Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)
Кафедра Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

Розробка інтелектуальної туристичної системи на базі ASP.NET Core MVC з використанням аналізу відгуків та рекомендаційного модуля
(тема)

Виконав:
здобувач четвертого року навчання,
групи ІТШІ-21-2
Ярослав Суровяткін
(власне ім'я, прізвище)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)
Тип програми освітньо-професійна
Освітня програма Штучний інтелект
(повна назва освітньої програми)
Керівник доц. Ігор Магдаліна
(посада, власне ім'я, прізвище)

Допускається до захисту

Завідувач кафедри ШІ _____ Олег ЗОЛОТУХІН
(підпис) (власне ім'я, прізвище)

2025 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук

Кафедра Штучного інтелекту

Рівень вищої освіти перший (бакалаврський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Штучний інтелект
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____» _____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Суровяткіну Ярославу Володимировичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка інтелектуальної туристичної системи на базі ASP.NET Core MVC з використанням аналізу відгуків та рекомендаційного модуля

затверджена наказом університету від 19 травня 20 25 р. № 378 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 25 червня 20 25 р.

3. Вихідні дані до роботи дані Інтернет-джерел, науково-технічні публікації, документація C#, документація ASP .NET Core MVC, відкритий репозиторій даних, набори даних для тренування ML-моделей, документація ML.Net, документація SQL Server

4. Перелік питань, що потрібно опрацювати в роботі
1) Аналіз предметної галузі
2) Опис методів реалізації
3) Практична реалізація

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Строк / терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	19.05.2025	виконано
2	Аналіз предметної галузі	20.05.2025 – 25.05.2025	виконано
3	Огляд існуючих платформ і рішень	26.05.2025 – 27.05.2025	виконано
4	Постановка задачі	27.05.2025	виконано
5	Огляд методів реалізації	27.05.2025 – 28.05.2025	виконано
6	Програмна реалізація	28.05.2025 – 10.06.2025	виконано
7	Огляд можливостей розвитку	11.06.2025 – 13.06.2025	виконано
8	Оформлення пояснювальної записки	14.06.2025 – 17.06.2025	виконано
9	Перевірка на академічний плагіат	17.06.2025	виконано
10	Нормоконтроль	18.06.2025	виконано
11	Підготовка презентації та доповіді	18.06.2025 – 20.06.2025	виконано
12	Попередній захист	20.06.2025	виконано
13	Рецензування	21.06.2025	виконано
14	Захист перед ЕК	25.06.2025	

Дата видачі завдання 19 травня 2025 р.

Здобувач _____
(підпис)

Керівник роботи _____ доц. Ігор Магдаліна
(підпис) (посада, власне ім’я, прізвище)

РЕФЕРАТ

Пояснювальна записка: 70 с., 15 рис., 6 ф., 1 дод., 3 табл., 42 джерела.

АЛГОРИТМ, БАЗА ДАНИХ, БУСТИНГ, ВЕБ-ДОДАТОК, ВІЗУАЛЬНИЙ ІНТЕРФЕЙС, ДОДАТОК, ТУРИЗМ, ТУРИСТИЧНЕ АГЕНСТВО, ASP.NET CORE MVC, BOOTSTRAP, C#, ENTITY FRAMEWORK CORE, ML-МОДЕЛІ, K-NN, SQL SERVER.

Об’єкт дослідження – процес надання персоналізованих туристичних рекомендацій на веб-платформах.

Предмет дослідження – застосування методів аналізу текстових відгуків і рекомендаційних алгоритмів у середовищі ASP.NET Core MVC.

Мета роботи – розробка програмного застосування з використанням технологій ASP.NET Core MVC та Entity Framework Core. Розроблене програмне застосування повинне забезпечити користувачам персоналізовані рекомендації щодо подорожей на основі відгуків.

Методи дослідження – теоретичний (збір та аналіз інформації з наукових публікацій, стандартів і підходів до NLP та рекомендаційних систем), експериментальний (програмна реалізація веб-додатку та тестування). Розробка базується на технологіях C#, ASP.NET Core MVC, Entity Framework Core, Bootstrap, HTML, CSS, MS SQL Server, а також на бібліотеках машинного навчання ML.NET для роботи з природньою мовою та побудови моделей рекомендацій.

У результаті роботи проведено теоретичний аналіз галузі туристичних послуг та алгоритмів рекомендацій, що в ній використовуються. Також було розглянуто різні моделі обробки тексту та фільтрації. Для веб-додатку було реалізовано прототип, який включає в себе машинну модель, яка навчена на реальних відгуках з відкритих наборів даних. На базі цього система вже може надавати рекомендації туристичних маршрутів.

ABSTRACT

Bachelor’s thesis contains: 70 pp., 15 fig., 6 f., 3 tabl., 1 ann., 42 references.

ALGORITHM, APPLICATION, ASP.NET CORE MVC, BOOSTING, BOOTSTRAP, C#, DATABASE, ENTITY FRAMEWORK CORE, K-NN, ML MODELS, SQL SERVER, TOURISM, TRAVEL AGENCY, VISUAL INTERFACE, WEB APPLICATION.

The object of this research is the process of providing personalized travel recommendations on web platforms.

The subject of this research is the application of text-review analysis methods and recommendation algorithms within an ASP.NET Core MVC environment.

The aim of the work is to develop a software application using ASP.NET Core MVC and Entity Framework Core technologies that deliver users personalized travel suggestions based on review analysis.

Research methods include theoretical collection and analysis of scientific publications, standards, and approaches to natural language processing and recommender systems, alongside experimental implementation and testing of the web application. Development relies on C#, ASP.NET Core MVC, Entity Framework Core, Bootstrap, HTML, CSS, MS SQL Server, and the ML.NET library for natural language processing and building recommendation models.

As a result, a theoretical analysis of the tourism services domain and its recommendation algorithms was conducted, various text-processing and filtering models were reviewed, and a prototype web application was implemented, integrating a machine-learning model trained on real-world open-dataset reviews. This enables the system to generate personalized tourism route recommendations.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

Авіа-GDS – Global Distribution System – глобальна система дистрибуції авіаквитків;

Оцінка Marco F1 – середнє арифметичне отриманих оцінок середньо-гармонійного середнього;

ALS – Alternating Least Squares – метод матричної факторизації;

ASP .NET Core MVC – фреймфорк модель-представлення-контролер;

CARS – Context-Aware Recommender System – контекстно-усвідомлена рекомендаторна система;

CB – Content-Based Filtering – контент-орієнтована фільтрація;

CF – Collaborative Filtering – колаборативна фільтрація;

CI/CD – Continuous Integration/Continuous Delivery – постійна збірка і доставка;

CNN – Convolutional Neural Network – згорткова нейронна мережа;

CRM-функції – Customer Relationship Management functions – функції управління взаємодією з клієнтами;

Docker-контейнер – ізольований процес із власним файловим простором та середовищем виконання;

ELK-стек – Elasticsearch + Logstash + Kibana – рішення для збору, обробки та візуалізації логів;

GOSS – Gradient-Based One-Side Sampling – одностороння вибірка на основі градієнта;

Gunicorn/Uwsgi процес – HTTP-сервери WSGI для розгортання Python-додатків;

Jaeger – система розподіленого трасування запитів для моніторингу мікросервісів;

K-NN – K-Nearest Neighbors – K-найближчих сусідів;

Kubernets – платформа оркестрації контейнерів для автоматичного розгортання, масштабування та управління кластером;

Latency – затримка відповіді системи або сервісу;

ML-модель – Machine Learning models – моделі машинного навчання;

ML.NET – фреймворк машинного навчання від Microsoft для .NET;

ONNX – Open Neural Network Exchange – стандарт для обміну ML-моделями;

OTA – Online Travel Agencies – туристичні онлайн сервіси;

RAZOR – синтаксис шаблонів ASP .NET Core для генерації HTML-подань;

REST – Representational State Transfer – архітектурний стиль API;

RNN – Recurrent Neural Network – рекурентна нейронна мережа;

SaaS – Software as a Service – програмне забезпечення як послуга;

Servise mesh – інфраструктурний шар для керування мережею мікросервісів;

SHAP-теплова карта – SHapley Additive exPlanations – візуалізація впливу фіч на прогноз моделі;

SGD – Stochastic Gradient Descent – стохастичний градієнтний спуск;

SVM – Support Vector Machine – підтримкові векторні машини;

TF-IDF вектори – Term Frequency-Inverse Document Frequency – вагове представлення тексту;

XLM-R – Cross-Lingual Language Model – мультимовна трансформерна модель від Hugging Face.

