документацію [26]. Також

асинхронність самої мови в поєднанні з однопоточністю платформи для серверної розробки сприяє зниженню обчислювальної продуктивності.

### 2.1.3.Мова Ruby

Ruby – інтерпретована мова програмування з повною підтримкою ООП та динамічною типізацією; якісно вирізняється високою ефективністю розробки програм. Синтаксис Ruby подібний до синтаксису мови Perl, а підхід до реалізації ООП нагадує відповідний підхід мови програмування Smalltalk; деякі властивості були запозичені з інших мов – Java, Python, Lisp [27].

Інтерпретатор мови розповсюджується безкоштовно та доступний для більшості платформ і ОС. Незалежно від ОС мова надає власну реалізацію багатопоточності та автоматичну збірку сміття.

Ruby має широкий спектр застосування – з допомогою мови:

–розробляються програмні продукти різного призначення;

–відбувається автоматизація та налаштування додатків;

–можливе написання адміністративних утиліт.

Динамічні засоби мови з одного боку сприяють ефективності програмування, однак знижують продуктивність.

Ruby не містить примітивні типи, надаючи перевагу цілком об’єктно-орієнтованій організації: всі дані у програмах є об’єктами, всі функції – методами. Мова не має підтримки множинного наслідування.

Мультипарадигмальність втілена в наявності рис процедурного стилю (визначення функцій та змінних поза межами класу), об’єктно- орієнтованого та функціонального стилю (анонімні функції, замикання); також розробнику доступні рефлексія, метапрограмування, інформація про типи змінних на стадії виконання.

Мова постачається зі стандартною бібліотекою великого об’єму, що містить засоби для роботи з мережевими протоколами на клієнтській та серверній стороні, засоби для обробки та представлення різноманітних форматів даних.

Додаткові можливості надають бібліотеки Ruby для модульного тестування, роботи з архівами, датами, матрицями та засоби для системного адміністрування, розподілених обчислень. Бібліотека Rubyreports призначена для легкої реалізації звітів і створення діаграм на основі даних з БД та текстових файлів CSV.

Мова Ruby є порівняно молодою, незадовго після її створення з'явився веб-фреймворк RubyonRails, що користується популярністю і сьогодні через легкість побудови типових веб-додатків. Також існують веб- фреймворки Nitro, Webby, Sinatra.

### 2.1.4.Мова Python

Python – інтерпретована об'єктно-орієнтована мова програмування високого рівня зі строгою динамічною типізацією; підтримує кілька парадигм програмування: об'єктно-орієнтовану, процедурну, функціональну та аспектно-орієнтовану [28].

Мова використовується для вирішення різноманітних проблем в галузі розробки ПЗ:

–створення десктопних додатків з GUI (ігри, наукові програми);

–створення веб-додатків та веб-фреймворків;

–розробка програм для підприємств та бізнесу;

–розробка ОС;

–прототипування.

Інтерпретатор Python та стандартні бібліотеки мови доступні як у скомпільованій, так і в вихідній формі для всіх основних платформ та ОС. Стандартнабібліотекамістить великийоб'ємкорисних функцій.Із мінімалізмом синтаксису мова орієнтована на підвищення продуктивності програміста та читабельності коду.

Python є мовою програмування загального призначення, яка продовжує активно розвиватися; до мови додаються та змінюються існуючі властивості, проте основні риси залишаються сталими. Серед таких зокрема динамічна

типізація, автоматичне управління пам’яттю, повна інтроспекція, механізми обробки виключень, підтримка багатопоточності, розвинені високорівневі структури даних та динамічна перевірка типів.

Портованість мови дозволяє працювати на будь-якій з відомих платформ; втім, на відміну від інших портованих систем, Python надає підтримку характерних для платформи технологій. Існує спеціальна версія мови для віртуальної машини Java – Jython, рішення для інтеграції з платформою Microsoft .NET – IronPython, Python.Net.

Всі дані в програмах, написаних на Python, є об’єктами, в тому числі функції, методи, класи та модулі.

Мова містить як примітивні типи, так і вбудовані колекції - списки, кортежі, словники, множини.

Система класів Python підтримує одиночне та множинне наслідування, метапрограмування. Можливе наслідування від більшості вбудованих типів та типів розширень. Метапрограмування в мові дозволяє швидко та елегантно реалізовувати складні шаблони проектування. Python- фреймворки, такі як Django, DRF, SQLAlchemy, використовуютьметапрограмування для забезпечення легкої розширюваності та повторного використання коду.

Характерними для мови є чіткість та послідовність синтаксису, модульність та розширюваність. Модулі Python (розміщені в каталогах файлової системи або ZIP-архівах) об’єднуються в пакети, формуючи розроблені додатки чи бібліотеки. Для отримання доступу до простору імен модуля використовується оператор підключення import.

Репозиторій ПЗ для мови Python (PyPI) містить велику кількість сторонніх модулів:

–для веб-розробки (фреймворки, мікрофреймворки, системи управління контентом, засоби для синтактичного аналізу веб- сторінок, для підтримки різних Інтернет-протоколів та роботи з повідомленнями електронної пошти);

–для організації взаємодії з БД (пакети для доступу до різних СУБД, зокрема PostgreSQL, Microsoft SQL Server, MySQL та SQLite);

–для проведення спеціалізованих математичних та наукових розрахунків (пакети для обробки та аналізу даних та математичного моделювання, роботи з багатомірними масивами та алгоритмами оптимізації);

–для створення кросплатформних програм з GUI (пакети tkinter, wxPython; Pygame для розробки ігор та роботи з мультимедіа, а також засоби для роботи з растровою графікою).

Засоби каталогу PyPI в поєднанні зі стандартною бібліотекою Python надають розробнику будь-якого рівня необмежені можливості для розв’язання широкого класу інженерних задач.

Python може використовуватись в діалоговому режимі з дебагером та вичерпною системою довідки, що є перевагою для експериментування та розв’язання простих задач. Такий режим корисний як новачкам, так і досвідченим програмістам для тестування будь-яких фрагментів коду перед його використанням в основній програмі.

Мова програмування має вдалу і потужну реалізацію ООП, відмінну від решти об'єктно-орієнтованих мов, з наступними особливостями, крім наведених раніше:

–поліморфізм – всі функції віртуальні;

–інкапсуляція, приховані члени доступні для використання та помічені як приховані особливими іменами;

–спеціальніметодидлякеруванняжиттєвимцикломоб'єктів: конструктори, деструктори, розподільники пам'яті;

–перевантаження всіх операторів, крімів,'.','=' і символьних логічних;

–управління доступом до полів (частковий доступ, емуляція полів і методів);

–наявність методів для виконання поширених операцій (істинносне значення, глибоке копіювання, серіалізація, ітерація по об'єкту);

–повна інтроспекція;

–класові та статичні методи, класові поля;

–класи, вкладені у функції та інші класи.

Існує встановлений стандарт взаємодії між Python-програмами, виконуваними на стороні сервера, та самим веб-сервером (наприклад, HTTP-сервер Apache) – WSGI. Він надає простий на універсальний спосіб взаємодії між більшістю веб-серверів та веб-додатків чи фреймворків [29, 30]. WSGI визначає middleware-компоненту для надання інтерфейсів як серверу, так і веб-додатку, а також:

–обробки сесій;

–автентифікації/авторизації;

–управління URL (маршрутизації запитів);

–розподілу навантаження;

–пост-обробки вихідних даних.

Популярними сумісними з WSGI Python веб-фреймворками є Django, Flask, Dash, CherryPy, TurboGears, Pylons.

У табл. 3 узагальнено відомості про порівняні мови програмування для розробки серверної частини системи спільного управління фінансами громади.

На основі порівняльної таблиці в якості мови програмування було обрано Python – портовану мову з широким вибором доступних модулів для веб-розробки,детальноюдокументацією,вичерпнимтазручним синтаксисом, вдалим дизайном ООП та інтерфейсами до найпоширеніших БД. Не меншою мірою вибір мови обґрунтований наявністю інтерактивних засобів тестування та налагодження коду.

Таблиця 2.1

Порівняльна характеристика мов програмування

| Ознака | PHP | JavaScript | Ruby | Python |
|---|---|---|---|---|
| Мультипара-дигмальність, підтримка ООП | Підтримка функціональ ного, процедурно- го стилів, слабко розвинене ООП | Підтримка функціональ ного, процедурно- го стилів, прототипне програму- ваня на противагу ООП | Підтримка функціональ ного, процедурно-го,об'єктно-орієнтовано- го стилів | Підтримка функціональ ного, процедурно-го,об'єктно-орієнтовано го стилів |
| Стандартна бібліотека | Доступна для вирішення типових проблем PHP | Відсутня | Доступна, багатофунк-ціональні засоби | Приваблива сторона мови – велика кількість модулів |
| Наявність фреймворків для веб- розробки | + | + | + | + |
| Інтерпретова-ність | + | + | + | + |

## 2.2 Аналіз особливостей просування інтернет платформ

З кожним роком кількість інтернет-ресурсів неперервно збільшується, і станом на початок 2020 року ця кількість перевищує два мільярди. Разом з цим, зростає і кількість користувачів інтернету, що за даними Google складає понад п'ять мільярдів, з яких близько 80% активно використовують соціальні мережі та месенджери [4]. Враховуючи це, сучасна пошукова оптимізація може бути розділена на два основних напрямки: роботу над внутрішньою структурою

ресурсу, зокрема, наповнення його якісним контентом та покращення його користувацької дружності [5], і роботу зовнішніми факторами, яка включає створення цільових посилальних зв'язків за допомогою авторитетних ресурсів, використання комерційних методів просування, зокрема контекстної реклами, а також використання соціальних медіа [6].

Соціальні медіа охоплюють різноманітні онлайн-технології, які надають засоби комунікації та соціальної взаємодії з іншими користувачами, такі як спільні проекти, блоги, соціальні мережі та контент-спільноти [7]. За статистичними даними [8], близько 90% користувачів інтернету в Україні щодня використовують соціальні медіа для перегляду новин та коментарів, отримання інформації про товари та послуги, аналізу оглядів. Інтернет-спільнота може значно впливати на репутацію компанії та просування товару або послуги. Тому популяризація комерційних інтернет-ресурсів неможлива без їхнього просування в соціальних медіа, які надають можливості впливати на рейтинг і сприяють визнаності бренду [9].

### 2.2.1 Потреба у просуванні

Аналіз стратегій просування (реклами) інтернет-ресурсів є важливим для розуміння того, як компанії та організації використовують інтернет для привертання уваги цільової аудиторії та просування своїх товарів, послуг або ідей. Застосування ефективних стратегій просування допомагає залучати більше користувачів та збільшувати вплив рекламодавця в онлайн-середовищі.

Одна з основних стратегій просування в інтернеті - контекстна реклама. Вона ґрунтується на відображенні рекламних оголошень, що відповідають контексту сторінки або запиту користувача. Ця стратегія базується на використанні ключових слів, тематики або пов'язаних змістом ресурсу, що дозволяє показувати рекламу, яка максимально відповідає інтересам користувачів.

Інша популярна стратегія - ретаргетинг (повторне відстеження). Вона використовується для показу реклами тим користувачам, які вже взаємодіяли з

веб-сайтом чи раніше проявляли інтерес до певного продукту або послуги. Рекламні оголошення можуть з'являтися на інших сайтах, які відвідує ця аудиторія, що нагадує їм про певну пропозицію і спонукає до подальших дій.

Додатково, стратегії просування в інтернеті включають банерну рекламу, спонсоровані публікації в соціальних мережах, електронну розсилку, впливовий маркетинг, а також розміщення контенту на платформах відеострімінгу, блогах та форумах.

Аналіз стратегій просування інтернет-ресурсів дозволяє виявити ефективні підходи та інструменти, що сприяють досягненню маркетингових цілей та забезпечують максимальний вплив на цільову аудиторію в онлайн-середовищі.

Проведений аналіз показав, що в сучасному світі соціальні медіа стали невід'ємною частиною життя користувачів і призвели до появи нових підходів до просування інтернет-ресурсів [10, 11]. Якщо раніше при класичному SEO просуванні основною стратегією була оптимізація контенту для пошукових систем, то з поширенням соціальних медіа акцент переноситься на привабливість для користувачів і включає такі елементи як створення та управління групами за інтересами, використання вподобань користувачів, мультимедійний зміст та оптимізацію для мобільних платформ [6].

Однак, новаторський підхід і постійні зміни в алгоритмах пошукових систем призвели до того, що на сьогоднішній день наукові дослідження переважно зосереджуються на аналізі семантичного наповнення інтернет-ресурсів [12–14], контекстній рекламі [15], оптимізації внутрішніх і зовнішніх факторів просування [16, 17] і алгоритмах розширених сніпетів [18]. Враховуючи це, актуальним завданням є аналіз існуючих моделей просування та розробка алгоритму для популяризації комерційних інтернет-ресурсів за допомогою соціальних медіа.

### 2.2.2 Моделі популяризації інтернет платформ

Існують дві поширені моделі просування комерційних ресурсів: SMM (Соціальний маркетинг у соціальних медіа) та SMO (Оптимізація в соціальних медіа). SMM передбачає популяризацію інтернет-ресурсу або послуг компанії за допомогою соціальних медіа. Застосування методів, що використовуються в рамках цієї моделі, дозволяє привернути інтернет-трафік на комерційний ресурс без використання пошукових систем, що дає можливість гнучко вибирати комунікаційні канали в залежності від цільової аудиторії. Після проведення дослідження були визначені основні елементи моделі SMM, які представлені наступним чином:

$$WSMM =< C,B,RM,PB > \quad (2.1)$$

*C* - показник ком'юніті-бренду, який визначається шаблоном взаємодії в спільноті та часовими показниками актуальності інформації про брендовані товари;

*B* - показник блогосфери, який відображає загальну кількість блогів, що стосуються комерційного інтернет-ресурсу (товари, послуги і т.д.);

*RM* - показник репутаційного менеджменту компанії (бренду);

*PB* - показник персонального брендингу, що включає в себе інформацію, створену представниками компанії та зацікавленими особами.

Давайте детальніше розглянемо особливості цих елементів моделі.

Комюніті бренду - це об'єднання користувачів навколо продукту, товару або послуги. Цей елемент моделі є одним з найвагоміших факторів просування, оскільки він не вимагає значних бюджетних витрат і надає можливості для впровадження корпоративних цінностей та стратегій розвитку. Ком'юніті сприяє підвищенню лояльності до бренду, а серед прихильників потенційний покупець може знайти як нових знайомих, так і друзів, що призводить до інтеграції бренду в життя користувача. Facebook є прикладом сервісу, який використовує цей підхід, дозволяючи користувачам виражати свої уподобання стосовно певного бренду, товару або послуги та автоматично повідомляти про це інших учасників

групи. Цей підхід змінює структуру взаємодії між новою інтернет платформою та користувачами, створюючи "тристоронні відносини": “ІП” - користувач - потенційний користувач. Комунікація на цих рівнях розглядається окремо: “ІП”-користувач, користувач-потенційний користувач та потенційний користувач-”ІП" (рис. 2.1).

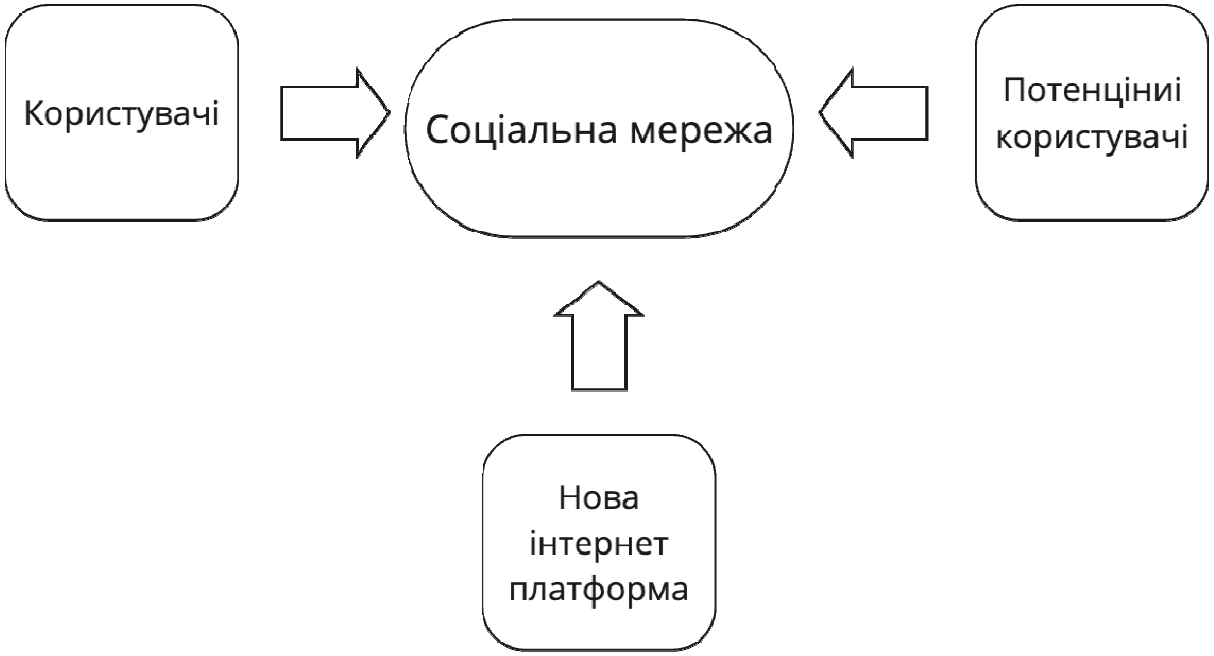



Рисунок 2.1 – Структура процесу комунікації з використанням соціальних медіа

Згідно з класифікацією K.Komaromi, структура комюніті може бути утворена пулами, мережами або вузлами [19]. Структура "пулу" характеризується мінімальною кількістю взаємозв'язків між окремими особами комюніті. Представники цієї структури часто проявляють прив'язаність до конкретної компанії, її продукції та цінностей (наприклад, користувачі торгової марки Apple). У структурі "мережі" користувачі активно взаємодіють між собою та з відповідною групою, і ця взаємодія є ключовим фактором належності до відповідного співтовариства (наприклад, групи інтересів у соціальних мережах). У "вузлах" користувачі мають стійкі взаємозв'язки з центральним елементом (наприклад, кумиром), але між собою взаємозв'язки є слабкими (наприклад, співтовариство підписників Елона Маска).

Дослідження показало, що для розвитку комерційного інтернет-ресурсу доцільно використовувати гібридну схему ком'юніті, що буде вибиратися залежно від об'єкта просування та очікуваного результату популяризації.

Термін "блогосфера" відноситься до каналу комунікації, що виник з розвитком Інтернету і включає сукупність інтернет-блогів, які зосереджені на певній темі і призначені для пошуку та обміну інформацією. Вони зазвичай мають одного автора і можливість коментування. У контексті комерційного просування часто використовуються бізнес-блоги, які підтримують відповідні маркетингові відділи замовника і стають важливим комунікаційним каналом з кінцевим користувачем. Авторитетність конкретного блогу напряму залежить від кількості підписників та репостів на його сторінки.

"Репутаційний менеджмент" означає процес створення та управління іміджем компанії в Інтернеті за допомогою контент-маркетингу та пошукової оптимізації. У цифровому просторі діяльність будь-якої компанії стає абсолютно прозорою. В такому контексті завжди існують користувачі або конкуренти, які можуть бути незадоволені брендом і мають намір погіршити його репутацію. Це може призвести до втрати потенційних клієнтів та зниження рейтингу компанії у пошукових системах.

Згідно з даними аналітичної компанії Deloitte, понад 70% користувачів читають відгуки перед придбанням товару або послуги, і майже 90% з них відмовляється від замовлення у компанії з негативною репутацією. В результаті проведеного дослідження була розроблена модель "репутаційного менеджменту", яка включає наступні складові:

- емоційну привабливість;
- якість товарів та послуг;
- відгуки;
- досягнення;
- необхідність товарів та послуг.

Просування комерційного інтернет-ресурсу у соціальних медіа може включати стратегію персонального брендингу. Сучасні технології пошукової оптимізації надають широкий спектр інструментів для контролю за брендом компанії. Коли потенційний клієнт не має досвіду співпраці з даною компанією, він вводить назву компанії в пошукову систему. Отримана інформація, яка з'являється в результаті пошуку, утворює особистий бренд. Особливо важлива є перша сторінка пошукової видачі, оскільки на її основі потенційний клієнт формує своє уявлення про компанію.

З огляду на це, інформація, яку компанія розміщує в мережі, створює цифровий слід (DF), а відгуки та відомості, залишені сторонніми людьми про компанію, утворюють цифрову тінь (DS). Сукупність цих елементів формує поняття про персональний бренд компанії.

Модель SMO (SocialMediaOptimization) включає в себе внутрішні технічні роботи, спрямовані на покращення ефективності взаємодії комерційного інтернет-ресурсу з соціальними медіа. Основні складові цієї оптимізації включають налаштування елементів інтерфейсу (наприклад, інтеграцію з популярними соціальними мережами) і актуалізацію інформаційного наповнення. Принципи SMO включають наступні аспекти:

Створення доступного та зрозумілого контенту: Створення актуального інформаційного наповнення, яке привертає увагу потенційної аудиторії і сприяє підвищенню конверсії комерційного інтернет-ресурсу.

Додавання релевантної інформації для цільової аудиторії: Контент повинен бути відповідним інтересам цільової аудиторії. Наприклад, технічні спеціалісти можуть бути зацікавлені в інформації про програмування, а не в материнському вихованні.

Зручність користування (юзабіліті): Використання кнопок та полів, таких як "Поділитися", "Retweet", "Підписатися" та "Коментувати", що надають зручність користування та засоби для спілкування з цільовою аудиторією.

Додавання мультимедійних елементів (встановлення віджетів): Використання інформаційних стрічок та віджетів для відображення основних новин ресурсу на сторінці користувача, що збільшує час відвідування комерційного ресурсу і сприяє SMO-оптимізації.

З огляду на ці фактори, виокремлено дві стратегії просування: пасивну та активну. Пасивна стратегія включає такі кроки (рис. 2.2).

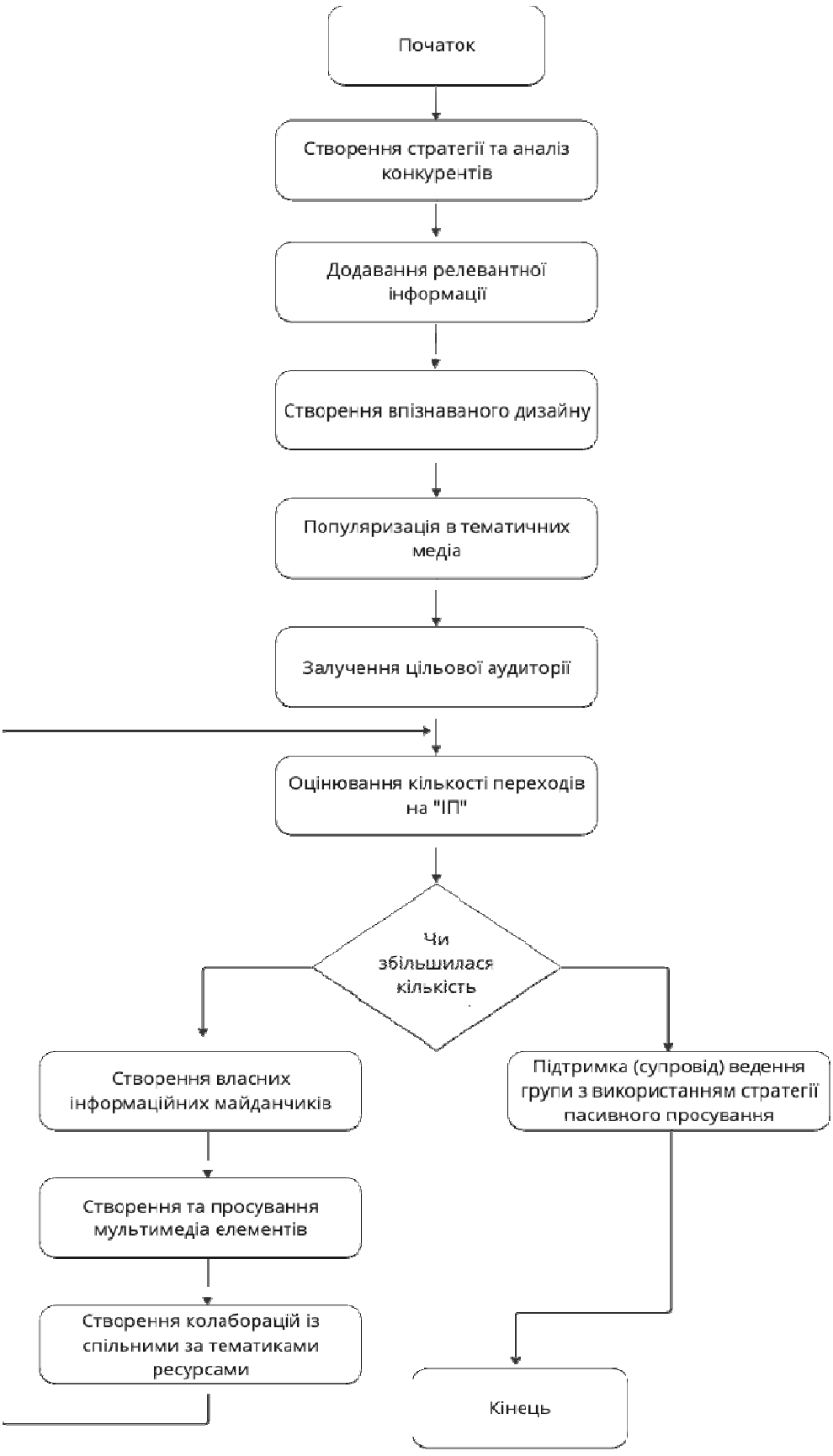



Рисунок 2.2 – Алгоритм популяризації інтернет-ресурсів при використанні технологій соціальних медіа

Побудова концепції та аналіз конкурентів: Визначення мети створення групи і аналіз популярних сторінок конкурентів у соціальних медіа для побудови концепції власного представництва.

Наповнення групи релевантною інформацією: Створення корисного контенту, який збуджує бажання користувачів поділитися ним з друзями і формує впізнаваний голос бренду.

Проведення персонального брендингу: Популяризація інформації про комерційні особливості бренду або бізнесових характеристик окремих працівників для привертання потенційних клієнтів.

Популяризація в тематичних медіа: Поширення контенту на новинних сервісах, геолокаційних сервісах та за допомогою рекомендаційних систем для залучення уваги до бренду.

Залучення цільової аудиторії: Організація онлайн та офлайн заходів для знайомства користувачів з продукцією бренду, а також використання спеціальних спецпроектів і сторітелінгу для мотивації суб'єктів дії.

Оцінювання кількості переходів на комерційний ресурс: Моніторинг соціальних медіа та використання сервісів моніторингу для аналізу ефективності просування і оцінки дій конкурентів.

Ці кроки допоможуть покращити ефективність використання соціальних медіа для просування комерційного інтернет-ресурсу.

Якщо досягнуті результати відповідають обраній стратегії просування, тоді починається процес підтримки або супроводу групи. У противному випадку застосовується активна стратегія просування, що включає такі кроки:

Створення власних інформаційних майданчиків: Цей крок передбачає ряд заходів, таких як створення корпоративного блогу, написання гостьових постів на тематичних ресурсах і ведення активної комунікації з аудиторією.

Проведення інтерактивних акцій: Реалізація флешмобів, опитувань і інших інтерактивних заходів, що зацікавлять та залучать увагу цільової аудиторії.

Створення та просування мультимедійних елементів: Розробка маркетингових роликів, віджетів та трансляція новинних стрічок на популярних інформаційних ресурсах для привертання уваги до бренду та розширення аудиторії.

Створення колаборацій з іншими ресурсами: Укладання партнерських угод та співпраця з суміжними брендами, що дозволяє обмінюватися аудиторіями та підвищувати впізнаваність серед цільової аудиторії.

Ці кроки допомагають активно просувати бренд, залучати увагу цільової аудиторії та підвищувати його впізнаваність.

Враховуючи активне використання соціальних медіа різними верствами населення, та зважаючи на високе охоплення аудиторії, беручи до уваги необхідність просування нової інтернет платформи в обмежені строки задля досягнення більшої аудиторії, найефективнішою початковою стратегією просування нової інтернет платформи вважаю саме SMM метод, а саме популяризацію інтернет-ресурсу або послуг компанії за допомогою соціальних медіа.

Наступним етапом в стратегії просування продукту, після досягнення певного числа користувачів можна використати SMO підхід.

## Розділ 3. Математичне забезпечення

### 3.1 Зміст та застосування

Математичне забезпечення представляє собою сукупність методів, правил, математичних моделей і алгоритмів, що використовуються для вирішення задач. Це може бути загальне математичне забезпечення, яке орієнтоване на організацію обчислювального процесу на конкретній електронно-обчислювальній машині (ЕОМ), або спеціальне математичне забезпечення, призначене для розв'язання конкретних завдань.

Рівень розвитку математичного забезпечення визначає ефективність використання певної інформаційної технології. Тенденція в сучасності полягає в збільшенні частки витрат на розроблення математичного апарату в загальних витратах на проект інформаційної системи.

Фахівці з організаційно-технологічних рішень, як постачальники проблемних задач керування, та фахівці з формалізації процесу прийняття управлінських рішень відповідають за побудову математичних моделей задач керування. Важливо, щоб спрощення процесу моделювання були обґрунтовані, аби уникнути непотрібних узагальнень у процесі керування. Також варто зауважити, що потреби в інформатизації виробництва покищо випереджають можливості прикладної математики, що може призводити до значного спрощення моделей, особливо у зв'язку із застосуванням лінійних моделей, хоча більшість залежностей в економіці та управлінні підприємством є нелінійними. Останнє десятиліття характеризується значним розвитком математичних дисциплін, методи яких використовуються для вирішення завдань в інформаційних системах.

### 3.2Методи математичного забезпечення

*Мережеві методи* широко використовуються в проектуванні, дозволяючи визначати параметри мережевих моделей та аналізувати виконання виробничих планів. У межах мережевого моделювання може проводитися одно- або багато-критеріальна оптимізація, включаючи оптимізацію за часом та ресурсами.

*Евристичні методи* застосовуються для вирішення слабко структурованих задач, які не можуть бути ефективно вирішені повним перебором варіантів, наприклад, задачі календарного планування. Есенція евристичного методу полягає в тому, щоб планувати роботи найкоротшим шляхом, ураховуючи встановлений рівень ресурсів. Зазвичай використання евристичних методів передбачає взаємодію з користувачем, під час якої обчислення та проміжні результати представляються на комп'ютері, включаючи графіки та діаграми. Користувач, залежно від отриманих даних, визначає подальший напрямок розрахунків.

*Методи комбінаторики*, математичної логіки та інформаційної алгебри використовуються для розв’язання інформаційно-логічних задач, таких як групування та впорядкування даних, об'єднання масивів даних та коригування інформації.

*Математичне програмування* об'єднує лінійне, нелінійне, динамічне і стохастичне програмування. Транспортні задачі, які розв'язуються за допомогою лінійного програмування, є особливо визначеними. Лінійне програмування використовується для аналізу та розв'язання питань, таких як розробка та складання прогнозів розвитку галузей та оптимальний розподіл ресурсів.

*Нелінійне математичне програмування* застосовується рідше, і часто нелінійні задачі розв'язуються методами лінійного програмування за допомогою лінеаризації криволінійних залежностей. Динамічне програмування використовується для розподілу капітальних вкладень, календарного планування, пошуку оптимальної послідовності постачання товарів та управління запасами. Сутність динамічного програмування полягає в тому, що довший шлях до результату відкидається для зменшення обсягу обчислень на ЕОМ.

*Стохастичне програмування* характеризується введенням ймовірнісних значень параметрів, що відображають ризик і невизначеність.

*Методи теорії ігор* використовуються для формалізації та розв'язання задач, які зазвичай розв'язуються емпірично без використання кількісних

вимірників, наприклад, аналіз конфліктних ситуацій в умовах невизначеності інформації про дії учасників.

*Теорія черг або масового обслуговування* досліджує імовірнісні моделі поведінки систем, а основою для вирішення задач масового обслуговування є теорія ймовірностей. Математична статистика, яка є розділом теорії ймовірностей, дозволяє оцінювати певні сукупності даних. Метод статистичних іспитів також використовується для вивчення імовірнісних систем і застосовується при моделюванні різних ситуацій.

*Метод теорії розкладів* дозволяє знаходити оптимальну послідовність побудови об'єктів за певним критерієм. Наприклад, критерієм може бути "найменший термін будівництва", "мінімум простоїв виконавців на об'єктах" або "максимальна щільність робіт на об'єктах".

*Методи теорії множин* дозволяють більш компактно описувати задачі керування та знаходити ефективні шляхи їх вирішення.

### 3.3 Роль та виклики

Математичне забезпечення використовується для побудови математичних моделей для задач керування, що відіграє важливу роль у вирішенні проблем, пов'язаних з організаційно-технологічними вирішеннями та формалізацією процесів прийняття управлінських рішень. Важливим аспектом в даній діяльності є якісне математичне підґрунтя для розроблення обґрунтованих моделей, уникнення зайвого спрощення та врахування інформатизації виробництва.

Останнє десятиліття відзначається значним розвитком математичних дисциплін, які використовуються в інформаційних системах. Методи, такі як мережеві моделі, евристичні методи та теорія ігор, широко використовуються в проектуванні та розв'язанні складних завдань.

Сучасні виклики - використання лінійних моделей у сфері економіки та управління підприємствами, хоча є стандартним, проте не завжди адекватно враховує нелінійні аспекти цих сфер. Потреби в інформатизації виробництва

випереджають можливості прикладної математики, що викликає значне спрощення моделей та їхню ефективність.

## 3.4Використані матиматичні моделі

Основні математичних моделі використаних  при розробці данного проекту: Алгоритми та не чітка логіка.

Алгоритми та нечітка логіка в синергії надають потужний інструментарій для вирішення складних завдань у математичному забезпеченні, дозволяючи враховувати різноманіття ситуацій та невизначеність в реальних задачах.

### 3.4.1 Алгоритми

Алгоритми представляють собою конкретні інструкції або кроки, які вказують, як вирішити конкретну задачу чи виконати конкретне завдання. В контексті математичного забезпечення, алгоритми використовуються для вирішення різних завдань, включаючи обчислення, оптимізацію, сортування, пошук, моделювання та інші.

В інформаційних технологіях алгоритми є основним елементом розробки програмного забезпечення. Вони дозволяють ефективно вирішувати завдання на різних рівнях, від найнижчого рівня обчислень до високорівневого програмування.

#### 3.4.1.1 Види Алгоритмів

Сортування:Алгоритми, які впорядковують набір даних за певним критерієм, наприклад, алгоритм швидкого сортування чи сортування злиттям.

Пошук:Алгоритми для знаходження конкретного елемента у впорядкованому або неупорядкованому списку, такі як бінарний пошук або пошук за зразком.

Графи:Алгоритми для роботи з графами, включаючи пошук найкоротших шляхів, топологічне сортування та інші.

Машинне Навчання:Алгоритми, використовувані для навчання моделей та вирішення завдань у галузі штучного інтелекту.

### 3.4.1.2 Нечітка Логіка

Нечітка логіка є математичним підходом, який дозволяє враховувати нечіткість та невизначеність в прийнятті рішень. Замість традиційного "істинно" чи "хибно", нечітка логіка дозволяє призначити ступінь істинності від 0 до 1, що відображає ступінь приналежності об'єкта певному класу.

Елементи Нечіткої Логіки:

Функції Приналежності:Визначають ступінь приналежності елемента до певної множини.

Логічні Операції:Операції "І", "АБО", "НІ" застосовуються до нечітких значень, враховуючи ступені приналежності.

Висновок на Основі Правил:Використання правил, які базуються на нечітких умовах, для прийняття висновків.

У сфері інформаційних технологій, нечітка логіка використовується в системах управління, прийнятті рішень, обробці сигналів, та в інших областях, де важливо враховувати невизначеність та нечіткість.

Цю математичну модель використано для визначення ступеня зацікавленості типами подій, результати в подальшому використані при фільтрування данних в умовах відсутності збігів у запиті.

Нечітка логіка може використовуватися для визначення найбільш відвідуваних типів подій на основі нечітких правил і множників приналежності. Для прикладу, давайте розглянемо ситуацію, де є певні типи подій, і ми хочемо визначити, які з них є часто відвідуваними, а які - рідко відвідуваними. Для цього ми можемо використати нечіткі множини та правила нечіткої логіки.

Припустимо, у нас є типи подій: A, B, Сі …. Ми хочемо визначити ступінь належності кожної події до категорії "часто відвідувані" та "рідко відвідувані". Ми можемо використовувати три нечіткі множини для кожної події:

Часто відвідувані:

Множина A_часто, B_часто, C_часто, … _часто.

Рідко відвідувані:

Множина A_рідко, B_рідко, C_рідко, … _рідко.

Кожні множини можуть мати свої функції належності, які визначають, наскільки події A, B,…і n - належать до відповідних категорій.

Потім ми можемо визначити нечіткі правила для визначення ступеня частості відвідування:

Якщо A_часто велике, то A - часто відвідувана.