.

ВСТУП

Подорожі, відпустки, пізнання нових культур – все це завжди було бажанням багатьох, але мало кому доступне. Згодом почали з’являтись перші туристичні агентства, а зараз їх існує стільки, що кожен може знайти щось для себе.

В наші часи багато хто змінив своє місце проживання, а разом з тим отримав більше можливостей до подорожей. Про це говорить статистика, адже лише за 9 місяців у 2023 кількість таких подорожей досягла 975 млн. Проте, не дивлячись на таку кількість охочих звідати нові країни, мало хто звертається до звичайних туристичних агентств. А все завдяки розвитку цифрових технологій та, як не дивно, пандемії.

Саме в пандемію відбувся великий бум, і майже всі сфери почали шукати можливість перенести свої послуги в онлайн формат. І якщо туристичні додатки і сайти існували й до цього, то під час цього процесу вони ще більше розширили свої можливості. Щоб підтвердити це, достатньо подивитись на обсяг цифрових туристичних послуг, який вже оцінюється у 800 млрд. доларів, а з таким темпом розвитку до 2032 року вартість перевищить 1 трлн євро.

Все це говорить про те, що класична модель агентства з ручним підбором з бази турів практично не актуальна. Звісно, з одного боку, ніхто не хоче витрачати час на самостійний перегляд можливих варіантів подорожі, підбору житла, екскурсій, читанню коментарів і оцінок, порівняння цін і вибору з всього спектру можливих варіантів одного – ідеального. Тому звертаються до такого агентства, щоб хтось це все зробив за них. Проте, ніхто не гарантує, що фахівець, нехай він буде найкращім у світі, обере саме те, чого ви хотіли.

Саме тому було створено веб-застосунок, який буде сортувати всі можливі варіанти відпочинку та показувати лише ті, які відповідають

бажанням клієнта, а також які вже будуть підібрані відповідно до бюджету, емоційному наповненню і минулому досвіду.

Для його реалізації було використано технологію ASP.NET Core MVC, що поєднує потужність мови C# з гнучкою та безпечною архітектурою веб-застосунків. ASP.NET Core MVC[4] забезпечує чітке розділення логіки, представлення та контролерів, що дозволяє створювати масштабовані та легко підтримувані проекти. Крім того, ця платформа підтримує інтеграцію з Entity Framework Core, що дозволяє ефективно працювати з базою даних без написання складних SQL-запитів[5]. Відповідно теж було розроблено базу даних туристичного агентства, щоб всі дані надійно зберігались. Також, для досягнення необхідної якості пропозицій, було використано модулі машинного навчання. Мовний компонент, за допомогою глибоких нейронних мереж, забезпечує визначення тональності відгуків користувачів і виокремлює необхідні сутності. Рекомендаційний модуль на базі алгоритму, що поєднує ознаки поведінки і контенту, забезпечує реалізацію персоналізованих турів. Обидва модулі навчені та розгорнуті за допомогою ML.NET, що забезпечує єдність та полегшує масштабування в майбутньому.

Такий набір технологій є доцільним з огляду на надійність, продуктивність, активну підтримку спільноти та високу швидкість розробки. Використання ASP.NET Core MVC дозволило сфокусуватися безпосередньо на бізнес-логіці застосунку та створити функціональну і водночас зручну систему для надання професійних послуг у сфері туризму.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

Світ туризму завжди був цікавий людям з різних причин: хтось прагнув збагатитись знаннями про інші культури, хтось порівнював життя в інших країнах із своєю, а хтось знаходив у цьому всьому духовний спокій і сили для подальшої роботи. Час летить і все змінюється: ставлення людей до різних задач, ринок праці, екологія і туризм не є винятком. В цій сфері змінюється все: туристичні місця, підхід країн до прийняття і націленості на туристів, спосіб пошуку наступної подорожі та способи бронювання та реалізації цієї подорожі. Саме тому слід приділити всім цім змінам належну увагу.

1.1 Туризм, його стан та тенденції ринку

Туризм – це явище переміщення людей до інших країн або місць, що не є їхнім звичним середовищем перебування, де вони знаходяться для особистих, ділових або професійних цілей [1].

Відповідно до визначення туристом вважають того, хто подорожує принаймні 80 км від свого будинку принаймні 24 години з будь-яких причин. Проте відомо, що не кожен мандрівник лишається в місці призначення на ніч. Багато кому вистачає кілька годин, щоб відвідати визначні місця, місцеві ресторани та вирушити далі. Яскравим прикладом таких подорожей виступають так звані bus-tours.

1.1.1 Автобусні тури

Автобусний тур – це вид туризму, який відбувається протягом певного періоду, наприклад тиждень, автобусом. Такі тури включають в себе в залежності від часу певну кількість країн.

Такий вид подорожей є одним з найдоступніших подорожей, оскільки за короткий проміжок часу можна побачити і відмітити на своїй мапі подорожей більше 3 країн. Ціна такого туру залежить від країн, які обираються та тривалості. Проте певним плюсом є те, що не потрібно додатково витрачатись на житло. Також такий вид подорожей стає все більш популярним, оскільки ціни на авіаквитки значно зростають [2].

Багато різних фірм займається такими турами, з різним видом послуг: від звичайних «приїхати-погуляти-поїхати» до турів із відпочинком на морі.

Проте, при виборі такого формату відпочинку слід теж пам’ятати, що всі проведені ночі будуть без комфортного сну, у наповненому автобусі, а також на кордонах між країнами можна провести не одну годину без можливості вийти.

Тож тут необхідно кожному особисто все переглядати та шукати більш комфортний автобус, переглядати відгуки про туристичну фірму. Хоча в цю сферу теж потроху входять персоналізовані рекомендації.

1.1.2 Стан ринку та тенденції

Світовий туризм зазнав значного падіння під час пандемії коронавірусу, проте зараз знову набирає обертів.

Станом на кінець 2024 року рівень туристів майже досяг допандемічного рівня [3]. Ця кількість налічує 1,4 мільярди подорожей, що на 11% вище за показники 2023 року. Відповідно до такої щорічної статистики від Всесвітньої туристичної організації (UNWTO) будуються прогнози щодо подальшого зросту попиту в сегменті туризму, який сприятиме соціально-економічному розвитку як розвинених дотепер, так і нових туристичних напрямків.

В свою чергу якість послуг цього сегменту теж не стоїть на місці. Зокрема мова йдеться про онлайн формат. Станом на 2024 рік вартість глобального цифрового ринку туристичних послуг складала 566,7 млрд

доларів. На рисунку 1.1 представлені прогнози щодо майбутньої вартості цього сегменту, за умови, що щорічні темпи зростання будуть на рівні 9,85% [4].

Рисунок 1.1 – Прогнози щодо розвитку онлайн сегменту

За даними які надає BusinessWire розвиток онлайн сегменту буде ще швидшим і вже до 2032 року його вартість буде оцінюватись в 1,97 трлн [5].

Такі прогнози виникають з того, що з'являється багато можливостей для віддаленої роботи, а разом з цим більше часу для подорожей; різні програми для студентів таких як Erasmus, розвиток швидкості інтернету та загальна діджиталізація світу.

Опираючись на такі тенденції ринок стає більш орієнтованим на персоналізовані сервіси та індивідуальні враження. А це в свою чергу збільшує конкуренцію на ринку і змушує використовувати інтелектуальні технології для підтримання конкурентоспроможності.

1.2 Сучасні туристичні фірми

1.2.1 Система туристичних агенств

Для планування коротких подорожей люди зараз мають безліч джерел: одні розкажуть які місця варто відвідати, інші розкажуть про закони і традиції іншої країни, на окремих форумах можна знайти «секретні» локації, а на всесвітніх платформах забронювати квитки, житло та все необхідне. Проте досі існують туристичні фірми, які пропонують безліч турів на різний смак і бюджет, де замовнику достатньо обрати і оплатити, а все інше – завдання агентства. Тож варто розібратись більш детально як вони працюють та чим займаються.