Якщо B_часто велике, то B - часто відвідувана.

Якщо …_часто велике, то … - часто відвідувана.

Якщо n_часто велике, то n - часто відвідувана.

Якщо A_рідко велике, то A - рідко відвідувана.

Якщо B_рідко велике, то B - рідко відвідувана.

Якщо …_рідко велике, то … - рідко відвідувана.

Якщо n_рідко велике, то n - рідко відвідувана.

Ці правила використовують ступені належності для визначення, наскільки події відповідають визначеним категоріям. Формули для розрахунку ступенів належності можна знайти нижче:

Операцію максимізації використано для визначення ступіня належності об'єднання подій оскільки важливо визначити, яка подія має найбільший вплив чи найвищий ступінь важливості серед усіх вхідних подій.

Ступінь_належності_взаємодія=max(μA(A),μB(B),...,μn(An))

Ця формула використовує операцію максимізації (max max), щоб визначити максимальний ступінь належності серед усіх подій A, B, ..., n.

де

μA - ступінь належності для події A,

μB - ступінь належності для події B,

μn - ступінь належності для події n.

### 3.4.1.3Алгорим пошуку з використанням нечіткої логіки

У попередньому розділі описано реалізаця визначення найбільш шуканих слів. Використовуючи нечітку логіку було досягнуто об'єднання значень ступенів відповідності для кожного слова. Знайденийрезультат використано уалгорітмі пошуку.

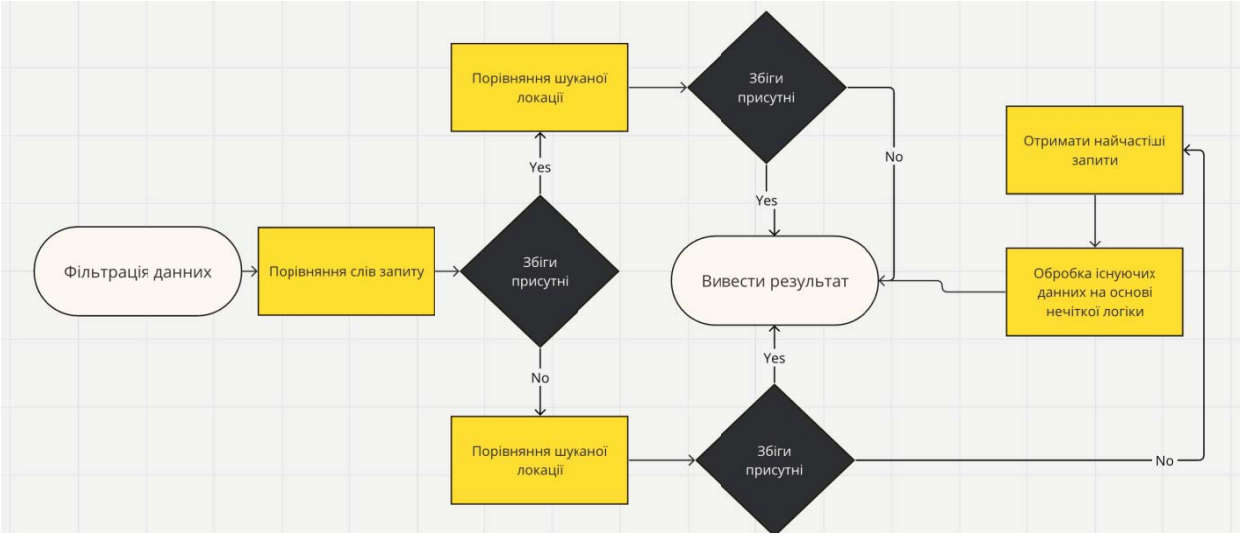



Рисунок 3.1 Алгоритм пошуку з використанням нечіткої логіки

За допомогою цього алгоритма платформа дає змогу аналізувати збіги в запиті і виводити інформації про них, І/АБО шукану локацію, І/АБО найбільш шукані події.

## 3.5Вдосконаланення математичного забезпечення

### 3.5.1 Представлення інтересів користувачів у вигляді векторів або матриць

Створення математичної моделі для представлення інтересів користувачів у вигляді векторів або матриць і використання алгоритмів кластеризації для групування користувачів зі схожими інтересами - це типовий завдання в області рекомендаційних систем. Давайте розглянемо цей процес крок за кроком:

Вектор користувача Ui=[I1,I2,...,In]

де I1,I2,...,In - це інтереси користувача, представлені числовими значеннями або вагами.

1.Представлення інтересів користувачів у векторній формі:

Створіть вектор або матрицю для представлення інтересів користувачів. Кожен користувач може бути представлений вектором, де кожний елемент відповідає конкретній характеристиці або категорії інтересу. Наприклад:

Вектор користувача

- це інтереси користувача, представлені числовими значеннями або вагами.

2. Збирання даних:

Зібрати дані про інтереси користувачів. Це може бути інформація про перегляди, покупки, оцінки чи інші дії користувачів в системі.

3. Векторизація тексту або числові оцінки:

Якщо ви маєте текстові дані, такі як описи інтересів, використовуйте методи векторизації тексту (TF-IDF, wordembeddings) для перетворення тексту в числовий вектор. Якщо у вас вже є числові оцінки, використовуйте їх безпосередньо.

4. Вибір алгоритму кластеризації:

Виберіть алгоритм кластеризації, такий як K-Means, агломеративна кластеризація, DBSCAN тощо. Кількість кластерів може бути визначена апріорі або визначена за допомогою методів визначення оптимальної кількості кластерів (наприклад, метод ліктя).

5. Навчання моделі:

Застосуйте алгоритм кластеризації до векторів інтересів користувачів. В результаті отримаєте групи користувачів зі схожими інтересами.

6. Аналіз та інтерпретація кластерів:

Аналізуйте отримані кластери. Це може допомогти вам зрозуміти, які групи інтересів є великими серед користувачів і як вони відрізняються одна від одної.

7. Рекомендації:

Після створення кластерів ви можете використовувати їх для рекомендацій. Наприклад, рекомендувати користувачеві події чи інтереси, які популярні серед інших користувачів у тому ж кластері.

8. Оптимізація та оновлення:

Оптимізуйте алгоритм та періодично оновлюйте кластери та модель на основі нових даних.

Цей підхід дозволяє автоматично групувати користувачів за їхніми інтересами та надавати рекомендації на основі схожості.

### 3.5.2 Алгоритми кластерізації

Кластеризація є технологією машинного навчання, яка використовується для групування точок даних. Подавши набір даних на вхід, алгоритм кластеризації дозволяє класифікувати кожну точку даних у відповідну групу. Властивості точок даних, які потрапляють в одну групу, повинні бути схожими, тоді як властивості точок даних у різних групах мають багато відмінностей. Кластеризація відноситься до методів навчання без учителя та знаходить застосування в різних галузях.

Існує два основних типи кластеризації: жорстка та м'яка. У жорсткій кластеризації кожна точка даних повністю належить або не належить конкретному кластеру. З іншого боку, у м'якій кластеризації, замість визначення чіткої належності, кожній точці даних призначається ймовірність належності до кожного з кластерів.

Різні методи кластеризації мають свої моделі та алгоритми, проте їхня спільна мета полягає в об'єднанні схожих об'єктів у групи. Один із найвідоміших методів - метод К-середніх, спрямований на побудову заданої кількості кластерів, розташованих якнайдалі один від одного.

Алгоритм К-середніх має кілька етапів, які можна описати наступним чином:

Ініціалізація центрів:

Спочатку випадковим чином вибираємо масив з К центральних точок. Оцінка значення К може визначатися аналізом даних для визначення кількості окремих угруповань.

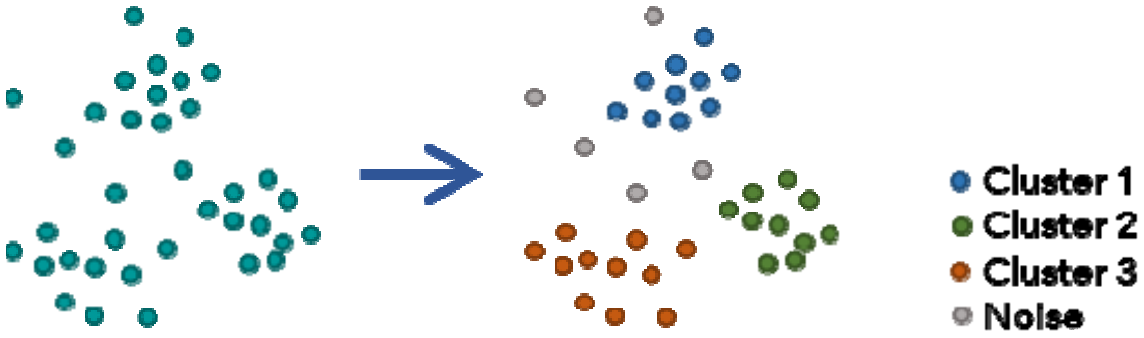



Рисунок 3.2 Візулізація кластеризації

Класифікація точок:

Для кожної точки даних обчислюємо відстань між цією точкою та кожною з центральних точок. Потім відносимо точку до групи з найближчим до неї центром.

Перерахунок центрів:

Виходячи з класифікованих точок, перераховуємо центр групи, взявши середнє значення всіх векторів групи.

Повторення:

Повторюємо кроки 2-3 задану кількість разів або до тих пір, поки зміна центрів груп на кроці 3 не буде менше наперед заданого значення.

Перевага методу К-середніх полягає в його швидкості. Оскільки він має лінійну складність O(n), цей алгоритм ефективний при обробці великих обсягів даних. Крім того, він є ефективним і для різноманітних типів даних та групувань.

З іншого боку, у методу К-середніх існує декілька недоліків. По-перше, потрібно заздалегідь визначати кількість кластерів для розбиття, що не завжди є очевидним. По-друге, результат роботи алгоритму залежить від того, яким чином ми виберемо початкові центри кластерів. Таким чином, виконуючи алгоритм декілька разів, можемо отримати різні результуючі кластери. У порівнянні з іншими методами кластеризації, метод К-середніх менш однозначний.

Метод К-медіан є модифікацією методу К-середніх, де при перерахуванні центральних точок кластеру використовується не середнє значення векторів, а медіанне значення векторів даної групи. Цей метод менш чутливий до викидів, завдяки використанню медіани, проте значно повільніший для великих наборів

вхідних даних, оскільки для кожної ітерації, при обчисленні медіанного вектору групи, потрібно виконувати їх сортування.

Метод зсуву середнього значення є алгоритмом, який проводить пошук області точок даних з найвищою щільністю. Основна мета алгоритму полягає в знаходженні центральних точок кожного кластеру.

## 3.6. Висновки

Математичне забезпечення в сучасному світі є ключовим елементом для розвитку наук, технологій та індустрії. Використання математичних методів дозволяє розв'язувати складні проблеми, розробляти нові технології, покращувати прийняття рішень та вдосконалювати наш розуміння навколишнього світу.

Математика взаємодіє з різними галузями науки і промисловості, використовуючи абстракції, логіку та аналітичні методи для вирішення реальних завдань. Вона є мовою, яка дозволяє точно висловлювати ідеї та створювати конкретні моделі для подальшого вивчення.

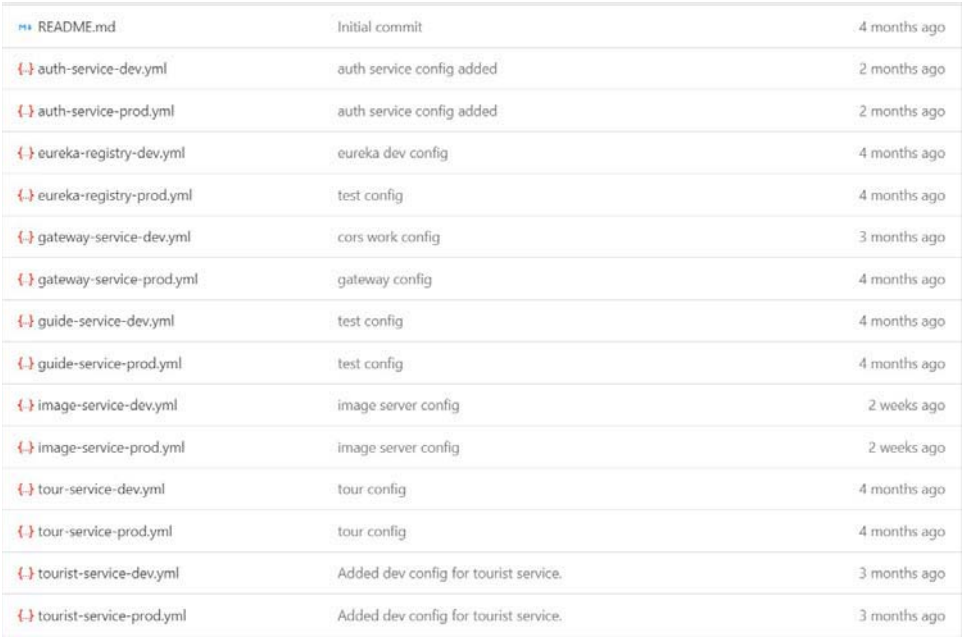
Математичне мислення розвиває критичний та аналітичний спосіб мислення, навички розв'язання проблем та творчого підходу до вирішення завдань. Воно є основою для навчання в інших науках, інженерії та технологіях.

У сучасному інформаційному суспільстві, де дані та технології відіграють важливу роль, математичне забезпечення визначає успіх у різних галузях, від науки і освіти до бізнесу і інновацій. Таким чином, володіння математикою вважається необхідним для розвитку і успіху в сучасному світі.

## Розділ 4. Програмне забезпечення

### 4.1 Налаштування мікросервісів

Коженсервісвикористовуєспеціальніфайликонфігурацій,дляналаштування, при його запуску, особливо, коли існує декілька середовищ депрограма може бути запущена. Для централізованого зберігання всіх файлівпрофілівзапуску,різнихмікросервісів,булостворенноспеціальнийGitLabрепозиторійде вонизберігаються (рис.4.1).



| | | |
|---|---|---|
| README.md | Initial commit | 4 months ago |
| {-} auth-service-dev.yml | auth service config added | 2 months ago |
| {-} auth-service-prod.yml | auth service config added | 2 months ago |
| {-} eureka-registry-dev.yml | eureka dev config | 4 months ago |
| {-} eureka-registry-prod.yml | test config | 4 months ago |
| {-} gateway-service-dev.yml | cors work config | 3 months ago |
| {-} gateway-service-prod.yml | gateway config | 4 months ago |
| {-} guide-service-dev.yml | test config | 4 months ago |
| {-} guide-service-prod.yml | test config | 4 months ago |
| {-} image-service-dev.yml | image server config | 2 weeks ago |
| {-} image-service-prod.yml | image server config | 2 weeks ago |
| {-} tour-service-dev.yml | tour config | 4 months ago |
| {-} tour-service-prod.yml | tour config | 4 months ago |
| {-} tourist-service-dev.yml | Added dev config for tourist service. | 3 months ago |
| {-} tourist-service-prod.yml | Added dev config for tourist service. | 3 months ago |

Рисунок4.1–Репозиторійконфігурацій

Файли налаштувань містять різні конфігурації, відповідно до потреб іпризначення того, чи іншого мікросервісу. Наприклад, yml файл профілю devдляGatewayміститьналаштуванняпортузапускупрограми,доступудореєстру сервісів,атакожCORS(рис. 4.2).

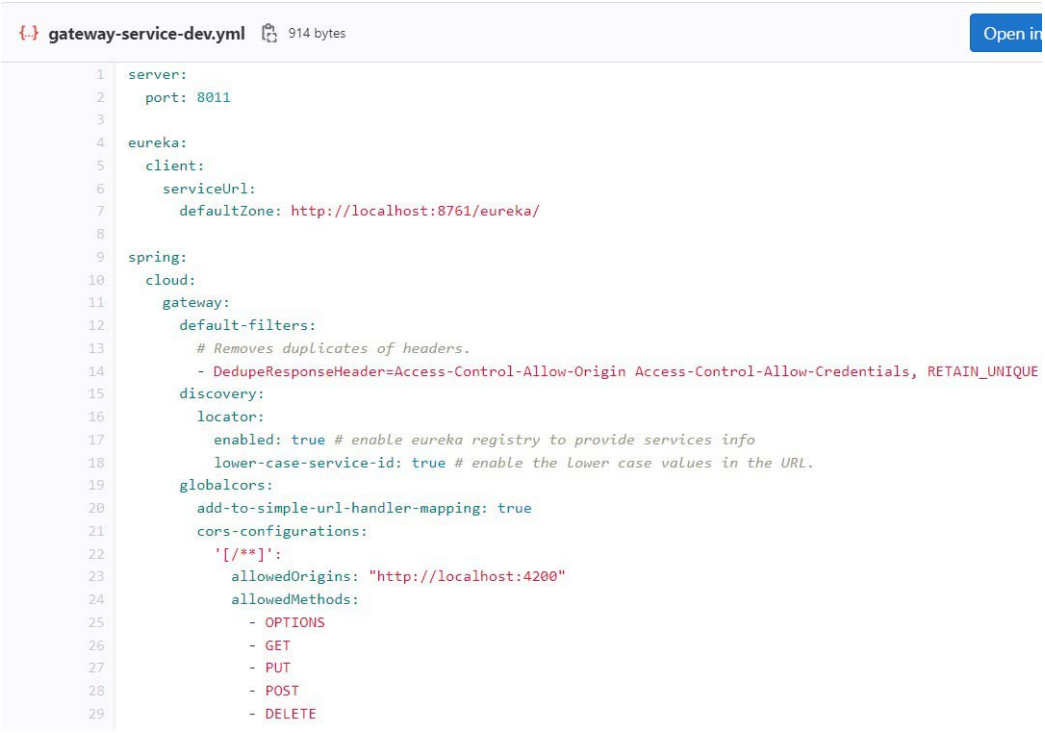



Рисунок 4.2 – Файл налаштувань профілю dev для сервісу Gateway

Також, всімікросервісимають файлналаштувань за замовчанням,якийвикористовуєтьсядлявстановленнякритичнихпараметрівдоступудоцентралізованогорепозиторію(рис.3.3).Післязапускупрограми,здійснюєтьсяпідключеннядосервісуналаштуваньімікросервісотримуєнеобхіднийдляпевногопрофілюнабірпараметрів.


Рисунок 4.3 – Файл налаштувань за замовчанням для сервісу Gateway

Длязапускумікросервісузпевнимпрофілемнеобхідновказатийого,якзмінну призапускупрограми(рис.4.4).