Основним завданням такої фірми є об’єднання товарів і послуг, що задовольняли б потреби клієнта під час подорожі, в єдиний туристичний продукт. В Україні в Законі «Про туризм» прописано, що такий продукт повинен поєднувати в собі не менше ніж дві туристичні послуги, що пропонуються або реалізуються за визначеною ціною. До його складу входять: послуги перевезення, розміщення, організації відвідування об’єктів культури, відпочинку та розваг, реалізації сувенірної продукції тощо [6].

Туристичні послуги та товари можна розбити на дві групи: характерні та супутні. До першої групи відносяться основні послуги, без яких подорож неможлива – розміщення, харчування і транспортування; також можуть бути програмні послуги, які відповідають меті подорожі і регламентуються договором; додаткові послуги, які можуть надаватися за окрему плату (екскурсії, анімаційні послуги, медичні та інші). До другої групи відносяться послуги і товари, які призначені не лише для туристів: благоустрій пляжів, оглядових майданчиків, басейни, тенісні корти, театри, кіноконцертні зали, сувенірна продукція тощо.

Процес створення такого продукту відповідно до підходу структурного проектування ділиться на 5 етапів. Схема підходу наведена на рисунку 1.2 [7].

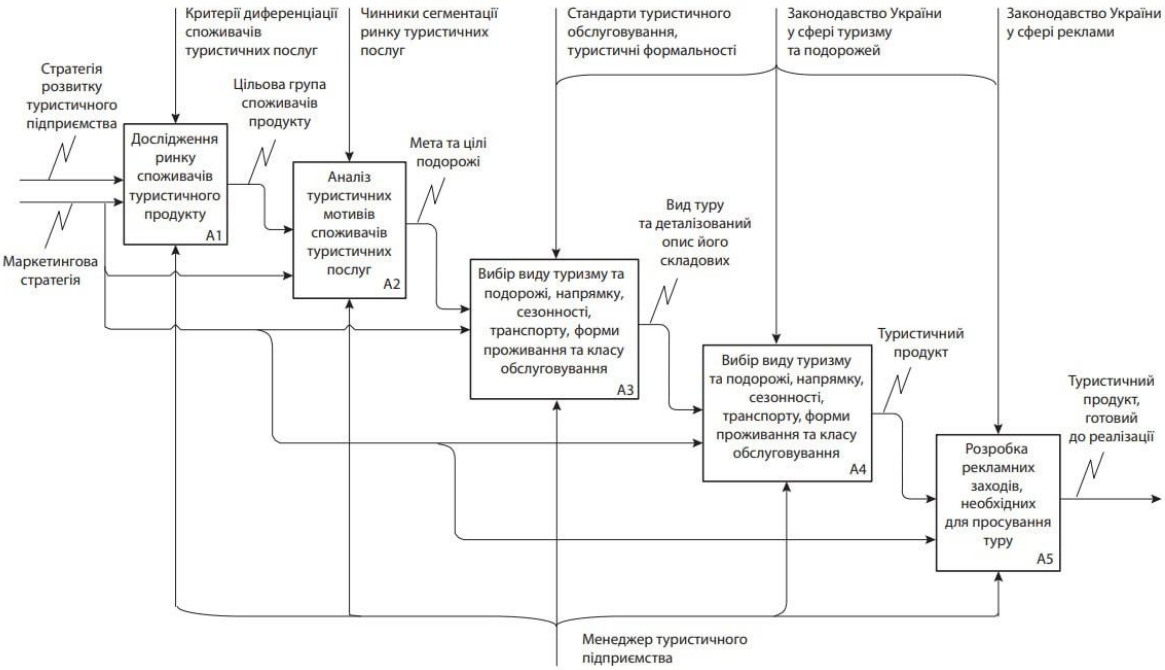

Рисунок 1.2 – Схема створення туристичного продукту

Як можна помітити, що звичайна туристична фірма, яка працює ще за старим підходом має певні слабкі місця, які в свою чергу можуть стати Ахіллесовою п’ятою у ринковій конкуренції. До них відносяться:

– ручний підбір позицій, що значно збільшує час реагування;

– для нових клієнтів відсутня інформація щодо вподобань, тож буде складніше щось для них підібрати;

– затримки між реалізацією певних процесів, які виконуються різними відділами.

Ефективним рішенням цих проблем і є створення єдиної платформи, що об’єднує CRM-функції, модулі аналітики відгуків і системи рекомендацій.

1.2.2 Джерела даних у туризмі

Як і в будь-якому секторі галузі, що має на меті продаж якогось продукту клієнту, в секторі туризму теж необхідні дані клієнтів, щоб краще підбирати пропозиції і бути ближчими до клієнтів.

Такі дані можна поділити на три групи: структуровані, напівструктуровані та неструктуровані. До перших відносяться дані, які опубліковані в офіційних реєстрах або ж певними фірмами, які офіційно займаються статистичними даними (це корисно для загального розуміння ситуації на ринку). До другої групи відносяться дані з «живою» інформацією (наприклад про ціни, розклад або відгуки). Ці дані становлять найбільшу цінність, проте можуть мати необов'язкові поля, а також швидко змінюються. Варто теж зазначити, що деякі джерела офіційної статистики переходять в напівструктурований формат, що дозволяє отримувати дані в режимі онлайн, без завантаження їх в ручну. Ну і до останньої групи входять дані у форматі текстів, відео, фотографій тощо. Найяскравішим прикладом таких даних є відгуки та оцінки. Тут можна виділити головних «поставників»: Booking і GoogleMaps. Перший поділився 1,6 мільйонами відгуків, анонімних звісно, про понад 40 тисяч об'єктів [8], [9]. Мапи ж містять в собі майже 73% всіх оцінок [10].

В сукупності всі ці групи даних створюють потужну базу, на основі якої можна робити точні прогнози, таргетовані пропозиції та будувати клієнтську довіру.

1.3 Аналіз відгуків: методи та моделі

1.3.1 Важливість аналізу

При плануванні своєї подорожі не достатньо лише обрати країну та місто. Для комфортного відпочинку необхідно теж обрати фірму перельоту,

а що не менш важливо – обрати місце відпочинку: квартира, готель або ж хостел. Адже після насиченого дня прогулянок хочеться прийти і спокійно та комфортно набратись сил для подальшої реалізації намічених планів. Тому подорожуючі ретельно обирають місце для сну і в цьому питанні важливу роль відіграють оцінки та відгуки. Згідно зі статистикою 81% туристів читають більше 10 відгуків перед тим як остаточно визначитись з готелем або туром [11]. Саме тому аналіз відгуків, їхня тональність і конкретні тематичні аспекти є важливим питанням.

Аналіз даних – це процес певної послідовності виконуваних дії, що мають на меті інтерпретацію відповідей та їх перетворення у статистику, яка у свою чергу необхідна для ухвалення рішень [12]. Процес аналізу даних можна поділити на три етапи, які представлені на рисунку 1.3.

Рисунок 1.3 – Процес аналізу даних

1.3.2 Методи та моделі аналізу

У зв’язку із залежністю від емоціонального забарвлення контенту, старі методи аналізу стають менш актуальними і їм на зміну приходять більш сучасні технології, які можуть точніше оцінити текст і його настрій.

Якщо говорити про методи аналізу, то перші з них опирались на набори слів та TF-IDF вектори, які оброблялись за допомогою алгоритмів таких як Naive Bayes та SVM. Станом на 2024 рік середня Macro-F1 (середнє

арифметичне отриманих оцінок середньо-гармонійного середнього) таких підходів становила 0,75. Головною проблемою стала майже стовідсоткова не чутливість до багатомовності та контексту [13].

Далі почали з’являтись оновлені моделі, які вже могли більш глибоко аналізувати тексти. Серед них можна виділити FastText і LSTM-мережі, тестування яких збільшило показник F1 до 0,93 [14].

Наступним проривом у сфері аналізу відгуків стала поява так званих трансформерів.