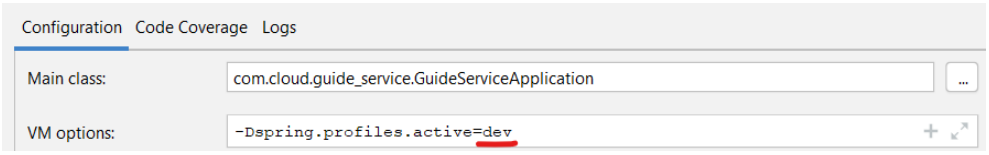



Рисунок4.4–Налаштуваннязмінноїпрофіляпередзапуском

## 4.2 Автоматична збірка модулів

Для збирання проектів було використано систему автоматичної збіркиGradle. Вона побудована на базі даних ApacheAnt та ApacheMaven, алестворюєDSLнабазіGroovyтаKotlinзамістьXML-подібноїформипредставленняпроекту[15].

Сервіси обробки данихмають схожі залежності, саме тому, міститьдужебагатооднаковихзалежностей(рис.4.5).

```
dependencies {
    implementation 'org.springframework.cloud:spring-cloud-starter-config'
    implementation 'org.springframework.cloud:spring-cloud-starter-netflix-eureka-client'
    implementation 'org.springframework.cloud:spring-cloud-starter-openfeign'

    implementation 'org.springframework.boot:spring-boot-starter-web'
    implementation 'org.springframework.boot:spring-boot-starter-data-jpa'
    implementation 'org.springframework.boot:spring-boot-starter-validation'
    implementation 'org.springframework.boot:spring-boot-starter-actuator'

    compileOnly "org.projectlombok:lombok"
    annotationProcessor "org.projectlombok:lombok"

    runtimeOnly 'mysql:mysql-connector-java'

    testImplementation 'org.springframework.boot:spring-boot-starter-test'
}
```

Рисунок4.5–ФайлавтоматичноїзбіркиGradledлясервісівданих

Кожназалежністьєкритично-необхідноюіпідсвоєюназвоюміститьвеликучастинувже готовогофункціоналу (табл.4.1).

Уданомупроекті,такожбуливикористанідопоміжнібібліотекиідрайвери. Наприклад,бібліотекаLombok,длягенераціїкодунаетапікомпіляції.Зарахунока нотацій,проводитьсяаналізвихідногокодуістворюються необхідні методи. Такий підхід зменшує кількість шаблонногокоду,покращуєчитабельністьідопомагаюсконцентруватиувагунаг оловних елементах системи. Найпопулярніші анотації, які були використаніунаявнійсистемі–цеData,NoArgConstructorAllArgConstrictor.

Таблиця4.1

Описзагальнихбібліотекмікросервісівданих

| Назвабібліотеки | Функціональність |
|---|---|
| spring-cloud-starter-config | Функціональність для клієнт-серверноїкомунікації Spring длязберіганнята обслуговуваннярозподіленихконфігурацію кількохпрограмахісередовищах. |
| spring-cloud-starter-netflix-eureka-client | ФункціональністьдлявключитиклієнтарепозиторіяEurekaупроект. |
| spring-cloud-starter-openfeign | Функціональністьдлялегкого включеннямікросервісноїкомунікації. |
| spring-boot-starter-web | Функціональність для підключення обробкивебзапитів,якавключаєHTTP-сервері контейнерServlet,якиймаємożливість обслуговуватистатичнийідинамічнийвміст. |
| spring-boot-starter-data-jpa | Функціональністьдляпідключеннязастосунку дореляційноїбазиданих. |
| spring-boot-starter-validation | Функціональністьдляперевіркикоректностіда них об’єкта,якаможебутилегко налаштованазадопомогоюанотацій. |
| spring-boot-starter-actuator | Функціональність, яка використовує кінцевіточкиHTTPдлянаданняопераційної інформації про будь-яку запущену програму.Основнаперевагавикористанняцієїбіблі отекиполягає в тому, що ми отримуємо показникипрацездатностітамоніторингуготовихд о виробництвапрограм. |

## 4.3 Моніторинг і статус

Післястворенняновогосервісу,йогоналаштуваньізапускувінповиненбуд ивідображенийуреєстрісервісів(рис. 3.6),цесвідчитьпро те, що сервіс був успішно запущений і зконфігурований. Статус програмиможе змінитися, якщо відбулись якісь проблеми і мікросервіс, наприклад,переставвідповідати.

Instances currently registered with Eureka

| Application | AMIs | Availability Zones | Status |
|---|---|---|---|
| AUTH-SERVICE | n/a (1) | (1) | UP (1) - localhost:auth-service:8055 |
| CONFIG-SERVER | n/a (1) | (1) | UP (1) - localhost:config-server:8888 |
| GATEWAY-SERVICE | n/a (1) | (1) | UP (1) - localhost:gateway-service:8011 |
| GUIDE-SERVICE | n/a (1) | (1) | UP (1) - localhost:guide-service:8020 |
| IMAGE-SERVICE | n/a (1) | (1) | UP (1) - localhost:image-service:8040 |
| TOUR-SERVICE | n/a (1) | (1) | UP (1) - DESKTOP-SVIK3I8.fritz.box:tour-service:8021 |
| TOURIST-SERVICE | n/a (1) | (1) | UP (1) - localhost:tourist-service:8023 |

Рисунок4.6–Реєстрмікросервісів

За потреби отримувати більш детальну інформацію, а також кастомніметрики для бізнесу, можна інтегрувати такі системи як Prometheus абоInfluxDB, які дозволяють здійснювати гнучкий моніторинг внутрішніхпроцесівсистеми.

## 4.4 Призначення мікросервісів

Кожен компонент системи відповідальний за певні унікальні функції.Саме такий підхід, мається на увазі, при побудові мікросервісної архітектури.Коженнесевідповідальністьзапевнуфункціональність.Такимчином,зазвичай при розробці системи на мікросервісній архітектурі, створюєтьсявелика кількість незалежних програм, які можуть комунікувати за рахуноквідкритих інтерфейсів.

Данасистемаміститьрізнісервісизрізнимицілямиіфункціями(табл. 3.2),якіповинніякісновиконуватитількитіоперації,якізнаходяться у колі їх відповідальності і не перебирати чужі функції. Цимпідходомуникаєтьсяповторневикористаннякоду.Дуплікаціялогікиєодніє юзнайгіршихпрактик упрограмуванні.

Таблиця4.2

Функціональнівідповідальностімікросервісів

| Назвамікросервісу | Функціональнавідповідальність |
|---|---|
| Config-server | Надаєконфігурації їншиммікросервісамз репозиторіюконфігурацій. |
| Eureka-registry | Реєструєімоніторитьстанвсіхіншихсервісів.Також, має важливу роль у перенаправленні запитів,такякможенадаватиінформаціюпророзташування мікросервісузайогоім'ям. |
| Gateway-service | Приймаєзапитивідклієнтськихзастосунківіпередаєїїдо певногомікросервісубазуючисьна URL. |
| Guide-service | Оброблюєвсюінформаціюпробізнескористувачів,асам еїхперсональнідані–їхзбереження, оновленняі видалення. |
| Tourist-service | Оброблюєвсюінформаціюпротуристів,асамеїх персональнідані–їхзбереження,оновленняівидалення. |
| Tour-service | Оброблює всю інформацію про пропозиції від гідів, асаметідані,якізавантажуютьсябізнескористувачем проактивність,крімзображень. |
| Comment-service | Оброблюєвсюінформаціюпрокоментарідотурів. |
| Image-service | Відповідальнийзазбереженняівидаленнязображень длятурів. |
| Auth-service | Відповідальнийзавхідвсистему,асаме,завидачу токенівдоступу. |

## 4.5 Внутрішні структури

Як уже було описано раніше, у даній системі використовується підхідоб'єктно-реляційного і об'єктно-документного відображення. Такий підхіддозволяє описати структуру бази даних, безпосередньо у коді програми ізгенерувати всі необхідні таблиці в базі при завантаженні застосунку. Такимчином, наприклад, описується модель туру, як клас у систему, додаютьсяспеціальніанотаціїдлявзаємодіїзбазоюіприкомпіляціївідбуваєтьсяге нераціятаблиціінеобхіднихпараметрівцієїжтаблицітурів(рис.4.7).

```java
@Data
@Entity
@Table(name = "tours")
public class Tour implements Serializable {

    @Id
    @GeneratedValue(generator = "uuid2")
    @GenericGenerator(name = "uuid2", strategy = "uuid2")
    @Column(name = "tour_id", updatable = false, nullable = false, columnDefinition = "VARCHAR(36)")
    @Type(type = "uuid-char")
    private UUID id;

    @Column(name = "name")
    private String name;

    @Column(name = "description")
    private String description;

    @Column(name = "start_date")
    private Date startDate;

    @Column(name = "price")
    private float price;

    @Column(name = "duration")
    private long duration;

    @Column(name = "main_img_id")
    private String mainImgId;
```

Рисунок 4.7 – Клас відображення моделі туру без полів зв’язків

Існуютьрізнізв'язкиіобмеженняубазіданихміжрізнимитаблицями(табл. 4.3), тому правильне створення цих відносин є дуже необхідним, віншомувипадку,даніможутьнезберігатицілісністьпризберігання,оновленні чи видаленні або навіть пошуку. Для побудови зв’язків на рівнімоделі об’єкту вкоді програмивикористовуютьсяспеціальніанотації, тфкіякOneToOne,OneToMany,ManyToManyiManyToOne (рис.4.8).

```java
@Column(name = "guide_id", nullable = false, updatable = false, columnDefinition = "VARCHAR(36)")
private String guideId;

@OneToOne(cascade = CascadeType.ALL, fetch = FetchType.LAZY, mappedBy = "tour")
@JoinColumn(name = "loc_id", referencedColumnName = "loc_id")
private Location location;

@ManyToOne(cascade = CascadeType.MERGE, fetch = FetchType.LAZY)
@JoinColumn(name = "type_id", referencedColumnName = "type_id")
private TourType type;

@CreationTimestamp
@Column(name = "created_at", updatable = false, nullable = false)
private Timestamp createdAt;

@UpdateTimestamp
@Column(name = "updated_at")
private Timestamp updatedAt;
```

Рисунок 4.8 – Клас відображення моделі туру з полями зв’язків

Таблиця4.3

Описанотаційоб’єктно-реляційноговідображеннямоделі

| Анотація | Призначення |
|---|---|
| @Entity | Модельмаєвикористовуватись,як сутністьубазідани. |
| @Table | Повиннябутистворенатаблицязпевним мім’ям,якщозаданийпараметрname.По озамовчанню беретьсяім’якласумоделі. |
| @Id | Обов’язковаанотаціяприствореннюб ’єктно-реляційного/документноговідображення.Позначає,щополе використовується,якідентифікатор. |
| @GeneratedValue | Вказуєнате,якимчиномбуде генеруватисязначення. |
| @Column | Позначає, що поле це колонка, хочавсіполяавтоматичностанутьстовпчикамиутаблиці.Параметрnameвідповідаєзаім’яколонкиу базі,uniquеза унікальністьзначення,nullableзаможливістькомірки прийматизначенняnull. |
| @OneToOne,@OneToMany, @ManyToMany,@ManyToOne | Анотації,якіописуютьвідносини сутностей всховищіданих. |

| | |
|---|---|
| @CreationTimestamp, @UpdateTimestamp | Спеціальніанотаціїдлявідстеженняч асустворенняіоновленняоб’єквівусхо вищі відповідно. |

Використанняанотаційузастосункахнабазіплатформи Springєгарною практикою для створення різних типів зв’язків і об’єктів програми.Такяквикористанийфреймворкреалізуєпатернпроектування– інверсіїконтроля,асамевпровадженнязалежностейтоможназалишатисядужегн учкимустворенніоб’єктіввглобальномуконтейнері,апотімвикористовуватиїху різниммісцяхпрограми.Якщопозначитикласанотацією @Service, @Repository, @RestConrtoller або @Component то Springбуде намагатися створити об’єкт такого класу і зареєструвати його в своємуконтейнері DI (рис. 4.9).

```
@Service
public class TourServiceImpl implements TourService {
```

Рисунок4.9–Використанняанотації@Service

Інколи необхідно створити об’єкт вручну бо система, просто, не знаєзвідкистворитипевнийелемент,втакомувипадкувикористовуєтьсяанотація @Beanізазвичай,реалізуєтьсяпевнийметод,якийповертаєбажанийоб’єкт(рис. 4.10).


Рисунок4.10–Описствореннявоб'єктівзанотацією@Bean

Післястворенняневідхідноб'єктіввконтейнерівпровадженнязалежностей, їх можна отримати трьома шляхами, а саме через поле, сеттерабоконструкторукласі,ціелементимаютьбутипоміченіанотацією@Autowired (рис. 4.11). Зазвичай використовують конструктор або методсеттердляотриманняоб'єкту.Такожгарноюпрактикоюєвикористанняодногопідходуувсійпрограмізавиняткомдеяких ситуацій.

```
@Service
public class TourServiceImpl implements TourService {

    private TourRepository tourRepository;
    private TourTypeRepository tourTypeRepository;

    @Autowired
    public TourServiceImpl(TourRepository tourRepositor
        this.tourRepository = tourRepository;
        this.tourTypeRepository = tourTypeRepository;
    }
```

Рисунок4.11–Впровадженняоб'єктіврепозиторіївуклассервіс

ЗавдякиреалізаціїпідходуORMiODMможнастворитиспеціальніінтерфейси,якіматимутьдоступдокомунікаціїзісховищемданих(рис.4.12),такийпідхідзначноспрощуєвзаємодіюзбазоюізменшує кількість непотрібного коду за рахунок того, що інтерфейс якийнаслідується надаєвжебагатоіснуючихстандартнихзапитів. Занеобхідності в дочірньому інтерфейсі можна створити кастомні запити, слідуючи певнимправилам в іменуванні або написати запит вручну використовуючи анотацію@Query.

```java
@Repository
public interface TourTypeRepository extends JpaRepository<TourType, UUID> {
    Optional<TourType> findTourTypeByName(String name);
}
```

Рисунок4.12–Реалізаціярепозиторіютипівтурів

Спеціальнікласисервісивідповідаютьзабізнеслогіку.Внихвідбувається обробка даних, отриманих за рахунок запитів до контролерівклієнтом. Наприклад, мікросервіс турів, при отриманні запиту на збереженнянового туру, отримує інформацію через API, тобто контролер отримує запит,відбуваєтьсяперевіркавхіднихданихіпотімінформаціяпередаєтьсядоклас у TourServiceImpl. В якому, в свою чергу, реалізований метод зберіганняновоготуру(рис.4.13),алепередзбереженням,виконуєтьсяетапперевірки на рівні сховища а також етап конвертації даних пересилання, такзваних,DTOоб'єктів,умодельданихтуру.Якщовсіетапипроходятьуспішно тоновий тур зберігається усховищі.

```java
@Override
@Transactional
public TourDtoResponse create(TourDtoRequest tourDtoRequest, String guideId) {
    TourType type = checkTourTypeExistence(tourDtoRequest.getType());

    Tour tour = new Tour();
    tour.setType(type);
    tour.setGuideId(guideId);

    tour = tourRepository.save(TourMapper.tourDtoRequestToTour(tourDtoRequest, tour));
    return TourMapper.tourToTourDtoResponse(tour);
}
```

Рисунок4.13–Реалізаціяметодузберіганнятуру

Datatransferobjects – це патерн який використовується у даній системідля обмеження доступу і зменшення навантаження. Наприклад, при запитіінформаціїпрокористувача,приватнаінформація,така,якпарольненадсилається,щодозволяєуникнутирозкриттяперсональноїінформації.

Майже для кожного типу запиту існує свій DTO клас, який описує структуруоб'єкту(рис.4.14).

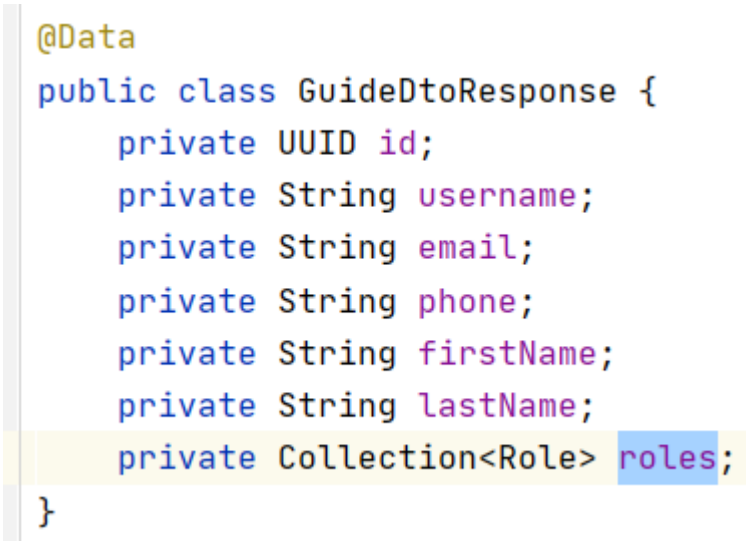



Рисунок 4.14 – Клас для опису відповіді при запиті інформації про гіда

Класиконтролеріє саметимиелементами,якідозволяють налаштуватиінтерфейссистемиіоброблюватизапити,щонадходятьвідпрограмклієнтів.БільшачастинаконфігурацієюAPIбазуєтьсянаналаштуваннях за допомогоюанотацій(табл. 4.4).

Таблиця4.4

Описанотаційкласівконтролерів

| Анотація | Опис |
|---|---|
| @RestController | Позначає,щокласєREST контролером. |
| @RequestMapping("/шлях") | Позначає,щозапити,якімаютьтакуклад овуURLповиннінаправлятися дозазначеногокласу. |
| @GetMapping, @PostMapping,@PutMapping,@ DeleteMapping | Позначає,якийвидHTTPзапитів можеоброблюватиметодкласу,атакож вказуєтьсящлях до нього. |
| @RequestBody | Позначаєтілозапиту,якенадходить відклієнтськогозастосунка. |
| @Valid | Цяанотація,дозволяєперевірититілоз апикузакритеріями,якщо вониналаштовані. |

## 4.6 Зовнішні інтерфейси

Важливу роль у системі відіграють класи контролери, за допомогоюякихстворюєтьсявебінтерфейсдоступудозастосунку.Впрограміреалізованірізніконтролеридлярізнихмоделей.Коженкласмаєвсобіспеціальнімето

дидлястворення,оновлення,отриманняівидаленняінформації. Відповідно, кожен метод контролеру це endpoint – URL по якомудоступнапевнафункціональність.Функціяотримуєконфігураціїзадопомогою анотацій, в яких налаштовується шлях до кінцевої точки і httpметод(рис. 4.15).

```
@PutMapping("/{tourId}")
public ResponseEntity<TourDtoResponse> updateTourById(@Validated @RequestBo
    return new ResponseEntity<>(tourService.update(tourDtoRequest, tourId),
}

@DeleteMapping("/{tourId}")
public ResponseEntity deleteTourById(@PathVariable UUID tourId) {
    tourService.delete(tourId);
    return new ResponseEntity(HttpStatus.OK);
}

/**
 *
 * @param guideId - ID of a guide.
 * @param pageNo - Page number.
 * @param itemsNumber - Amount of items per page.
 * @return List of tours created by a Guide.
 */
@GetMapping("/guide/{guideId}")
public ResponseEntity<Collection<TourDtoResponse>> getToursByGuideId(
        @PathVariable String guideId,
        @RequestParam(defaultValue = "0") int pageNo,
        @RequestParam(defaultValue = "10") int itemsNumber) {
```

Рисунок4.15–Деякіметодикласуконтролерабізнескористувачів

Створивши і налаштувавши контролери програма отримує певний API.Знаючиінформаціюналаштуваньінтерфейсубулостворенospеціальнуколекціюзадопомогоюпрограми Postman,якадозволяєнадсилатиhttpзапити (рис.4.16). Даний застосунок дуже зручно використовувати дляручного тестуванняприрозробціпродукту.