Трансформери – це тип архітектур, які використовуються для обробки та інтерпретації послідовних даних. Вони відрізняються від попередніх моделей тим, що мають паралельну архітектуру. Завдяки цьому, така модель може знаходити залежності між будь-якими елементами послідовності (хоч вони знаходяться близько або далеко один від одного). Такий результат досягається завдяки основному блоку наскрізного самовважання [15]. Крім цього механізму також використовуються й інші:

– енкодери положення, які дозволяють змоделювати приблизне положення елементу;

– позиційно-мультиплікативна вага, яка допомагає контролювати рівень уваги до того чи іншого елементу;

– мультиголовочні, які дозволяють захопити всі рівні взаємозалежностей.

Трансформери існують різних типів з різними особливостями:

– Transformer – це оригінальна модель, що була створена компанією Google;

– BERT – двонаправлена модель від початку до кінця, створена Google AI;

– GPT – генеративний трансформатор від OpenAI;

– T5 – текстово-текстова модель від Google AI.

Звісно від цих моделей теж створювались певні модифіковані моделі. Так, наприклад DCM-BERT, модель, яка була створена на базі

BERT (представлення моделі показано на рисунку 1.4) для сільського туризму Китаю показала точність в 94%, кількість помилок для нових об’єктів скоротилась на 18% [16].

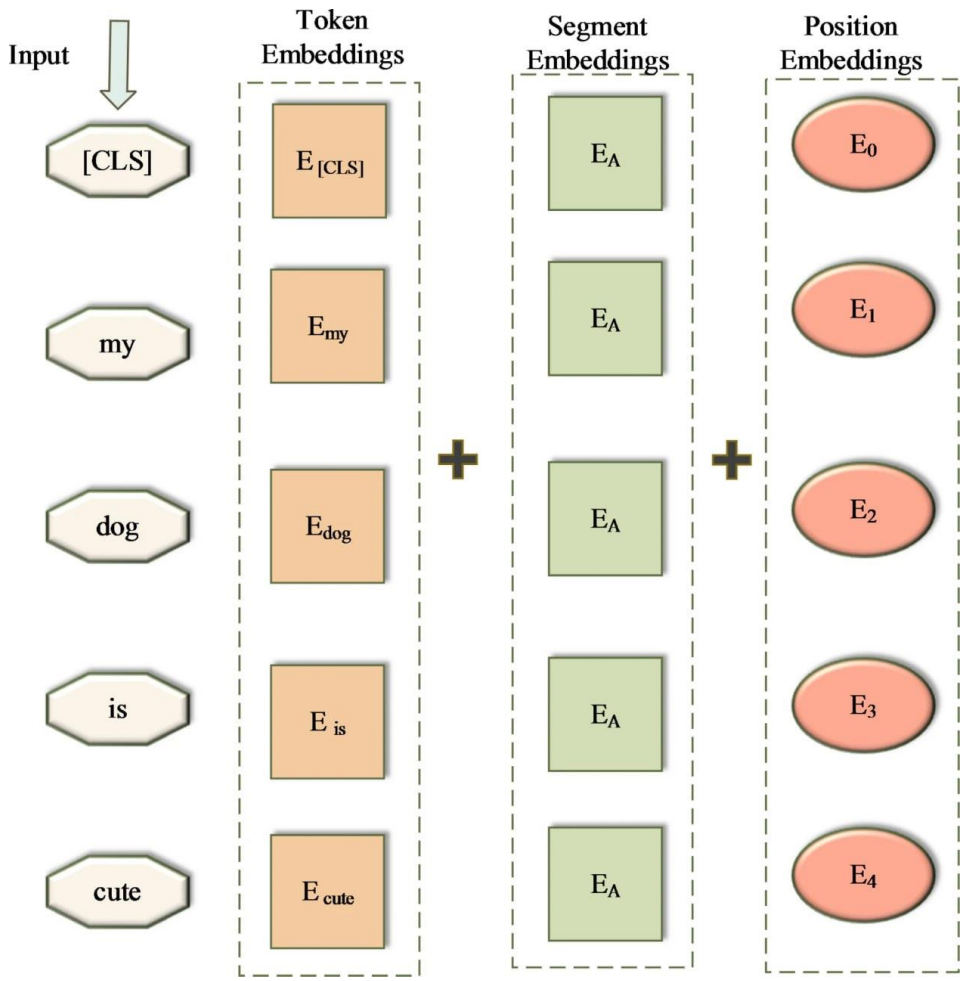

Рисунок 1.4 – Представлення моделі BERT

Також варто відзначити на цьому етапі моделі, які вміють аналізувати дані без додаткового тонкого донавчання. Так, наприклад модель XLM-R Large, в такому «нульовому» режимі зміг показати оцінку F1 з показником 82% на рецензіях англійською та іспанською мовами [17].

З подальшим розвитком та покращенням таких моделей стає ясно, що кращі результати показують процеси, в яких на одному конвеєрі поєднуються різні архітектури. Такий підхід дозволяє знизити вплив шуму на результат виконання аналізу. Так, наприклад поєднання BERT з

BiLSTM-шару з механізмом уваги додало до п’яти пунктів точності у порівнянні із звичайною моделлю BERT [18]. На рисунку 1.5 представлені результати експерименту, у якому досліджувались різні варіанти моделі BERT.

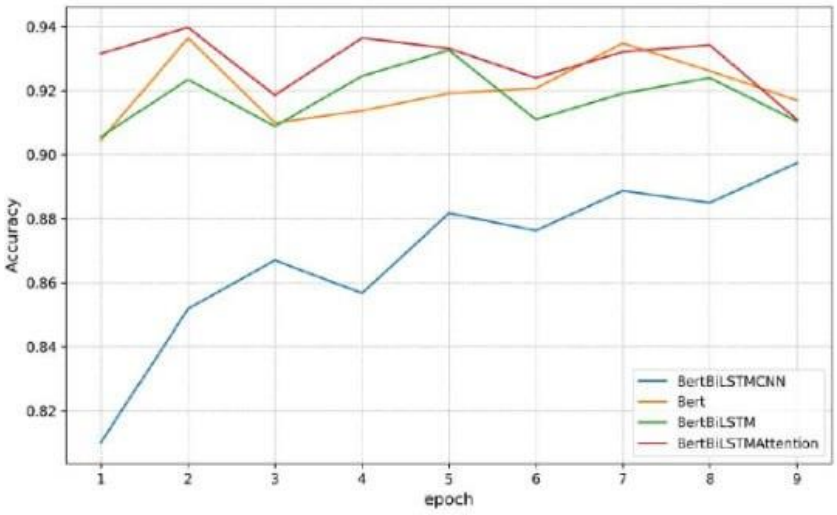

Рисунок 1.5 – Результати порівняння BERT моделей

Наступним різновидом аналітичних моделей є аспект-орієнтовані. Вони залишаються відносно не вибагливими до CPU-ресурсів і показують більше 80% позитивного результату. В цьому ж сегменті існують моделі трансформерів, які дозволяють користувачу бачити які моменти викликають негативну реакцію за допомогою візуалізації SHAP-теплових карт [19].

Теж необхідно пам’ятати про такі мовні моделі як GPT або Claude, які працюють без доменного навчання, мають досить швидке розгортання відповідей і підтримку різних мов. При всьому цьому теж показуються результати у рамках 80% коректності.

У зв’язку з великою кількістю моделей було проведено безліч експериментів, щоб визначити модель, яка б була найкращою для аналітики, проте відомо, що не буває нічого ідеального. Тому оптимальною для сфери

туризму є гібридна форма, в якій моделі типу SBERT або XLM-R відповідають за багатомовні представлення, BiLSTM додає чутливості до послідовностей, а класичні моделі знижують вплив шуму.

Такий підхід дозволяє утримувати якість результатів на рівні 90% при цьому за досить короткий час, без великих ресурсних витрат і що головне, результати лишаються зрозумілими для користувача, який переглядає оцінки того чи іншого продукту.

1.4 Рекомендаційні системи в туристичній сфері

1.4.1 Важливість рекомендаційних систем

Ринок туризму розвивається невпинно, а разом з цим ростуть і пропозиції для туристів.

Зараз існує безліч готелів, екскурсій різного характеру, музеїв, авіакомпаній, автобусних операторів і платформ, які дозволяють знаходити та бронювати необхідні квитки, житло або інші послуги. Звісно обсяг доступних варіантів на таких платформах величезний і кожен зможе знайти те, що йому необхідно. Проте є нюанс – в такій кількості можливих пропозицій легко загубитись. Саме для полегшення користувацького досвіду у сфері цифрового туризму було створено рекомендаційні системи.