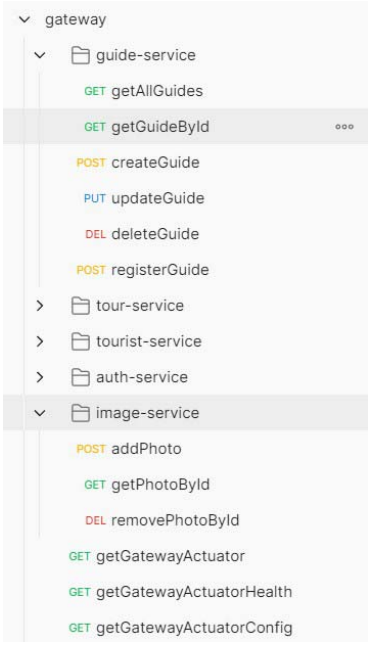



Рисунок4.16–ІнтерфейсимікросервісівупрограміPostman

## 4.7 Аутентифікація і авторизація

Майже всі сучасні системи потребують тієї чи іншої системи захисту іконтролю доступу до ресурсів. Даний застосунок не є виключенням. У данійсистемібулореалізованодварізнихтипикористувачів,асаметуристиігіди – бізнескористувачі.Головноювідмінністьцихдвохролейєдоступдопропозицій – їх створення, редагування і видалення. Ці два типи акаунтівповністюнезалежні,щоозначає,якщотуристзахочезапропонуватиексурсіютойому,буденеобхідноувійтиабостворитиновийбізнесакаунт.

На даний момент, у системі використовується підхід авторизації з JWT.Такий варіант доступу до ресурсів, дозволяє користуватися одним токеном іотримувати доступ до різних частин різних мікросервісів, що значно спрошуєрозробкузастосунку.Дляналаштуванняаутентифікаціїбувикористанийспеціальнийфреймворк SpringSecurity.ЗавдякичомуконфігуруваннядоступузводитьсядореалізаціїдекількохметодівнеберучидоувагиавторизаціюзадопомогоюJWT(рис. 4.17).

```
@Autowired
public CustomSecurityConfig(JwtFilter jwtFilter) { this.jwtFilter = jwtFilter; }

@Bean
public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {
    http.cors() CorsConfigurer<HttpSecurity>
            .and().httpBasic().disable() HttpSecurity
            .csrf().disable()
            .sessionManagement().sessionCreationPolicy(SessionCreationPolicy.STATELESS)
            .and() HttpSecurity
            .authorizeRequests() ExpressionUrlAuthorizationConfigurer<H>.ExpressionInterceptUrlRegistry
            .antMatchers( ...antPatterns: "/auth/admin").hasAuthority("ADMIN")
            .antMatchers( ...antPatterns: "/auth/**").permitAll()
            .anyRequest().authenticated()
            .and() HttpSecurity
            .authenticationManager(customersAuthenticationManager())
            .addFilterBefore(jwtFilter, UsernamePasswordAuthenticationFilter.class);
    return http.build();
}
```

Рисунок4.17–Налаштуваннябезпекидоступу

Для доступу до ресурсів, буларозроблена спеціальна структура JWTяка містить інформацію про власника даного токену (рис. 4.18). Токенміститьінформаціюпропоштовускринькуіролькористувача,термінприда тностітокену.Данаінформацієзберігаєтьсясерединітокенаієдоступною для всіх хто його має. Велику роль має «сіль»- підпис, якийзберігається у застосунку і за рахунок якого токен дуже складно підробити.Такимчином,базуючисьніпідписігенеруєтьсяпевнапослідовністьси мволів,щоі захищаєтокенвідредагуваньі зовнішньоговпливу.

```
@Component
public class JwtProvider {

    @Value("${jwt.secret}")
    private String secret;

    @Value("${jwt.type}")
    private String type;

    public String generateAccessToken(String email, Collection<String> roles) {
        return Jwts.builder()
                .setHeaderParam( name: "typ", type)
                .claim( name: "roles", roles.toArray())
                .setSubject(email)
                .setExpiration(
                        Date.from(LocalDateTime.now()
                                .plus(Duration.of( amount: 30000, ChronoUnit.MINUTES))
                                .atZone(ZoneId.systemDefault()).toInstant())
                )
                .signWith(SignatureAlgorithm.HS512, secret)
                .compact();
    }
```

Рисунок4.18–РеалізаціялогікистворенняJWT

При виконанні запиту на вхід і успішному його обробленні, користувачотримуєтокендоступу(рис.4.19).Який,уподальшомубудевикористов уватидля того,щоботримати дані зсистеми.

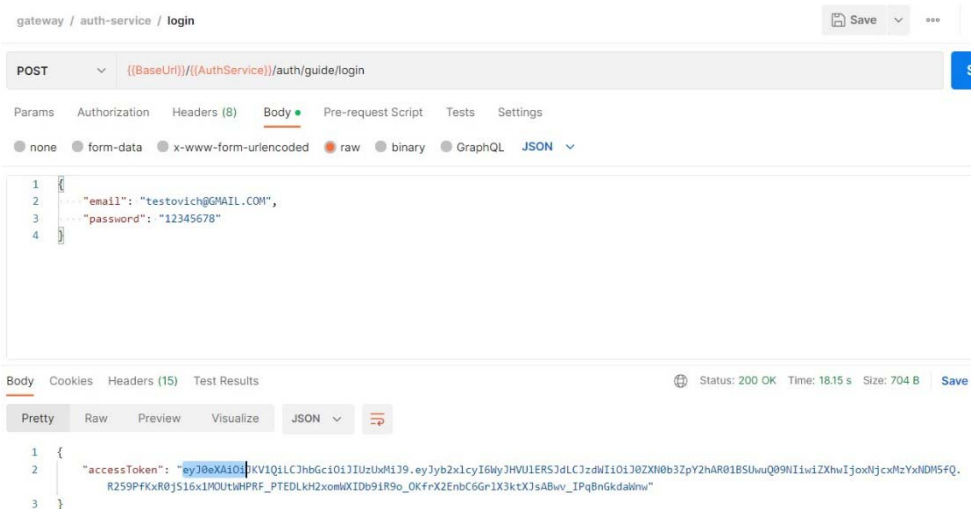



Рисунок4.19–Тестуваннязапитунавхід,черезпрограму Postman

Невсіданімаютьбудизахищені.Наприкладколиз'являєтьсяновийкористувач застосунку – турист, то він не мусить мати акаунт для того, щобпереглянути наявні пропозиції, але якщо він захоче забронювати тур то йомубуденеобхідностворити акаунт.

Створенняакаунту,щеодинважливийетап,користувачмаєобов’язково вказати свою пошту і придумати пароль, а також надати ім’я іпрізвище(рис.4.20),ціданієкритичноважливимидляподальшогокористуваннясис темою.

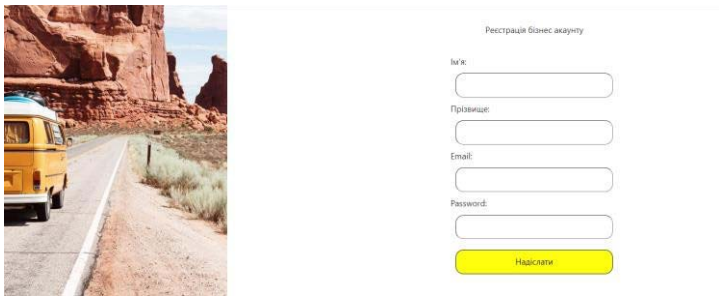



Рисунок4.20–формаствореннябізнесакаунту

Взаємодія і реалізація всіх мікросервісів є доволі складною і міститьбагаторізнихелементіконфігурацій,класівіінтерфейсів,яківвсюючергу реалізуютьпевнібізнеспроцесиіописуютьповедінкурізнихелементів(ДодатокА.Ч астинапрограмногокодумікросервісів).

## 4.8 Клієнтський застосунок

Клієнтбувпобудований,якодносторінковийзастосунокзвикористанням платформи Angular. Даний підхід дозволяє використовуватиготові модулі і компоненти багаторазово, що пришвидшує розробку сайтівтакого типу.

Застосунокскладаєтьсяздвохосновнихчастин,асамеінтерфейсутуристіві бізнес користувачів. При першому потраплянні на сайт,клієнтбачитьголовнусторінкунаякійможнашукатитуризарізнимипараметрами (рис. 4.21).

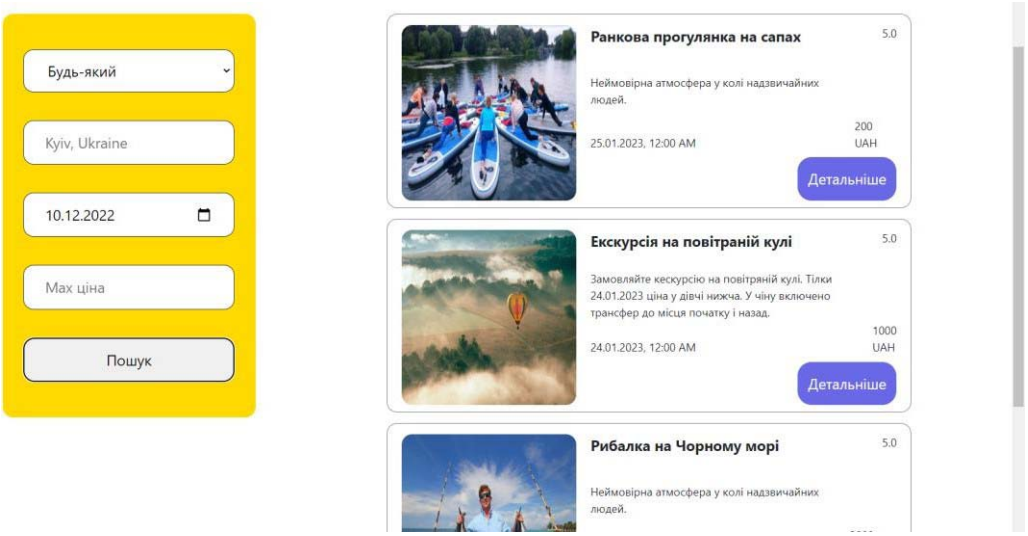



Рисунок4.21–Головнасторінказформоюпошукутурів

Якщокористувачзацікавленийуствореннівласнихпропозицій,тойомунеобхідно,пройтиаутентифікаціюубізнесакант(рис. 4.22).

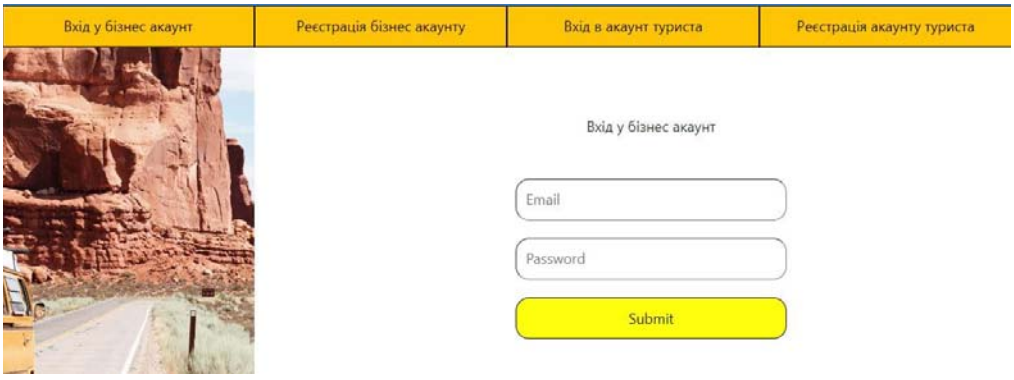



Рисунок4.22–Сторінкавходувбізнесакаунт

Після успішного входу в систему, користувачу буде наданий доступ доінтерфейсугідів,дебізнесклієнтзможестворюватиновіпропозиціїіредагуватиісну

ючі (рис.4.23).

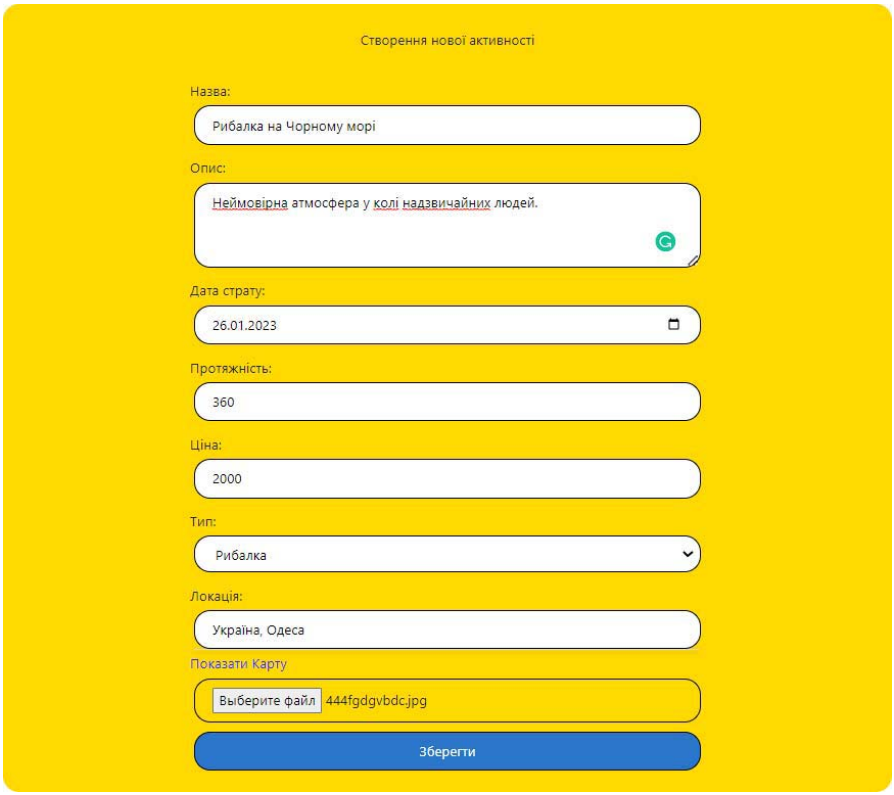



Рисунок4.23–Формастворенняіредагуванняактивності

Длястворенняновоїактивностігідунеобхіднозаповнитиформуізберегтиїї.Форма створенняпропозиціїміститьтакіполя:

– Назва.

– Опис.

– Датастарту.

– Протяжністьухвилинах.

– Цінаугривнях.

– Тип.

– Локація.

– Зображеннядлятуру.

Вподальшому,данаформаможебутирозширенаіоптимізованавідповіднодоп отребклієнтів.

Веб-застосунокскладаєтьсязмодулів,якіреалізуютьрізніфункціональностісайту(рис.4. 24).Розділеннялогікизамодулямидозволяєкращеорієнтуватисявіснуючомукодіїза

вантажуватитількинеобхідні функціїпевногомодулю.

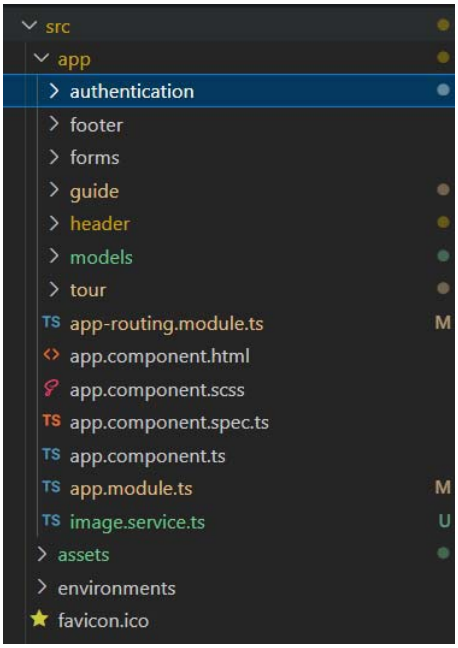



Рисунок4.24–Файловаструктуравеб-застосунку

Кожен модуль містить в собі унікальну функціональність і включаєнеобхідні бібліотеки для подальшого використання в компонентах і сервісах,які в свою чергу реалізують всю логіку взаємодії з користувачем і поведінкузастосунку в цілому, маючи певну реалізацію бізнес-логіки (Додаток Б.Частинапрограмногокоду веб-клієнта).

## Розділ 5.Розроблення стартап-проекту

### 5.1Опис ідеї стартапу

В даний час власна ідея створення веб-платформи для популяризації активного відпочинку виглядає досить досяжною. Однак розробка дійсно ефективного продукту, який буде приваблювати користувачів і дійсно сприятиме популяризації активного відпочинку, є завданням, що вимагає великих зусиль. Важливо добре розуміти, з якого етапу почати, які дії виконати, і мати на увазі, що досягнення успіху вимагає часу та терпіння, а також не зупинятися після можливих невдалих спроб.

Основним етапом перед тим, як анонсувати проект як новий стартап, є виконання ряду конкретних дій. Ключовим кроком є розробка інформаційної картки проекту, що містить усю необхідну та коротку інформацію.Нижче опишу короткі відомості про проект.

1.Назва номінації - Webapplication (Інтернет-застосунок)

2.Назва проекту – Веб платформи для популяризації активного відпочинку

3.Назва ВНЗ, факультету, спеціальності -НЛТУ, кафедра інформаційних технологій, комп’ютерні науки

4.Автор – студент

5.Цілі проекту:

- Підвищення рівня фізичної активності населення.
- Зменшення стресових ситуацій та покращення психічного благополуччя.
- Залучення тимчасово переміщених осіб та людей пенсійного віку.

Задачі проекту:

Розробка функціоналу веб-платформи (створення системи фільтрації та пошуку заходів за типами активностей, регіонами, часом тощо).Розробка функціоналу для взаємодії користувачів, обміну враженнями та рецензіями.

Адаптація до потреб цільових аудиторій (врахування особливостей тимчасово переміщених осіб та людей пенсійного віку при розробці та рекламі платформи).

Партнерство з організаторами подій та підприємствами (встановлення партнерських відносин для розміщення активних заходів та пропозицій на платформі).

Забезпечення зручного інтерфейсу (розробка інтуїтивно зрозумілого та доступного інтерфейсу для користувачів з різним рівнем технічної грамотності).

Просування та маркетинг (розробка стратегії маркетингу та реклами для привертання уваги цільової аудиторії, залучення користувачів через різні медіа-канали та соціальні мережі).

Оцінка ефективності та збір зворотного зв'язку (запровадження системи аналітики для вимірювання активності користувачів та ефективності платформи та збір та аналіз зворотного зв'язку для подальших вдосконалень).

Технічна підтримка та оновлення (забезпечення надійної технічної підтримки та регулярних оновлень платформи).

Ці цілі та задачі становлять основну стратегічну рамку для успішного впровадження та функціонування проекту з популяризації активного відпочинку через веб-платформу.

6.Короткий зміст - проект передбачає створення веб-платформи для популяризації активного відпочинку у сучасному суспільстві, яке стикається з проблемами завантаженості, стресів та фізичної неактивності. Зокрема, ідеться про розробку системи, яка спростить пошук та участь у різноманітних активних заходах для різних груп населення, включаючи тимчасово переміщених осіб та людей пенсійного віку. Проект спрямований на покращення фізичного та психічного здоров'я, зменшення стресів та розвиток галузі послуг, спрямованих на активний відпочинок. Метою є розробка зручного та функціонального інструменту, який стане платформою для знаходження та обміну активними відпочинковими можливостями.

7.Термін виконання проекту –6 місяців

## 5.2Розроблення ринкової стратегії

В цьому пункті сформовано декілька стратегій, як правильно використовувати не лише програмне забезпечення, а й особистісні підходи до рекламодавців, клієнтів та, власне, цільову аудиторію, для якої і подається реклама.

Першим кроком для успішної реалізації проекту – це створення впізнаваного бренду. Немає значення – це звичайна маленька мережа чи велика платформа – такий сайт повинен стати популярним. Спершу потрібно інвестувати в рекламу мережі. Подбати про його хороші риси, які спершу, звісно, виробити. Ці риси подавати ненав'язливо, адже люди не люблять одноманітності та нав'язування. Потрібно про свої перспективні сторони бізнесу натякати, щоб клієнт сам зробив відповідні висновки. Розповісти про переваги чому слід користуватись саме моєю мережею, що в ній особливого і чим вона краща за інші. Така методика вплине на користувачів набагато ефективніше, адже так тоді він гадатиме, що сам дійшов до такого рішення.

Після того, як про ваш бренд дізнаються, слід подбати про дизайн, перше враження користувача так би мовити. Людям необхідно відчувати себе в зоні комфорту – тоді вони повертатимуться знову. Відвідавши один раз сайт людина зразу зможе зробити висновок, тому дуже важливо, щоб перше враження було позитивне, щоб користування було зручним і зрозумілим і головне, щоб все виглядало пристойно. Початкова загрузка сайту не повинна бути повільною, користувач не хоче довго чекати і є ймовірність, що він просто закриє вкладу. Також такі речі як реєстрація, створення профілю і тому подібне не повинно забирати в людей багато часу, все має швидко і зрозуміло, це допоможе зберегти більше людей і залучити нових.

Зацікавивши людей відвідувати мережу знову і знову можна переходити до наступного етапу – залучення рекламодавців. Як правило, при стрімкому розвитку бізнесу рекламодавці самі починають виходити на зв'язок, аби рекламувати свої

продукти використовуючи вашу платформу. Важливо пам’ятати, що все має свої обмеження, особливо реклама.

Мало хто з користувачів любить рекламу, тому вони найменше хочуть її бачити. Тому не варто зловживати. Не завжди потрібно показувати рекламу повністю. Звісно, сама реклама – це власна справа рекламодавця, але спробуйте вибрати більш цікаві варіанти. Іншими словами, слухаючи рекламу, у вас самого має виникати бажання стати клієнтом бізнесу рекламодавця. Важливо, щоб реклама не була одноманітною, адже вона швидко набридне, викличе лише дискомфорт і злість людей, що лише відразу призведе до продукту рекламодавця, що суперечить меті апріорної реклами. Останній етап – це аналіз, який насправді є статистикою. Ця статистика важлива не тільки для рекламодавців, а й для вас. Він показує, які рекламодавці не гідні розміщувати свої оголошення у вашій організації, а інші, навпаки – потрібно заохочувати, а при правильній аргументації ще й отримувати додаткові бонуси за успішність розміщення реклами продукту у вашому закладі.

Отже, дотримуючись усіх вищенаведених стратегій ведення бізнесу, вибірки рекламодавців і реклами,також поступове виконання необхідних кроків дозволить успішно отримувати дохід з реклами, що буде сприяти появі нових користувачів

### 5.3 Вимоги до технічного та програмного забезпечення

Длявідображеннятакористуванняданою веб платформою достатньо зайти у будь-який браузер(GoogleChrome, Mozilla, Safari, etc.).

Від швидкості інтернету залежить швидкість самої мережі і здійснення запитів на бек-енд і отримання відповідей на клієнтській стороні. Тому для стабільної і безперебійної роботи, слід мати хороше підключення до інтернету.

Крім цього користуватись веб платформою для популяризації активного відпочинкуможна навіть і з телефону, оскільки гнучкий дизайн дає змогу переглядати мережу на різних пристроях(мінімальна підтримка для пристроїв з екраном 360px).

# Висновки

Створення інтернет-платформи для популяризації активного відпочинку є необхідним у зв'язку зі зростаючою кількістю людей, які цікавляться активним проведенням вільного часу та великою кількістю різноманітних активних відпочинкових об'єктів (ВПО). Інтернет-платформа моженадатизручний та доступнийінструмент для пошуку та організаціїактивнихвідпочинковихзаходів, сприятивзаємодіїміжкористувачами, а такожпопуляризуватиактивнівидидозвіллясередширокоїаудиторії.

Для зберігання і управліннняданими про користувачів, активнівідпочинковіоб'єкти та іншіпотрібнідані, база даних є важливим компонентом. Використаннябазиданих у проекті дозволить зберігати, оновлювати та отримуватинеобхіднуінформаціюефективним способом.

Просуванняпопулярностіплатформиможназдійснити за допомогою SMM-стратегії (стратегіїсоціальнихмедіа). Вона дозволяєзалучатиувагу та залучатиаудиторію через соціальнімережі, створюватицікавий та віральний контент, проводитиінтерактивніакції та спілкуватися з користувачами. Використання SMM-стратегіїможезабезпечитишвидкий результат у популяризаціїплатформисередцільовоїаудиторії.

На базі розробленої технології спроєктовано і програмно реалізовано застосунок направлений на популяризацію і підтримку активного відпочинку, а також розвиток малого і середнього бізнесу. Вподальшому,данийпроектможебутилегкорозширенийіншимимікросервісами і функціональностями, які будуть задовольняти виникаючіпотребикористувачів.

## Список літератури


1. Сайт Міністерствасоціальноїполітикиhttps://www.msp.gov.ua/timeline/Vnutrishno-peremishcheni-osobi.html

2. “Велика УкраїнськаЕнциклопедія” https://www.msp.gov.ua/timeline/Vnutrishno-peremishcheni-osobi.html

3. “Укрінформ” https://www.ukrinform.ua/rubric-technology/2797152-v-ukraini-kilkist-internetkoristuvaciv-zrosla-do-23-miljoniv.html

4. Schaffner, A. (2019) Social Media Marketing Workbook 2019: How to Leverage The Power of Facebook

5. Advertising, Instagram Marketing, YouTube and SEO To Explode Your Business and Personal Brand. CreateSpace Independent Publishing Platform.Basyuk, T. (2018). The Popularization Problem of Websites and Analysis of Competitors. In: Shakhovska N.,Stepashko V. (eds). Advances in Intelligent Systems and Computing II. CSIT 2017. Advances in Intelligent Systems and Computing, vol 689. Springer, Cham pp. 54–65.

6. Odden, L. (2012) Optimize: How to Attract and Engage More Customers by Integrating SEO, Social Media, and Content Marketing. USA:Wiley.

7. Lipschultz, J. (2019) Social Media Measurement and Management. UK: Routledge.

8. Gemius Украина (2020). Возраст пользователей социальных медиа. Отримано з: http://www.gemius.com.ua/vse-stati-dlja-chtenija/vozrast-polzovatelej-socialnyx-media.html.

9. McDonald, J. (2016) Social Media Marketing Workbook: How to Use Social Media for Business.Workbook edition. CreateSpace Independent Publishing Platform.

10. Odabasi, K. (2019) Digital Marketing Strategies: Ultimate Guide to SEO, Google Ads, Facebook & Instagram Ads, Social Media, Email Newsletters. CreateSpace Independent Publishing Platform.

11. Басюк, Т., Василюк, А. (2016). Пошуковепросуваннякомерційнихінтернет-ресурсів. ВісникНаціональногоуніверситету “Львівськаполітехніка”. Серія: “Інформаційнісистеми та мережі”, № 887. 3–9.

12. Берко, А., Висоцька, В., Чирун, Л. (2015). Лінгвістичнийаналіз текстового комерційного контенту. ВісникНаціональногоуніверситету “Львівськаполітехніка”. Серія: “Інформаційнісистеми та мережі”, № 814, 203–228.

13. Буров, Є., Завущак, І. (2017). Методиопрацювання контексту в інтелектуальних системах. ВісникНаціональногоуніверситету “Львівськаполітехніка”. Серія: “Інформаційнісистеми та мережі”, № 872, 121–131. Популяризація комерційних інтернет-ресурсів із використанням соціальних медіа 19

14. Basyuk, T., Vasilyuk, A., Lytvyn, V. (2019). Mathematical model of semantic search and search optimization // CEUR Workshop Proceedings. – Vol. 2362 : Proceedings of the 3rd International conference on computational linguistics and intelligent systems, COLINS-2019, Kharkiv, Ukraine. pp. 96–105.

15. Davis, H. (2016). Google Advertising Tools: Cashing in with AdSense, AdWords, and the Google APIs. USA: O'Reilly Media.

16. Basyuk, T. (2018). Popularization of Internet resources by using ”featured snippets”. Proceedings of the 20-th International conference SAIT 2018. Kyiv. 190–191.

17. Komaromi, K. (2003). Building brand communities. – Отримано з: http://cdgroup.blogs.com/design_channel/brand_communities.pdf.

18. PHP – Вікіпедія [Електронний ресурс]. –Отримано з: https://uk.wikipedia.org/wiki/PHP

19. Port of node-ar-drone which allows user to control a Parrot AR Drone over PHP [Електроннийресурс]. –Отримано з: https://github.com/jolicode/php-ar-drone

20. Мультипарадигмальна мова програмування – Вікіпедія [Електронний ресурс]. – Отримано з: https://uk.wikipedia.org/wiki/Мультипарадигмальна_мова_програмуван ня

21. JavaScript | MDN [Електронний ресурс]. –Отримано з: https://developer.mozilla.org/ru/docs/Web/JavaScript

22. JavaScrit – Вікіпедія [Електронний ресурс]. –Отримано з: https://uk.wikipedia.org/wiki/JavaScript

23. AJAX – Вікіпедія [Електронний ресурс]. –Отримано з: https://uk.wikipedia.org/wiki/AJAX

24. РушійJavaScrit – Вікіпедія [Електронний ресурс]. –Отримано з: https://uk.wikipedia.org/wiki/Рушій_JavaScript

25. Ruby – Вікіпедія [Електронний ресурс]. –Отримано з: https://uk.wikipedia.org/wiki/Ruby

26. Python – Вікіпедія [Електронний ресурс]. –Отримано з: https://uk.wikipedia.org/wiki/Python

# Додаток А.Частинапрограмногокодумікросервісів

## Guide-Service

```
@RestController@RequestMapping("
/guide")publicclassGuideControll
er{

privateGuideServiceguideService;@Autowir

ed
publicGuideController(GuideServiceguideService)
{this.guideService=guideService;
}

@GetMapping("/all")
publicResponseEntity<Collection<GuideDtoResponse>>getAll(){
returnnewResponseEntity<>(guideService.getAll(),HttpStatus.OK);
}

@GetMapping("/{id}")
publicResponseEntity<GuideDtoResponse>getByGuideId(@PathVariable UUIDid){
returnnewResponseEntity<>(guideService.getById(id),HttpStatus.OK);
}

@GetMapping("email/{email}")
publicResponseEntity<GuideDtoResponse>getByEmail(@PathVariable Stringemail){
returnnewResponseEntity<>(guideService.getGuideByEmail(email),HttpStatus.O
K);
}

@PostMapping
publicResponseEntity<GuideDtoResponse>create(@RequestBodyGuideDto
RequestguideDtoRequest) {
returnnewResponseEntity<>(guideService.create(guideDtoRequest),HttpStatus.C
REATED);
}

@PutMapping("/{id}")
publicResponseEntity<GuideDtoResponse>update(@RequestBodyGuideDto
RequestguideDtoRequest,@PathVariableUUIDid){
returnnewResponseEntity<>(guideService.update(guideDtoRequest,
id),HttpStatus.CREATED);
}

@DeleteMapping("/{id}")
publicResponseEntitydelete(@PathVariable UUID id)
{guideService.delete(id);
returnnewResponseEntity(HttpStatus.OK);
}

@PostMapping("/reg")
publicResponseEntity<GuideRegDtoResponse>registerGuide(@RequestBodyGuideReg
DtoRequestregisterGuide) {
returnnewResponseEntity<>(guideService.registerGuide(registerGuide),
```

```
HttpStatus.CREATED);
}
}

@Repository
publicinterfaceGuideRepositoryextendsJpaRepository<Guide, UUID>
{Optional<Guide>findByEmail(Stringemail);
}

publicinterfaceGuideService{
GuideDtoResponsecreate(GuideDtoRequestguideDtoRequest);GuideDtoResponseupdate

(GuideDtoRequestguideDtoRequest, UUID guideId);voiddelete(UUID guideId);

Collection<GuideDtoResponse>getAll();GuideDtoResponsegetB

yId(UUIDguideId);

InnerLoginGuideDtoResponsegetGuideByEmailInner(StringguideEmail);GuideDto
ResponsegetGuideByEmail(Stringemail);

GuideRegDtoResponseregisterGuide(GuideRegDtoRequestregisterGuide);
}

@Service
publicclassGuideServiceImplimplementsGuideService{

privateGuideRepositoryguideRepository;privateR
oleRepositoryroleRepository;privatePasswordEnc
oderencoder;

@Autowired
publicGuideServiceImpl(GuideRepositoryguideRepository,
RoleRepositoryroleRepository, PasswordEncoderencoder) {
this.guideRepository=
guideRepository;this.roleRepository=
roleRepository;this.encoder= encoder;
}

@Override
publicGuideDtoResponsecreate(GuideDtoRequestguideDtoRequest)
{Guideguide=
guideRepository.save(GuideMapper.guideDtoRequestToGuide(guideDtoRequest,
newGuide()));
returnGuideMapper.guideToGuideDtoResponse(guide);
}

@Override
publicGuideDtoResponseupdate(GuideDtoRequestguideDtoRequest, UUIDguideId){
Optional<Guide>optionalGuide=
guideRepository.findById(guideId);if(optionalGuide.isEmpty()) {
thrownewRuntimeException("Guidenotfound.");
}
Guideguide=
GuideMapper.guideDtoRequestToGuide(guideDtoRequest,optionalGuide.get());
returnGuideMapper.guideToGuideDtoResponse(guideRepository.save(guide));
}
```

```
@Override
publicvoiddelete(UUIDguideId){
Guideguide= guideRepository.getById(guideId);if(guide==
null) {
thrownewRuntimeException("Guidenotfound.");
}
guideRepository.deleteById(guideId);
}

@Override
publicCollection<GuideDtoResponse>getAll() {return
GuideMapper.guidesToGuideDtosResponse(guideRepository.findAll());
}

@Override
publicGuideDtoResponsegetById(UUID guideId) {return
GuideMapper.guideToGuideDtoResponse(guideRepository.getReferenceById(guideId)
);
}

@Override
publicInnerLoginGuideDtoResponsegetGuideByEmailInner(StringguideEmail)
{
varguide=getByEmail(guideEmail);
varinnerGuide=
newInnerLoginGuideDtoResponse(guide.getEmail(),
guide.getPassword(),guide.getRoles().stream().map(r
->
r.getRole().name()).collect(Collectors.toList())
);
returninnerGuide;
}

@Override
publicGuideDtoResponsegetGuideByEmail(Stringemail){
returnGuideMapper.guideToGuideDtoResponse(getByEmail(email));
}

privateGuidegetByEmail(Stringemail){
varguideOptional=
guideRepository.findByEmail(email);if(guideOptional.isEmpty()) {
thrownewRuntimeException("Guidenotfound.");
}
returnguideOptional.get();
}

@Override
publicGuideRegDtoResponseregisterGuide(GuideRegDtoRequestguideReg
DtoRequest){
varguideOptional=guideRepository.findByEmail(guideRegDtoRequest.getEmail());
if(guideOptional.isPresent()){
returnnewGuideRegDtoResponse(RegStatus.FAILED,"Emailalready
exists.");
}

varnewGuide=

newGuide();newGuide.setPassword(encoder.encode(guideRegDtoRequest.getPassword()));
newGuide.setEmail(guideRegDtoRequest.getEmail());
```

```
newGuide.setFirstName(guideRegDtoRequest.getFirstName());newGuide.setLastName(guideR
egDtoRequest.getLastName());newGuide.setAccountState(AccountState.NEW);

varroleOptional=
roleRepository.findRoleByRole(UserRole.GUIDE);Rolerole=roleOptional.isEmpty
()?roleRepository.save(new
Role(UserRole.GUIDE)) :
roleOptional.get();newGuide.setRoles(Arrays.asList(role));

guideRepository.save(newGuide);
returnnewGuideRegDtoResponse(RegStatus.NEW,"Success.");
}
}

@Data@Ent
ity
@Table(name =
"guides")publicclassGuide
{

@Id
@GeneratedValue(generator =
"uuid2")@GenericGenerator(name="uuid2",strategy="uuid2")
@Column(name = "guide_id", updatable = false, nullable =
false,columnDefinition="VARCHAR(36)")
@Type(type = "uuid-
char")privateUUID id;

@Column(name = "username", unique = true/*, nullable =
false*/)privateStringusername;

@Column(name = "email", unique = true, nullable =
false)privateStringemail;

@Column(name = "phone", unique =
true)privateStringphone;

@Column(name =
"first_name")privateStringfirstNam
e;

@Column(name =
"last_name")privateStringlastName
;

@CreationTimestamp
@Column(name = "created_at", updatable = false, nullable =
false)privateTimestampcreatedAt;

@UpdateTimestamp@Column(name =
"updated_at")privateTimestampupda
tedAt;

@Column(name = "password", nullable =
false)privateStringpassword;

@ManyToMany(fetch =
FetchType.EAGER)@JoinTable(
name="user_role",
joinColumns = @JoinColumn(name =
"user_id"),inverseJoinColumns=@JoinColumn(name="role_id")
)
privateCollection<Role>roles;@Enumerated(value

=EnumType.STRING)
```

```
@Column(name = "account_state", nullable =
false)privateAccountStateaccountState;
}


@Data
publicclassGuideDtoRequest{

@NotBlank
@Size(min = 6, max =
40)privateStringusername;

@NotBlank@Email
privateStringemail;

@NotBlank
privateStringpassword;

@NotBlank
@Size(min = 7, max =
20)privateStringphone;

@NotBlank
@Size(min = 1, max =
40)privateStringfirstName;

@NotBlank
@Size(min = 1, max =
40)privateStringlastName;
}

@Data
publicclassGuideDtoResponse{priva
teUUID id;
privateStringusername;privateSt
ringemail;privateStringphone;pr
ivateStringfirstName;privateStr
inglastName;
privateCollection<Role>roles;
}

@Data
publicclassGuideRegDtoRequest{

@Email
privateStringemail;

@NotEmpty@Min(8
)
privateStringpassword;

@NotEmpty
privateStringfirstName;

@NotEmpty
privateStringlastName;
}

publicclassGuideMapper{
```

```
publicstaticGuideDtoResponseguideToGuideDtoResponse(Guideguide)
{GuideDtoResponseguideDtoResponse=
newGuideDtoResponse();guideDtoResponse.setId(guide.getId());guideDtoRespons
e.setEmail(guide.getEmail());guideDtoResponse.setFirstName(guide.getFirstNa
me());guideDtoResponse.setLastName(guide.getLastName());guideDtoResponse.se
tPhone(guide.getPhone());guideDtoResponse.setUsername(guide.getUsername());
guideDtoResponse.setRoles(guide.getRoles());
returnguideDtoResponse;
}

publicstaticCollection<GuideDtoResponse>guidesToGuideDtosResponse(Collection<Guid
e>guides){
returnguides.stream().map(guide->
guideToGuideDtoResponse(guide)).collect(Collectors.toList());
}

publicstaticGuideguideDtoRequestToGuide(GuideDtoRequestguideDtoR
equest, Guideguide){
guide.setEmail(guideDtoRequest.getEmail());guide.setPhone(guideDtoR
equest.getPhone());guide.setFirstName(guideDtoRequest.getFirstName());
guide.setLastName(guideDtoRequest.getLastName());guide.setUsername(
guideDtoRequest.getUsername());guide.setPassword(guideDtoRequest.ge
tPassword());guide.setRoles(newHashSet<Role>(Arrays.asList(new
Role(UserRole.GUIDE))));
returnguide;
}
}

@SpringBootApplication@E
nableEurekaClient@Enable
FeignClients
publicclassGuideServiceApplication{

publicstaticvoidmain(String[] args)
{SpringApplication.run(GuideServiceApplication.class,args);
}
}
```