Рекомендаційна система – це система, яка надає можливість накопичення інформації про вподобання користувачів і має за ціль забезпечення рекомендацій для користувача у вирішенні ним певних проблем [20]. Тобто завдання такої системи – фільтрування можливих варіанті і надання користувачу короткого списку того, що йому сподобаються.

Найпростіша форма такої моделі – фільтри. Їх можна знайти майже на будь-якому сайті. Це, можна сказати, механічна форма рекомендацій. В ній користувач сам задає необхідні параметри і з-поміж усіх пропозицій

лишаються лише ті, які відповідають виставленим вимогам. Проте, сучасні рекомендаційні системи не про ручну фільтрацію, а про автоматизовану, тож слід більше розібратись саме з цим видом моделей.

1.4.2 Еволюція рекомендаційних систем

Перші системи рекомендацій були згадані у 1990-х роках під назвою «цифрові книжкові полиці», а наприкінці того ж десятиліття вперше таку систему спробували реалізувати у вигляді відеомагнітофону, який мав на меті запис відповідних передач на основі вподобань користувачів. Оскільки на той час ще не існувало такої кількості пристроїв, які б витримували такі навантаження, тому основу системи складала колаборативна фільтрація.

Колаборативна фільтрація не використовувала жодних описів телепередач чи іншої семантики і створювала пропозиції лише на основі подібності поведінки. Такий самий підхід використовувався у перших туристичних рекомендаційних системах. У цьому випадку мандрівникам пропонували подібні тури, на основі порівнянь та виборах схожих користувачів. Схожість в цьому підході визначається за формулою:

$$\mathrm{sim}(u,i) = \frac{\sum_{u \in U_{ij}} (r_{u,i} - \bar{r}_u)(r_{u,j} - \bar{r}_u)}{\sqrt{\sum_{u \in U} ((r_{u,i} - \bar{r}_u))^2} \sqrt{\sum_{u \in U} ((r_{u,j} - \bar{r}_u))^2}}, \quad (1.1)$$

де $u \in U$ – множина користувачів;

$i \in I$ – множина об’єктів;

$(r_{ui}, u, i) \in D$ – множина подій або діяльність, яку виконують на об’єктами.

Підхід колаборативної фільтрації був далеко не ідеальним і не працював тут і зараз, мав проблеми холодного старту, оскільки потребував велику кількість вхідної інформації, проте на той час досить непогано справлявся із поставленою задачею. Але як і всі системи з розвитком

технологій, обсягів інформації та вимог до рекомендацій цей підхід себе вичерпав.

На зміну колаборативній фільтрації прийшли моделі з використанням байєвської класифікації, які дозволили оцінювати і розуміти причини вибору, який зробив користувач. Проте цей підхід швидко змінився іншим через проблему стрімкого зросту кількості інформації.

Наступний вид фільтрації – основана на контенті. У цьому випадку рекомендації базуються на знаннях про продукт, а не про користувача. Метою такої системи рекомендації є рекомендація подібних продуктів на основі його дій в системі. Такі системи базуються на припущеннях про майбутні вподобання користувача на основі попередніх.

Проблемою такої системи рекомендацій може стати одноманітність. Оскільки система базується лише на попередніх діях, то пропонувати буде лише варіанти одного типу, навіть у випадку, якщо клієнт щось не вподобав, рекомендації не зміняться з погляду на відсутність інформації. І так само, як і в колаборативному підході проблемою буде холодний страт, оскільки відсутня інформація про попередній досвід користувача.

Далі вже більш розширені системи рекомендацій, які працюють на основі знань про предметну область, товари та користувачів для прийняття рішення. Такі системи називають експертними. Такі системи можуть враховувати прямі обмеження користувача такі як ціна або оцінка місця/туру.

Проте й тут не все ідеально. Такі системи є складними для впровадження, тому їх використовують в областях, де майже повністю відсутня попередня інформація про вподобання користувача. Ще одним мінусом цього виду рекомендаційних систем є те, що вони створюються під конкретну задачу, тож її не можна буде використовувати для якоїсь видозміненої або повністю іншої задачі.

Наступним етапом розвитку рекомендаційних моделей стало впровадження глибокого навчання. Одною з таких систем є мультимодальна

система SelfAM-Vtrans, яка поєднує в собі візуальність, опис у текстовому форматі та порядок відвідувань [21]. Схема роботи цієї мультимодальної системи представлена на рисунку 1.6.

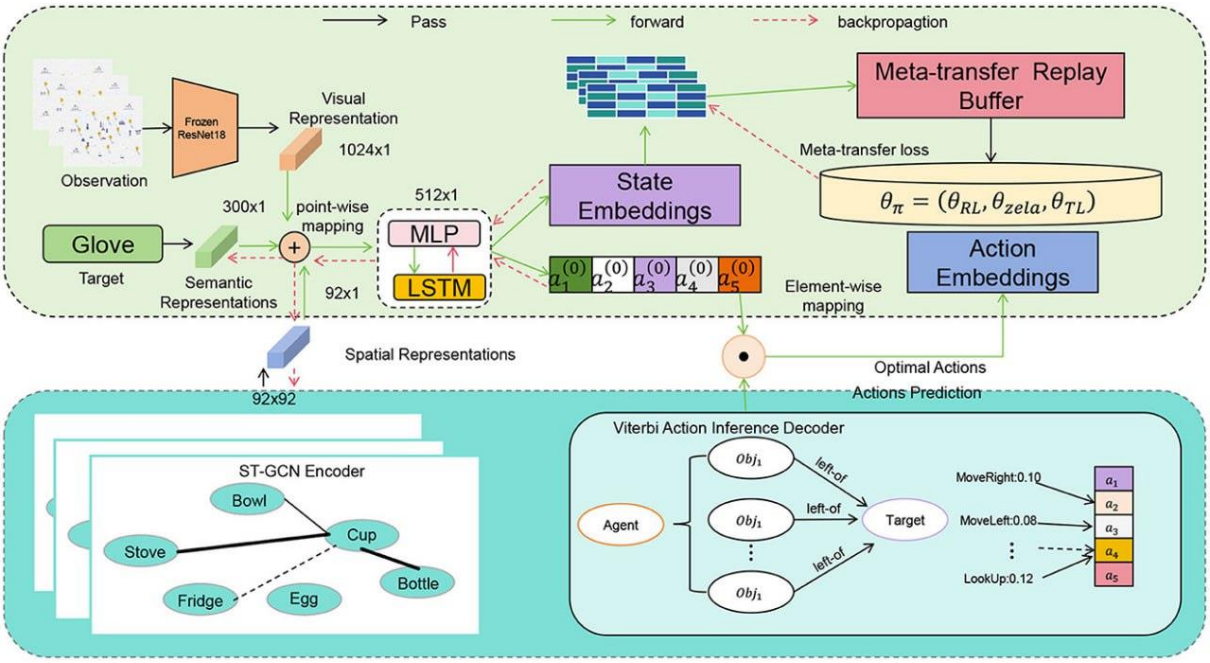

Рисунок 1.6 – Схема мультимодальної системи

Також до тенденцій розвитку рекомендаційних моделей можна віднести впровадження інтерактивного планування, яке запевняє перепланування маршруту на основі дії туриста з мапами. Такі системи вже впроваджені на певних ресурсах і показують непогані результати: на 12% більшу задоволеність клієнтів за менший час пошуку необхідних місць.

Окремим напрямком відзначають системи емоційних та багатокритеріальних рекомендацій. У такому поєднанні система пропонує можливі варіанти подорожі з урахуванням ризиків (таких як погодні умови, перенасиченість туристами тощо), а також «проблемних» варіантів. Таким чином користувач отримує найкращі варіанти для себе, що скорочує кількість відмов на 4% .

Після всіх можливих варіантів систем рекомендацій, зараз маємо одну об’єднану систему, яка враховує всі фактори для підбору персоналізованих рекомендацій. На рисунку 1.7 представлено схему такої гібридної моделі [22].

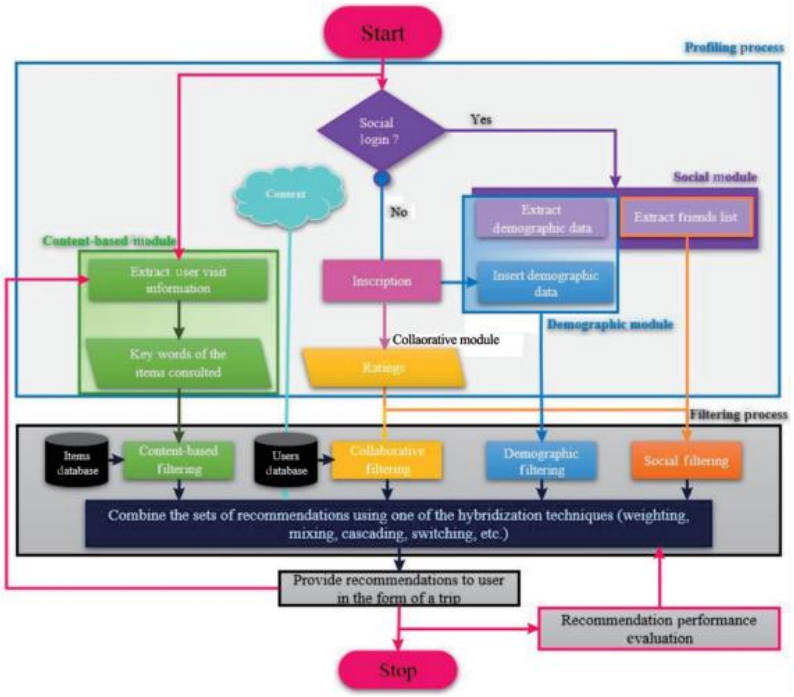

Рисунок 1.7 – Схема гібридної моделі рекомендаційної системи

Такий підхід дозволяє звести до мінімуму недоліки кожного з них, що в свою чергу збільшить задоволення користувача та збільшить їх кількість.

1.5 Існуючі платформи та рішення

На сьогоднішній день мережа наповнена різними платформами та сервісами, що пропонують туристичні послуги, або ж допомагають у самостійній організації відпочинку.

Такі сервіси можна розділити на три групи: OTA-гіганти, SaaS-платформи і нішеві системи. Нижче розглянуто кожну з них більш детально.

1.5.1 OTA-гіганти

OTA-гіганти (англ. Online Travel Agencies) – це платформи, на яких збирають пропозиції певного напрямку від різних постачальників та продають кінцевим споживачам напряму. Такі платформи працюють на B2B умові.

Сервіси цього типу є досить розповсюдженими в теперішній сфері цифрового туризму, частка OTA в бронюваннях показана на рисунку 1.8.

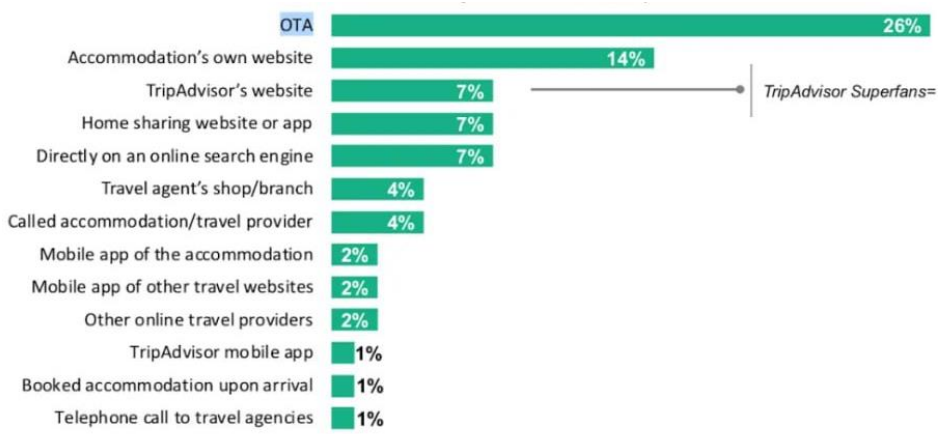

Рисунок 1.8 – Частка OTA на ринку бронювань

Одним з найвідоміших представників цього сегменту є Booking Holdings. Це платформа яка має безліч пропозицій з житлом різного формату: кімнати, готельні номери, квартири та цілі будинки. Цей сервіс пропонує понад 28 мільйонів варіантів житла у 226 країнах.

Цей проект створювався саме для житлових пропозицій, проте зараз ця платформа пропонує також перельоти, оренду автомобілів, можливості для організації дозвілля, таксі та комплекс із перельотом та готелем одразу. Під час реєстрації звісно кожен створює свій профіль і починає переглядати якісь варіанти. Після певних дій на головній сторінці починають з'являтись персоналізовані рекомендації житла та країн, які потенційно можуть сподобатись користувачу. Кожна пропозиція має розділ оцінок та

коментарів, на основі яких теж відбувається фільтрація пропозицій. За даними Yahoo!Finance станом на 2024 рік сума бронювань, які було здійснено через сервіс Booking становить 166 мільярдів доларів.

Щоб підкріпити звання найбільшого сервісу у сфері цифрового туризму варто відзначити, що середня відвідуваність сайту за 2024 рік становить 550 мільйонів унікальних відвідувань. А за березень 2025 показник відвідувань показує майже 525 мільйонів (графік представлено на рисунку 1.9) [23].

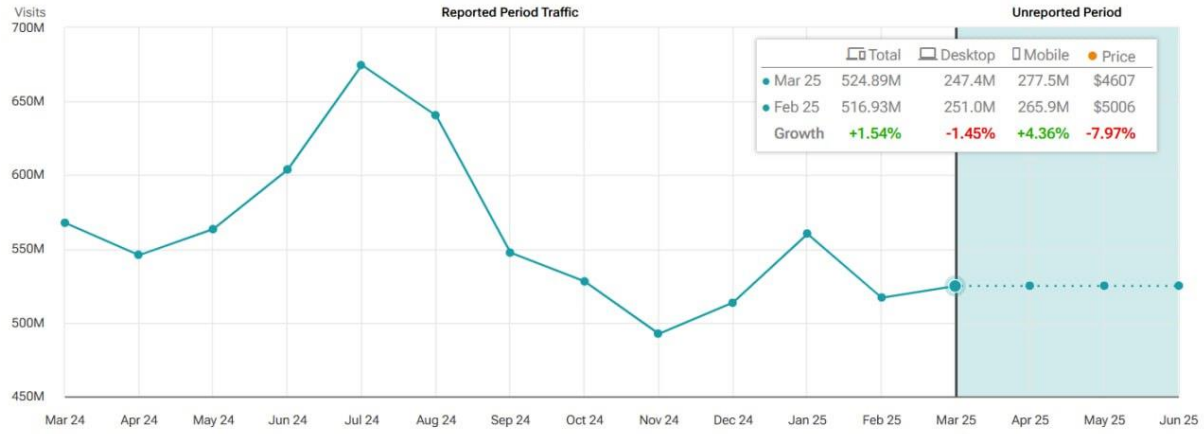

Рисунок 1.9 – Графік відвідувань Booking.com

Наступним гігантом у цій сфері є Airbnb. Це платформа, яка за сферою постачання своїх послуг є подібною до Booking. Головною задачею цього сервісу є надання послуг оренди житла, проте вони теж мають можливість додання до своєї подорожі додаткових послуг: фотосесій, спа-процедур, масажів, активного відпочинку та безліч іншого.

Цей сервіс займає значну частку ринку всіх бронювань. Про це свідчать кількість оброблених заявок, яка становить 115,1 мільйон станом на кінець другого кварталу 2024 року. Цей показник дозволив сервісу очолити перше місце в сфері альтернативних житлових пропозицій.

Дивлячись на представників цього сегменту можна зробити висновок, що саме вони диктують стандарти цієї сфери. Вони мають динамічну

підтримку цінової політики, мільярди відгуків та оцінок, а також практично миттєву відповідь на запити бронювання користувачів.

1.5.2 SaaS-платформи та нішеві системи

SaaS-платформи (Software as a Service) – це формат реалізації ліцензованого програмного забезпечення, до якого користувачі мають доступ через Інтернет.

Іншими словами це, в більшості випадків, сайти, які містять певний набір зібраних пропозицій і продають їх.