## Image-Service

```
@RestController@RequestMapping(
"/photo")publicclassPhotoContro
ller{

privatePhotoServicephotoService;

publicPhotoController(PhotoServicephotoService)
{this.photoService=photoService;
}

@PostMapping
publicResponseEntity<?>addPhoto(@RequestParam("image")
MultipartFileimage)throwsIOException{
returnnewResponseEntity<>(photoService.addPhoto(image),HttpStatus.O
K);
}
```

```
@DeleteMapping(value="/{photoId}")
publicResponseEntityremoveById(@PathVariable StringphotoId)
{photoService.removeById(photoId);
returnnewResponseEntity(HttpStatus.OK);
}

@GetMapping(value = "/{photoId}", produces =
{MediaType.IMAGE_JPEG_VALUE,MediaType.IMAGE_PNG_VALUE})
@ResponseBody
publicResponseEntity<ByteArrayResource>getPhoto(@PathVariable
StringphotoId){
PhotoloadFile=
photoService.getPhoto(photoId);loadFile.getFile().getData(
);

returnResponseEntity.ok()
.contentType(MediaType.parseMediaType(loadFile.getFileType()
))
.header(HttpHeaders.CONTENT_DISPOSITION,"attachment;
filename=\""+loadFile.getFilename()+"\"")
.body(newByteArrayResource(loadFile.getFile().getData()));
}
}

@Data
@Document(collection =
"photos")publicclassPhoto{
@Id
privateStringid;

privateStringfilename;privateStrin
gfileType;privatelongfileSize;priv
ateStringoriginalName;privateBinar
yfile;
}

@Data@NoArgsConstruct
or@AllArgsConstructor
publicclassImgDtoResponse{priva
teStringid;
}

@Repository
publicinterfacePhotoRepositoryextendsMongoRepository<Photo,String>{}

@Service
publicclassPhotoService{

privatePhotoRepositoryphotoRepository;@Autowir

ed
publicPhotoService(PhotoRepositoryphotoRepository)
{this.photoRepository=photoRepository;
}

publicImgDtoResponseaddPhoto(MultipartFilefile) throwsIOException{Photophoto
= newPhoto();

photo.setFilename(file.getName());photo.setOriginalName(file.getOriginalFilename());
```

```
photo.setFileSize(file.getSize());photo.setFileType(file.getContentType());
photo.setFile(newBinary(BsonBinarySubType.BINARY, file.getBytes()));photo=

photoRepository.insert(photo);

returnnewImgDtoResponse(photo.getId());
}

publicPhotogetPhoto(Stringid){
varphotoOptional=
photoRepository.findById(id);if(photoOptional.isEmpty()){
thrownewRuntimeException("Photonotfound.");
}
returnphotoRepository.findById(id).get();
}

publicvoidremoveById(StringphotoId)
{photoRepository.deleteById(photoId);
}
}

@SpringBootApplication@E
nableEurekaClient
publicclassImageServiceApplication{

publicstaticvoidmain(String[] args)
{SpringApplication.run(ImageServiceApplication.class,args);
}

}
```

**Auth-Service**

```
@Configuration@Enabl
eWebSecurity
publicclassCustomSecurityConfig{priva

teJwtFilterjwtFilter;

@Autowired
publicCustomSecurityConfig(JwtFilterjwtFilter)
{this.jwtFilter=jwtFilter;
}

@Bean
publicSecurityFilterChainfilterChain(HttpSecurityhttp)
throwsException{
http.cors()
.and().httpBasic().disable()
.csrf().disable()

.sessionManagement().sessionCreationPolicy(SessionCreationPolicy.STATELESS)
.and()
.authorizeRequests()
.antMatchers("/auth/admin").hasAuthority("ADMIN")
.antMatchers("/auth/**").permitAll()
.anyRequest().authenticated()
.and()
.authenticationManager(customersAuthenticationManager())
.addFilterBefore(jwtFilter,
```

```
UsernamePasswordAuthenticationFilter.class);retu
rnhttp.build();
}

@Bean
publicGuideDetailsServiceguideDetailsService()
{returnnewGuideDetailsService();
}

@Bean
publicTouristDetailsServicetouristDetailsService()
{returnnewTouristDetailsService();
}

@Bean
publicAuthenticationManagercustomersAuthenticationManager()
{returnauthentication -> {
varroles=
authentication.getAuthorities();if(roles.stream().anyMatch
(r ->
r.toString().equals(Role.TOURIST.name()))){
varuser=touristDetailsService().loadUserByUsername(authentication.getName());
if(user!=
null&&encoder().matches((CharSequence)authentication.getCredentials(),
user.getPassword())){
returnnewUsernamePasswordAuthenticationToken(authentication.getName(),authentication
.getAuthorities());
}
} elseif(roles.stream().anyMatch(r -
>r.toString().equals(Role.GUIDE.name()))){
varuser=guideDetailsService().loadUserByUsername(authentication.getName());
if(user!=
null&&encoder().matches((CharSequence)authentication.getCredentials(),
user.getPassword())){
returnnewUsernamePasswordAuthenticationToken(authentication.getName(),authentication
.getAuthorities());
}
}
thrownewUsernameNotFoundException(authentication.getName());
};
}

@Bean
publicPasswordEncoderencoder()
{returnnewBCryptPasswordEncoder();
}
}

publicclassCustomUserDetailsimplementsUserDetails{

privateStringemail;privateStr
ingpassword;
privateCollection<?extendsGrantedAuthority>grantedAuthorities;

@Override
publicStringgetUsername()
{returnemail;
}

@Override
publicStringgetPassword(){
```

```
returnpassword;
}

@Override
publicCollection<? extendsGrantedAuthority>getAuthorities()
{returngrantedAuthorities;
}

@Override
publicbooleanisAccountNonExpired()
{returntrue;
}

@Override
publicbooleanisAccountNonLocked()
{returntrue;
}

@Override
publicbooleanisCredentialsNonExpired()
{returntrue;
}

@Override
publicbooleanisEnabled()
{returntrue;
}


publicstaticCustomUserDetailsFromGuideToCustomUserDetails(AuthUserauthUser)
{
varuser= newCustomUserDetails();user.email=
authUser.getEmail();user.password=authUser.getPas
sword();
if(authUser.getRoles() == null|| authUser.getRoles().isEmpty())
{thrownewRuntimeException("Authuserdoesnothaveanyrole!");
}
user.grantedAuthorities=authUser.getRoles()
.stream()
.map(r->newSimpleGrantedAuthority(r))
.collect(Collectors.toList());returnuser;
}

publicstaticCustomUserDetailsFromTouristToCustomUserDetails(AuthTou
ristauthUser){
varuser= newCustomUserDetails();user.email=
authUser.getEmail();user.password=authUser.getPas
sword();
if(authUser.getRole() == null|| authUser.getRole().isEmpty())
{thrownewRuntimeException("Authuserdoesnothaveanyrole!");
}
user.grantedAuthorities=
Arrays.asList(newSimpleGrantedAuthority(authUser.getR
ole()));
returnuser;
}
}

publicclassGuideDetailsServiceimplementsUserDetailsService{

@Autowired
privateGuideClientguideClient;
```

```
@Override
publicUserDetailsloadUserByUsername(Stringemail)
throwsUsernameNotFoundException{
varauthUser=
guideClient.getInnerGuideByGuideEmail(email);returnCustomUserDetails.FromGu
ideToCustomUserDetails(authUser);
}
}

publicclassTouristDetailsServiceimplementsUserDetailsService{@Autowired
privateTouristClienttouristClient;

@Override
publicUserDetailsloadUserByUsername(Stringemail)
throwsUsernameNotFoundException{
varauthUser=touristClient.getInnerTouristByEmail(email);
returnCustomUserDetails.FromTouristToCustomUserDetails(authUser);
}
}

@Component
publicclassJwtFilterextendsGenericFilterBean{

privateJwtProviderjwtProvider;
privateGuideDetailsServicecustomUserDetailsService;

@Autowired
publicJwtFilter(@Lazy JwtProviderjwtProvider, @Lazy
GuideDetailsServicecustomUserDetailsService){
this.customUserDetailsService=
customUserDetailsService;this.jwtProvider= jwtProvider;
}


@Override
publicvoiddoFilter(ServletRequestrequest,
ServletResponseresponse,FilterChainchain)
throwsIOException,ServletException{

varhttpServletRequest=(HttpServletRequest)request;varhttpServletRe
sponse=(HttpServletResponse)response;

if(httpServletRequest.getRequestURI().contains("login"))
{chain.doFilter(request,response);
return;
}

Stringtoken=getTokenFromRequest(httpServletRequest);

if(token!= null&&jwtProvider.validateAccessToken(token))
{varuserEmail=jwtProvider.getEmailFromAccessToken(token);
varcustomUserDetails=customUserDetailsService.loadUserByUsername(userE
mail);
if(customUserDetails!=null){
varauth=
newUsernamePasswordAuthenticationToken(customUserDetails,null,customUserDetails
.getAuthorities());
SecurityContextHolder.getContext().setAuthentication(auth);
}
chain.doFilter(request,response);
}else{
```

```
httpServletResponse.setStatus(HttpStatus.UNAUTHORIZED.value());
}

}

privateStringgetTokenFromRequest(HttpServletRequestrequest)
{Stringbearer=request.getHeader("Authorization");
if(hasText(bearer) &&bearer.startsWith("Bearer "))
{returnbearer.substring(7);
}
returnnull;
}
}

@Component
publicclassJwtProvider{

@Value("${jwt.secret}")privat
eStringsecret;

@Value("${jwt.type}")privat
eStringtype;

publicStringgenerateAccessToken(Stringemail,Collection<String>roles)
{
returnJwts.builder()
.setHeaderParam("typ",type)
.claim("roles",roles.toArray())
.setSubject(email)
.setExpiration(
Date.from(LocalDateTime.now()
.plus(Duration.of(30000,ChronoUnit.MINUTES))
.atZone(ZoneId.systemDefault()).toInstant())
)
.signWith(SignatureAlgorithm.HS512,secret)
.compact();
}

publicbooleanvalidateAccessToken(Stringtoken) {try{
Jwts.parser().setSigningKey(secret).parseClaimsJws(token);returntrue;
} catch(Exceptione) {returnfalse;
}
}

publicStringgetEmailFromAccessToken(Stringtoken)
{returngetAccessTokenBody(token).getSubject();
}

publicCollection<String>getRolesAccessToken(Stringtoken){
          returngetAccessTokenBody(token).get("roles",Collection.class);
}

privateClaimsgetAccessTokenBody(Stringtoken) {return
Jwts.parser().setSigningKey(secret).parseClaimsJws(token).getBody();
}
}
```

```
@Data@NoArgsConstruct
or@AllArgsConstructor
publicclassAccessToken{privateSt
ringaccessToken;
}
@RestController@RequestMapping
("/auth")publicclassAuthContro
ller{

privateAuthenticationManagerauthenticationManager;privateJ
wtProviderjwtProvider;

publicAuthController(AuthenticationManagerauthenticationManager,JwtProvi
derjwtProvider) {
this.authenticationManager=
authenticationManager;this.jwtProvider=jwtProvider;
}

@PostMapping("/guide/login")
publicResponseEntity<LoginDtoResponse>loginGuide(@RequestBodyLoginDto
RequestloginDtoRequest) {
try{authenticationManager.authenticate(
newUsernamePasswordAuthenticationToken(loginDtoRequest.getEmai
l(),loginDtoRequest.getPassword(),Arrays.asList(new
SimpleGrantedAuthority(Role.GUIDE.name()))
)
);
}catch(Exceptione){
returnnewResponseEntity("Authenticationfailed!",HttpStatus.UNAUTH
ORIZED);
}

varaccessToken=
newAccessToken(jwtProvider.generateAccessToken(loginDtoRequest.getEmail(),
Arrays.asList(Role.GUIDE.name()))
);

returnnewResponseEntity(accessToken,HttpStatus.OK);
}

@PostMapping("/tourist/login")
publicResponseEntity<LoginDtoResponse>loginTourist(@RequestBodyLoginDto
RequestloginDtoRequest) {
try{authenticationManager.authenticate(
newUsernamePasswordAuthenticationToken(loginDtoRequest.getEmai
l(),loginDtoRequest.getPassword(),Arrays.asList(new
SimpleGrantedAuthority(Role.TOURIST.name()))
)
);
}catch(Exceptione){
returnnewResponseEntity("Authenticationfailed!",HttpStatus.UNAUTH
ORIZED);
}

varaccessToken=newAccessToken(
```

```
jwtProvider.generateAccessToken(loginDtoRequest.getEmail(),Arrays.asList(Role.TOURIS
T.name()))
);

returnnewResponseEntity(accessToken,HttpStatus.OK);
}
}
```

### Gateway-Service

```
@SpringBootApplication@E
nableEurekaClient
publicclassGatewayServiceApplication{publicsta
ticvoidmain(String[]args){
SpringApplication.run(GatewayServiceApplication.class,args);
}
}

@Configuration
publicclassCorsConfiguration{

@Bean
publicCorsWebFiltercorsFilter()
{org.springframework.web.cors.CorsConfigurationcorsConfiguration=
neworg.springframework.web.cors.CorsConfiguration();cor
sConfiguration.setAllowCredentials(true);

corsConfiguration.setAllowedOrigins(Arrays.asList("http://localhost:4200"));corsConfi
guration.setAllowedMethods(Arrays.asList("GET","POST",
"PUT","DELETE","OPTIONS","HEAD"));
corsConfiguration.addAllowedHeader(ALL);UrlBasedCorsConfigurationSo
urcesource=new
UrlBasedCorsConfigurationSource();source.registerCorsConfiguration("/
**", corsConfiguration);returnnewCorsWebFilter(source);
}

}
```