Прикладом такої платформи можна виділити Amadeus [24].

Ця платформа поєднує в собі готелі, залізниці, авіакомпанії, турагенції і навіть OTA сегмент. Такий набір дозволяє користувачу переглядати всі пропозиції в одному місці, оформлювати бронювання, купувати та змінювати квитки та багато іншого. Що важливо, так це підтримка цінової відповідності у реальному часі, тобто турист платить ту ціну, яку обрав без жодних раптових змін. При цьому такий мандрівник має єдиний логін на всю подорож, до якого підв’язуються всі пропозиції (житло, квитки, необхідні перевезення тощо).

Про частку цієї платформи на ринку свідчать доходи від послуг. За попередній рік цей показник становив більше ніж 6 мільярдів євро.

До переваг такого підходу можна віднести їх багатомодульність та якісну інтеграцію з авіа-GDS. Проте ціна таких платформ досить висока і якість дуже залежить від постачальників послуг, які буде продавати така платформа.

Якщо ж відійти від таких великих систем до менших, то розглядаються системи, які використовуються для конкретних сегментів туристичної сфери.

Такі системи використовуються у відносно невеликих бізнесах, оскільки не мають настільки розширеного функціоналу, а разом із цим є

менш вибагливими та гнучкими. Проте саме така обмеженість функціоналу разом із слабкими модулями машинного навчання можуть становити перешкоди для розвитку бізнесу.

Прикладом такої системи можна виділити FareHarbor [25]. Ця платформа націлена на допомогу у керуванні пропозиціями та виходом на глобальний рівень. На платформі зібрані різні пропозиції проведення часу від багатьох дистрибуторів активностей. За даними сервісу обсяг пропозицій знаходиться на рівні 125 тисяч.

FareHarbor є одним з найяскравіших та найбільших представників цього напрямку, що підтверджують річні доходи компанії, які становлять близько 169 мільйонів. Проте такі результати стають менш вражаючими, якщо звернути увагу на те, що ця платформа входить до Booking Holdings. Також сервіс користується популярністю, оскільки не вимагає щомісячної підписки, а лише стягує оплату у вигляді комісії, проте і цього можна уникнути, якщо обрати спосіб оплати на місці. Ну і звісно інтеграція з OTA-гігантом Booking збільшує рівень користувачів, оскільки при бронюванні житла на платформі гіганту одразу пропонують пакети «що зробити».

Підбиваючи підсумки, слід зазначити, що ніша цифрового туризму є досить різноманітною в плані способів надання послуг. Хоча кожен зі способів має свої переваги та недоліки, цей напрямок швидко змінюється, вподобання і потреби користувачів теж змінюються, тож ця ніша є відкритою для нових рішень.

1.6 Постановка завдання

Після проведеного аналізу предметної області можна відмітити, що ринок цифрового туризму є досить наповненим різними рішеннями та підходами, щодо запевнення всіх потреб користувача. Проте існуючі підходи є або занадто «важкими» в плані витрат ресурсів або занадто «легкими», які не покривають весь спектр необхідної функціональності.

Об’єктом досліджень цієї роботи є процес надання персоналізованих туристичних рекомендацій на веб-платформах.

Предметом досліджень виступає застосування методів аналізу текстових відгуків і рекомендаційних алгоритмів у середовищі ASP.NET Core MVC.

Метою роботи є розробка програмного застосування з використанням технологій ASP.NET Core MVC та Entity Framework Core, яке повинне забезпечити користувачам персоналізовані рекомендації щодо подорожей на основі відгуків.

Для досягнення поставленої мети необхідно виконати наступне:

– проаналізувати ринок туристичних сервісів;

– розглянути існуючі технології, які використовуються для платформ з подібним функціоналом;

– порівняти існуючі алгоритми з урахуванням їх переваг, недоліків та потреб даного веб-застосунку;

– проаналізувати різні підходи до реалізації;

– обрати відповідний підхід до структурної реалізації;

– обрати алгоритм обробки природної мови, з урахуванням тональності та емоційного забарвлення тексту;

– обрати алгоритм для визначення рекомендацій;

– спроектувати базу даних;

– програмно реалізувати алгоритм рекомендацій на основі результатів обробки відгуків;

– створити інтерфейс користувача;

– об’єднати всі частини програми відповідно до стандартів технології ASP.NET Core MVC;

– визначитись з найбільш відповідними наборами даних для навчання ML-моделей;

– провести навчання ML-моделей;

– провести тестування.

2 ОПИС МЕТОДІВ РЕАЛІЗАЦІЇ

У цьому розділі представлено огляд технологічних парадигм і методів реалізації інтелектуальної туристичної системи. Розглянуто архітектурні підходи, алгоритми машинного навчання та приклади систем з підходами, які вони використовують.

2.1 Порівняння архітектурних підходів

У сучасній інженерії програмного забезпечення вибір архітектурної парадигми визначає здатність системи забезпечувати належну продуктивність, масштабованість і надійність. Монолітна архітектура, в якій усі компоненти – від обробки HTTP-запитів до алгоритмів машинного навчання – впаковані в єдине ASP .NET Core MVC-застосункове середовище, дозволяє оперативно розгорнути прототип і мінімізувати мережеві оверхеди. Проте зі зростанням навантаження та розширенням функціоналу з’являються суттєві обмеження: оновлення будь-якого модуля потребує перезапуску всієї системи, а масштабування «вертикально» призводить до неефективного використання ресурсів і простою сервісів[26].

Натомість мікросервісна архітектура розбиває систему на набір незалежних служб, кожна з яких відповідає за окрему бізнес-або технічну функцію і може автономно масштабуватися. Цей підхід добре зарекомендував себе у великих платформах (наприклад, Booking.com та Expedia), де окремі сервіси для бронювання, відгуків і аналітики запускаються у Docker-контейнерах та оркеструються через Kubernetes, з використанням API Gateway і Service Mesh для забезпечення надійності та безперервної доставки. Проте впровадження мікросервісів потребує значних зусиль з налаштування інфраструктури: організації централізованого логування (ELK-стек), трасування запитів (Jaeger),

управління конфігураціями й міжсервісною безпекою (mTLS, OAuth2), що може відтягнути час виходу MVP на ринок [27].

Гібридний підхід, обраний для реалізації цієї туристичної системи, поєднує переваги обох моделей. Інтерфейс користувача та основна бізнес-логіка зберігаються в монолітному ASP .NET Core MVC-додатку, забезпечуючи швидкий старт і прості CI/CD-процеси, тоді як ресурсоємні сервіси – модулі NLP-аналізу та Recommendation Engine – винесені в окремі Docker-контейнери, які за потреби масштабуються в Kubernetes. Цей компроміс відповідає патерну «Strangler Fig», при якому новий функціонал поступово виокремлюється в мікросервіси, водночас зберігаючи монолітну основу для менш критичних задач [28].

Таким чином, гібридна архітектура забезпечує:

- швидкий MVP без затягування DevOps-процесів;
- цільове масштабування AI-модулів із незалежним нарощенням ресурсів GPU/CPU;
- гнучку еволюцію системи шляхом поступового виокремлення нових сервісів.

Це рішення оптимально збалансовує потребу в ефективному запуску продукту з можливістю горизонтального розширення під високі навантаження та постійну підтримку інтелектуальних компонентів.

2.2 Методи машинного навчання для аналізу відгуків

У межах автоматизованого аналізу відгуків розрізняють п’ять основних груп методів, кожна з яких має власні переваги й недоліки та обирається залежно від вимог до точності й продуктивності.

Перша група це TF–IDF + лінійні класифікатори. В цьому випадку на виході створюється документ, який кодується як вектор ваг TF–IDF, обчислений за формулою 2.1 [29].

$$TF\text{–}IDF(t,d) = TF(t,d) \times log\frac{N}{DF(t)}, \quad (2.1)$$

де N – загальна кількість документів;

DF(t) – кількість документів, що містять термін t.

Після цього застосовується SVM або Naïve Bayes [30].

До переваг цього методу можна віднести просту реалізацію, інтерпретованість ваг термінів та дуже низькі обчислювальні витрати. Серед недоліків можна виділити відсутність адаптації до порядку слів та залежності в контексті, що відіграє важливу роль в оцінці точності цього методу, яка показує себе на рівні 0,75.