### Tour-Service

```
@RestController@RequestMapping
("/tour")publicclassTourContro
ller{

privateTourServicetourService;privateG
uideClientguideClient;

@Autowired
publicTourController(TourServicetourService, GuideClientguideClient)
{this.tourService= tourService;
this.guideClient=guideClient;
}

@GetMapping("/{tourId}")
publicResponseEntity<TourDtoResponse>getTourById(@PathVariable UUIDtourId){
returnnewResponseEntity<>(tourService.getById(tourId),HttpStatus.O
K);
}
```

```
@PostMapping("/{guideId}")
publicResponseEntity<TourDtoResponse>createTourByGuideId(@Validated@Request
Body TourDtoRequesttourDtoRequest,
@PathVariable UUID guideId)
{InnerGuideDtoinnerGuideDto=
guideClient.getInnerGuideByGuideId(guideId);
returnnewResponseEntity<>(tourService.create(tourDtoRequest,innerGuideDt
o.getGuideId()),HttpStatus.CREATED);
}

@PutMapping("/{tourId}")
publicResponseEntity<TourDtoResponse>updateTourById(@Validated@RequestBodyTou
rDtoRequesttourDtoRequest,@PathVariableUUIDtourId){
returnnewResponseEntity<>(tourService.update(tourDtoRequest,tourId),Http
Status.OK);
}

@DeleteMapping("/{tourId}")
publicResponseEntitydeleteTourById(@PathVariable UUID tourId)
{tourService.delete(tourId);
returnnewResponseEntity(HttpStatus.OK);
}

@GetMapping("/guide/{guideId}")
publicResponseEntity<Collection<TourDtoResponse>>getToursByGuideId(@PathVar
iable StringguideId,
@RequestParam(defaultValue = "0")
intpageNo,@RequestParam(defaultValue="10")intitemsNumber){

returnnewResponseEntity<>(tourService.getToursByGuideIdAndPagination(guideId,pa
geNo,
itemsNumber),HttpStatus.OK);
}

@GetMapping("/page")
publicResponseEntity<TourDtoResponse>getPageWithFiltersAndPagination(@Reques
tParam(required = false, defaultValue = "0")
intpageNo,@RequestParam(required=false,defaultValue="10")int
itemsNumber,
@RequestParam(required = false)
DatestartDate,@RequestParam(required = false) UUID
typeId,@RequestParam(required=false)Floatprice){

returnnewResponseEntity(
tourService.getTourPageWithFiltersAndPagination(typeId,
startDate,price,pageNo,
itemsNumber),HttpStatus.OK);
}
}

@Service
publicclassTourServiceImplimplementsTourService{

privateTourRepositorytourRepository;
privateTourTypeRepositorytourTypeRepository;

@Autowired
publicTourServiceImpl(TourRepositorytourRepository,TourTypeRepository
```

```
tourTypeRepository){
this.tourRepository=
tourRepository;this.tourTypeRepository=tourTypeRepository;
}

@Override@Transactio
nal
publicTourDtoResponsecreate(TourDtoRequesttourDtoRequest, StringguideId){
TourTypetype=checkTourTypeExistence(tourDtoRequest.getType());

Tourtour =
newTour();tour.setType(type);tour.s
etGuideId(guideId);

tour =tourRepository.save(TourMapper.tourDtoRequestToTour(tourDtoRequest,tour));
returnTourMapper.tourToTourDtoResponse(tour);
}

@Override
publicTourDtoResponseupdate(TourDtoRequesttourDtoRequest,UUIDtourId)
{
TourTypetype=checkTourTypeExistence(tourDtoRequest.getType());

vartourOptional=
tourRepository.findById(tourId);if(tourOptional.isEmpty()) {
thrownewRuntimeException("Notourfoundtoupdate");
}

vartour =
tourOptional.get();tour.setType(type);
tour =tourRepository.save(TourMapper.tourDtoRequestToTour(tourDtoRequest,tour));
returnTourMapper.tourToTourDtoResponse(tour);
}

@Override
publicvoiddelete(UUIDtourId){
vartourOptional=
tourRepository.findById(tourId);if(tourOptional.isEmpty()) {
thrownewRuntimeException("Notourtodelete.");
}
tourRepository.deleteById(tourId);
}
@Override
publicTourDtoResponsegetById(UUIDtourId){
vartourOpt=
tourRepository.findById(tourId);if(tourOpt.isEmpty()) {
thrownewRuntimeException("Tournotfound.");
}
returnTourMapper.tourToTourDtoResponse(tourOpt.get());
}

@Override
publicCollection<TourDtoResponse>getToursByGuideIdAndPagination(StringguideId,
intpageNo,intitemsNumber) {
returnTourMapper.toursToTourDtosResponse(tourRepository
.findToursByGuideId(guideId,PageRequest.of(pageNo,
itemsNumber)));
}
```

```
privateTourTypecheckTourTypeExistence(TourTypeDtotypeDto){
vartypeOptional=
tourTypeRepository.findById(typeDto.getId());if(typeOptional.isEmpty()) {
thrownewRuntimeException("Tourtypenotfound");
}
returntypeOptional.get();
}

@Override
publicTourPageDtoResponsegetTourPageWithFiltersAndPagination(
UUID typeId, DatestartDate, Floatprice, intpageNo, intitemsNumber){

Page<Tour>tourPage;
startDate = startDate == null?
newDate(Calendar.getInstance().getTimeInMillis()):startDate;
varpriceVal=price==null?Float.MAX_VALUE:price;

if(typeId == null) {tourPage=
tourRepository.getTourPagesByStartDateGreaterThanEqualAndPriceLessThanEqual(startDa
te,
priceVal,
PageRequest.of(pageNo,itemsNumber)
);
}else{
tourPage=tourRepository.getTourPagesByTypeIdAndStartDateGreaterThanEqualAndPriceLessT
hanEqual(
typeId,startDate,priceVal,
PageRequest.of(pageNo,itemsNumber)
);
}

varpage=newTourPageDtoResponse(

TourMapper.toursToTourDtosResponse(tourPage.get().collect(Collectors.toList()
)),
tourPage.getNumber(),tourPage.getTotalPages(),tourPage.ge
tTotalElements(),tourPage.getNumberOfElements()
);
returnpage;
}

@Data@Ent
ity
@Table(name="tours")
publicclassTourimplementsSerializable{

@Id
@GeneratedValue(generator =
"uuid2")@GenericGenerator(name="uuid2",strategy="uuid2")
@Column(name = "tour_id", updatable = false, nullable =
false,columnDefinition="VARCHAR(36)")
@Type(type = "uuid-
char")privateUUID id;

@Column(name="name")
```

```
privateStringname;

@Column(name =
"description")privateStringdescript
ion;

@Column(name =
"start_date")privateDatestartDate;
@Column(name =
"price")privatefloatprice;

@Column(name =
"duration")privatelongduration;

@Column(name =
"main_img_id")privateStringmainImgI
d;

@Column(name = "guide_id", nullable = false, updatable =
false,columnDefinition="VARCHAR(36)")
privateStringguideId;

@OneToOne(cascade = CascadeType.ALL, fetch = FetchType.LAZY, mappedBy
="tour")
@JoinColumn(name = "loc_id", referencedColumnName =
"loc_id")privateLocationlocation;

@ManyToOne(cascade = CascadeType.MERGE, fetch =
FetchType.LAZY)@JoinColumn(name = "type_id", referencedColumnName =
"type_id")privateTourTypetype;

@CreationTimestamp
@Column(name = "created_at", updatable = false, nullable =
false)privateTimestampcreatedAt;

@UpdateTimestamp@Column(name =
"updated_at")privateTimestampupda
tedAt;
}
```

### Config-Service

```
@SpringBootApplication@E
nableConfigServer@Enable
EurekaClient
publicclassConfigServerApplication{publicstat
icvoidmain(String[]args){
SpringApplication.run(ConfigServerApplication.class,args);
}
}
```

### Registry-Service

```
@SpringBootApplication@E
nableEurekaServer
publicclassEurekaRegistryApplication{publicsta
ticvoidmain(String[]args){
SpringApplication.run(EurekaRegistryApplication.class,args);
}
}
```

### Tourist-Service

```
@RestController@RequestMapping("/
tourist")publicclassTouristContro
ller{

privateTouristServicetouristService;@Autowir

ed
publicTouristController(TouristServicetouristService){
this.touristService=touristService;
}

@GetMapping("/all")
publicResponseEntity<List<TouristDtoResponse>>getAll(){
returnnewResponseEntity<>(touristService.getAll(),HttpStatus.OK);
}

@GetMapping("/{id}")
publicResponseEntity<TouristDtoResponse>getById(@PathVariableUUIDid)
{
returnnewResponseEntity<>(touristService.getById(id),
HttpStatus.OK);
}

@PostMapping
publicResponseEntity<TouristDtoResponse>create(@RequestBodyTouristD
toRequesttouristDtoRequest){
returnnewResponseEntity<>(touristService.create(touristDtoRequest),HttpStatus.C
REATED);
}

@PutMapping("/{id}")
publicResponseEntity<TouristDtoResponse>update(@RequestBodyTouristD
toRequesttouristDtoRequest,@PathVariableUUIDid){
returnnewResponseEntity<>(touristService.update(touristDtoRequest,id),HttpStat
us.OK);
}

@DeleteMapping("/{id}")
publicResponseEntitydeleteById(@PathVariable UUID id)
{touristService.delete(id);
returnnewResponseEntity(HttpStatus.OK);
}

@PostMapping("/reg")
publicResponseEntity<TouristDtoResponse>registerTourist(@RequestBodyTouristD
toRegRequesttouristDtoRegRequest){
returnnewResponseEntity(touristService.registerTourist(touristDtoRegRequest),HttpStat
us.CREATED);
}
}

@Data@Ent
ity
@Table(name="tourists")
publicclassTouristimplementsSerializable{

@Id
@GeneratedValue(generator =
"uuid2")@GenericGenerator(name="uuid2",strategy="uuid2")
```

```
@Column(name = "tourist_id", updatable = false, nullable =
false,columnDefinition="VARCHAR(36)")
@Type(type = "uuid-
char")privateUUID id;

@Column(name = "username", unique =
true)privateStringusername;

@Enumerated(value =
EnumType.STRING)@Column(name = "role",
nullable = false)privateRolerole;

@Column(name = "email", unique = true, nullable =
false)privateStringemail;

@Column(name = "phone", unique =
true)privateStringphone;

@Column(name =
"first_name")privateStringfirstNam
e;

@Column(name =
"last_name")privateStringlastName
;

@CreationTimestamp
@Column(name = "created_at", updatable = false, nullable =
false)privateTimestampcreatedAt;

@UpdateTimestamp@Column(name =
"updated_at")privateTimestampupda
tedAt;

@Column(name = "password", nullable =
false)privateStringpassword;

@Enumerated(value=EnumType.STRING)
@Column(name = "account_status", nullable =
false)privateAccountStateaccountState;
}

@Service
publicclassTouristServiceImplimplementsTouristService{

privateTouristRepositorytouristRepository;privateP
asswordEncoderencoder;

@Autowired
publicTouristServiceImpl(TouristRepositorytouristRepository,Password
Encoderencoder) {
this.touristRepository=
touristRepository;this.encoder= encoder;
}

@Override
publicTouristDtoResponsecreate(TouristDtoRequesttouristDtoRequest)
{Touristtourist=
touristRepository.save(TouristMapper.touristDtoRequestToTourist(touristDtoRequest
,newTourist()));
returnTouristMapper.TouristToTouristDtoResponse(tourist);
}

@Override
```

```
publicTouristDtoResponseupdate(TouristDtoRequesttouristDtoRequest,UUIDtouri
stId) {
varoptionalTourist=
touristRepository.findById(touristId);Touristtourist=
touristRepository.save(TouristMapper.touristDtoRequestToTourist(touristDtoRequest
,optionalTourist.get()));
returnTouristMapper.TouristToTouristDtoResponse(tourist);
}

@Override
publicvoiddelete(UUIDtouristId){
vartouristOptional=
touristRepository.findById(touristId);touristRepository.deleteById(tou
ristId);
}

@Override
publicList<TouristDtoResponse>getAll() {return
TouristMapper.TouristsToTouristDtosResponse(touristRepository.findAll());
}

@Override
publicTouristDtoResponsegetById(UUIDtouristId){
varoptionalTourist= touristRepository.findById(touristId);return
TouristMapper.TouristToTouristDtoResponse(optionalTourist.get());
}

@Override
publicTouristDtoRegResponseregisterTourist(TouristDtoRegRequesttouristD
toRegRequest){
vartouristOptional=touristRepository.findByEmail(touristDtoRegRequest.getEmail());
if(touristOptional.isPresent()){
returnnewTouristDtoRegResponse(RegStatus.FAILED,
"Registrationfailed.Emailalreadyexists.");
}

Touristtourist=
newTourist();tourist.setEmail(touristDtoRegRequest.getEmail());tourist.setFirstNam
e(touristDtoRegRequest.getFirstName());tourist.setLastName(touristDtoRegRequest.get
LastName());

tourist.setPassword(encoder.encode(touristDtoRegRequest.getPassword()));tourist.s
etAccountState(AccountState.NEW);tourist.setRole(Role.TOURIST);
touristRepository.save(tourist);

returnnewTouristDtoRegResponse(RegStatus.NEW,"success");
}

publicTouristgetByEmail(Stringemail)throwsSQLException{

vartouristOptional=touristRepository.findByEmail(email);returntouristOptio
nal.get();
}
}
```

## Додаток Б. Частина програмного коду веб- клієнта

```
import{NgModule}from'@angular/core';
import{BrowserModule}from'@angular/platform-browser';
import{ HttpClientModule, HttpRequest}
from'@angular/common/http';import{JwtHelperService,JwtModule}from"@auth
0/angular-jwt";

import{ AppRoutingModule} from'./app-
routing.module';import{AppComponent}from'./app.component';
import{ FormsModule, ReactiveFormsModule}
from'@angular/forms';import{CommonModule,DatePipe}from'@angular/comm
on';
import{                                        HeaderComponent}
from'./header/header.component';import{       FooterComponent}
from'./footer/footer.component';import{TourModule}
from'./tour/tour.module';
import{ TourFormComponent} from'./guide/tour-form/tour-
form.component';import{GoogleMapsModule}from'@angular/google-maps';
import{GuideModule}from'./guide/guide.module';
import{ AuthenticationModule}
from'./authentication/authentication.module';import{NgbModule}from'@ng-
bootstrap/ng-bootstrap';

exportfunctiontokenGetter(request?: HttpRequest<any>)
{if(request){
console.log(request.url)
if(request.url.includes("tourist"))
{console.log("tourist_access_token")
returnlocalStorage.getItem("tourist_access_token")
} elseif(request.url.includes("guide"))
{console.log("guide_access_token")localStorage.getItem("guide_access_token");
}
}
returnnull;
}

@NgModule({declarations:
[
AppComponent,HeaderComp
onent,FooterComponent,T
ourFormComponent
],
imports:
[CommonModule,BrowserMo
dule,HttpClientModule,A
ppRoutingModule,
```

```
FormsModule,ReactiveForms
Module,GoogleMapsModule,T
ourModule,GuideModule,Aut
henticationModule,JwtModu
le.forRoot({
config:
{tokenGetter,
allowedDomains:
["http://localhost:8011"],throwNoTokenError: true,
},
}),
NgbModule,Date
Pipe,
],
providers:
[JwtHelperService
],
bootstrap:[AppComponent]
})
exportclassAppModule{}
```

```
constroutes:Routes =[
{path:'',        redirectTo:'/tours',pathMatch:'full'},
{
title:
"tours",path:"tours
",
component:TourSearchListComponent
},
{
title:"guid_control",path:
"guid_control",component:TourFor
mComponent
},
{
title: "guid_control_tour_list",path:
"guid_control_tour_list",component:GuideT
ourListComponent
},
{
title:"tour-details",
path: "tour-
details/:tourId",component:TourDeta
ilsComponent
},
```

```
path:"auth",
component:
AuthenticationComponent,children: [
{
title: "guide-
login",path:"guide-login",
component:GuideLoginComponent
},
{
title: "guide-reg",path:
"guide-reg",
component:GuideRegComponent
},
{
title: "tourist-
login",path:"tourist-login",
component:TouristLoginComponent
},
{
title: "tourist-
reg",path:"tourist-reg",
component:TouristRegComponent
},
]
}
];

@NgModule({
imports:
[RouterModule.forRoot(routes)],exports:[Rout
erModule]
})
exportclassAppRoutingModule{}


@Injectable({providedIn:
'root'
})
exportclassTourService{

privatebaseURL="http://localhost:8011/tour-
service/tour/%endpoint%"privatereadonlygetPage="page"
privatereadonlycreateEndpoint="guide/"

constructor(privatehttp:HttpClient,publicdatepipe:DatePipe){}

publiccreateTourWithGuideId(tour: Tour, guideId: string): Observable<Tour>
{returnthis.http.post<Tour>(this.getFinalUrl(guideId),tour);
}
```

```
publicgetToursByGuideId(guideId: string, pageNo= 0, itamsNumber=
20):Observable<Tour[]>{
constparams= newHttpParams().set('pageNo',
pageNo).set('itamsNumber',itamsNumber);
returnthis.http.get<Tour[]>(this.getFinalUrl(this.createEndpoint+guideId),
{params});
}

publicgetToursById(tourId: string): Observable<Tour>
{returnthis.http.get<Tour>(this.getFinalUrl(tourId));
}

privategetFinalUrl(tail:string):string{returnthis.bas
eURL.replace("%endpoint%",tail);
}

publicgetTourPageWithFiltersAndPagination(searchParams: TourSearchParams=
{pageNo:0,
itamsNumber:
10,typeId:null,
startDate: this.datepipe.transform(Date.now(), 'yyyy-MM-
dd'),price:Number.MAX_SAFE_INTEGER
}):Observable<TourPage>{

letparams=newHttpParams()
.set('pageNo',searchParams.pageNo?searchParams.pageNo:0)
.set('itamsNumber',searchParams.itamsNumber?searchParams.itamsNumber:
10)
.set('price',searchParams.price?searchParams.price:
Number.MAX_SAFE_INTEGER);


if(searchParams.typeId){
params=params.set('typeId',searchParams.typeId);
}

if(searchParams.startDate){
params=params.set('startDate',searchParams.startDate);
}


returnthis.http.get<TourPage>(this.getFinalUrl(this.getPage),{params});
}
}

@NgModule({declarations:
[
TourComponent,TourSearchListCo
mponent,
```

```
TourSearchFormComponent,TourDe
tailsComponent
],
imports:
[CommonModule,FormsModule,
ReactiveFormsModule,NgbMod
ule,
],
providers: [TourService,
DatePipe],exports:[
TourComponent,TourSearchListCompo
nent,TourDetailsComponent
]
})
exportclassTourModule{}

<formid="searchForm"[formGroup]="searchForm"(ngSubmit)="searchByForm()">
<selectid="type"name="type"formControlName="typeId">
<optionvalue=""selected>Будь-який</option>
<ng-container*ngFor="lettypeoftourTypes$|async">
<option[value]="type.id">{{type.name}}</option>
</ng-container>
</select>
<inputid="location" type="text"
formControlName="location"placeholder="Kyiv,Ukraine">
<inputid="startDate"type="date"formControlName="startDate">
<inputid="price" type="number" formControlName="price" placeholder="Махціна">
<buttontype="submit">Пошук</button>
</form>

#searchForm{
display:flex;
flex-direction:
column;background-color:
#ffd900;border-radius:
15px;padding: 30px30px;

input, select, button{margin:
20px 0px;font-size:
21px;padding: 15px
30px;border-radius:15px;
border:1pxsolid#343434;
}

button{
```

```
cursor:pointer;
}
}


@Component({
selector:'app-tour-search-form',
templateUrl: './tour-search-
form.component.html',styleUrls:['./tour-search-
form.component.scss']
})
export class TourSearchFormComponent implements OnInit

{@Output()searchEvent=newEventEmitter<TourSearchParams>();

protected searchForm: FormGroup =
this.formBuilder.group({typeId:[""],
location:[""],
startDate:[""],
price:[""],
});

tourTypes$:Observable<TourType[]>;

constructor(private formBuilder: FormBuilder, private
tourTypeService:TourTypeService){
this.tourTypes$=this.tourTypeService.getAllTourTypes();
}

ngOnInit(): void{
}

searchByForm(){
this.searchEvent.emit(this.searchForm.value as
TourSearchParams);console.log(this.searchForm.value);
}

}
```