Наступною групою є ембеддинг слів + RNN/CNN з увагою. Попередньо навчені ембеддинги (word2vec, FastText) подаються на двонаправні LSTM або CNN із механізмом уваги, що обчислює ваги токенів як:

$$\alpha_j = \frac{exp(e_i)}{\sum_j exp(e_j)}, \quad (2.2)$$

$$e_i = v^T tan(W[h_i; s]), \quad (2.3)$$

де α_j – нормована вага, яка визначає ступіть «уваги» до j-го токена у послідовності;

e_i – «енергія» i-го токена, що відображає його релевантність до контексту;

v – навчальний вектор, який трансформує прихований пристрій в скаляр енергії;

W – навчальна матриця проекція механізму уваги;

h_i – вихід RNN для і-ї позиції;

s – вектор глобального контексту;

$[h_i, s]$ – операція конкатенації локального та глобального представлень [31].

До недоліків цієї групи відносять витрати пам’яті та потребу GPU. Оцінка якості використання такого підходу знаходиться на рівні 0,9. Проте з переваг використання такого методу можна виділити урахування глобального та локального контексту.

До третього групи відносять трансформери (BERT-сімейство, XLM-R). Архітектура трансформерів базується на самоувазі:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \quad (2.4)$$

де Q – матриця запитів;

K – матриця ключів;

V – матриця значень;

d_k – розмірність простору ключів, безрозмірна величина.

Після тонкого донавчання на доменних корпусах BERT-моделі досягають оцінки 0,92 при latency ≈ 120 мс [32]. Серед переваг для цієї групи віділяють state-of-the-art точність, мультилингвальність. Серед мінусів виділяють великі розміри моделі, а також проблему оптимізації через ONNX або квантізацію.

Четверта група це аспект-орієнтований аналіз (ABSA). У цьому підході плюсом є деталізована аналітика «точок болю». Серед недоліків – висока складність розмітки датасету та latency (~200 мс).

До останньої групи відносяться гібридні та мультитаск-моделі. В цьому випаду Поєднання трансформерних ембеддингів із LSTM/CNN або ансамблем Random Forest/XGBoost додає 2–4 п.п. до F1, а multitask-навчання NER+SA скорочує latency на ≈ 30 % та підвищує F1 кожного завдання на ≈ 2 п.п. [33]. Серед переваг виділяють максимальну точність і

стійкість до шуму. А серед недоліків – висока складність для підтримки та вимоги до постійного моніторингу дріфту.

2.3 Методи машинного навчання для рекомендацій

Алгоритмічний апарат рекомендацій подорожей складається з кількох взаємодоповнювальних родин моделей: колаборативних, контент-орієнтованих, контекстно-обізнаних, а також глибоких і гібридних. Кожна з них розв’язує окремі обмеження класичного підходу «користувач × об’єкт» і підвищує релевантність у специфічних умовах туристичного домену.

Колаборативна фільтрація (КФ) - найстаріша й досі найпоширеніша парадигма побудови рекомендацій, у якій оцінка релевантності об’єкта ґрунтується виняткова на колективній поведінці користувачів. Центральним об’єктом КФ є матриця взаємодій $R \in R^{m \times n}$, де m – кількість користувачів, n – кількість турів (або інших продуктів), а елемент $r_{u,i}$ містить явний рейтинг чи неявний сигнал (перегляд, бронювання). Після отримання цієї матриці, вона оптимізується алгоритмом ALS або SGD.

Alternating least squares (ALS) – оптимізаційний алгоритм, який використовується для навчання моделей матричної факторизації. В цьому алгоритмі знаходять дві латентні матриці, які апроксимують матрицю оцінок, алгоритм фіксує одну з матриць і розв’язує задачу найменших квадратів для іншої, після чого робить те саме для іншої матриці. До переваг цього алгоритму можна віднести легку паралелізацію, оскільки кожен крок зводиться до незалежних систем лінійних рівнянь. Також можна виділити швидку розбіжність на великих, але розріджених матрицях. Проте недоліком є чутливість до вибору гіперпараметра і регуляції, а також потреба в достатньо «щільній» матриці оцінок [34].

Stochastic gradient descent (SGD) – теж алгоритм оптимізації, який використовується для навчання моделей матричної факторизації. У цьому алгоритмі мінімізується функція втрат, але оновлення параметрів

відбувається для кожної окремої пари з певним кроком навчання. Перевагами цього алгоритму є те, що він добре працює з дуже розрідженими даними, а також дозволяє додавати нові оцінки без повного перенавчання. Серед недоліків можна виділити складність паралелізації покрокового навчання без втрат розбіжності, а також істотний вплив налаштування темпу навчання і кількості епох на кінцевий результат [35].

У традиційному матричному підході рекомендаційний процес описує лише взаємодію користувач-об’єкт. Однак у туристичному домені релевантність пропозиції істотно залежить від контексту: пори року, погодних умов, дня тижня, бюджету подорожі, складу групи тощо. Контекст-обізнані системи (Context-Aware Recommender Systems, CARS) безпосередньо моделюють ці фактори, підвищуючи точність і різноманітність рекомендацій [36]. Класифікація підходів представлена в таблиці 2.1.

Таблиця 2.1 – Класифікація підходів

Підхід	Ідея
Prefiltering	Фільтрувати матрицю R за релевантним контекстом, а потім застосувати звичайний CF/CB
Post-filtering	Спершу побудувати список без контексту, потім переналаштувати ранжування
Contextual modeling	Інтегрувати контекст прямо у функцію передбачення

Перевагами такої фільтрації безумовно є підвищення точності й адаптація до сезонних та погодних коливань. Проте такий підхід має свої недоліки: складніша персоналізація, коли контекст недоступний; потреба «чистих» даних; складніша збірка логів; вибух розмірності при великій кількості контекстів.

Наступним методом є фільтрація орієнована на контент. Цей вид фільтрації ґрунтується на припущені, що користувачеві подобаються об’єкти, схожі на ті, що він позитивно оцінив раніше. В цьому підході формується вектор ознак туру, а також вектор профілю користувача, який обчислюється алгоритмом Rocchio за формулою 2.5[37].

$$u = a\frac{1}{|P|}\sum_{x\epsilon P} x - \beta\frac{1}{|N|}\sum_{x\epsilon N} x, \tag{2.5}$$

де P – множина позитивних турів;

N – множина негативних турів;

α, β – вагові коефіцієнти.

Перевагами цього методу є прозорість логіки і відсутність «холодного старту». Недоліками такої фільтрації виступають: слабке різноманіття, відсутність «соціального» ефекту (інформації отриманої з поведінки інших користувачів) та ризик перерокомендацій.

Останнім підходом який буде розглянуто є гібридизація. Це стратегія, що поєднує принаймні два різні підходи з метою компенсувати недоліки кожного з них і підвищити загальну релевантність рекомендацій. Можливі типи поєднання представлено в таблиці 2.2.

Таблиця 2.2 – Гібридні підходи

Тип поєднання	Суть
Weighted	Лінійна або нелінійна комбінація скорів CF і CB
Switching	Система вибирає алгоритм залежно від ситуації
Feature augmentation	Вихід першого модуля стає вхідною ознакою другого
Cascade	Алгоритм A фільтрує, алгоритм B переранжовує Top-N

Продовження таблиці 2.2

Тип поєднання	Суть
Meta-level	Повний профіль моделі A слугує даними для B
Mixed	Списки двох систем об’єднуються без агрегування балів
Cross-filtering	Рекомендація користувач-X як фільтр для користувач-Y

Серед цих поєднань зважений гібрид є найуважнішою схемою, в якій використовуються вектори матричної факторизації, контент-профілі Racchio, контекстні ознаки і ваги, що оптимізуються градієнтним бустингом (LightGBM). Дослід OAG-2024 показав, що поєднання CF і CB має значно кращий результат в порівнянні з чистим використанням цих методів.

Якщо говорити про використання на великих платформах таких як Booking, то звісно там використовується гібридний підхід, який до того ще розширюється ре-ранкерами (бустинговими моделями).

2.4 Алгоритми пошуку та ранжування

Побудова списку релевантних туристичних пропозицій починається з локального пошуку схожості, переходить до швидкої ймовірнісної фільтрації і завершується глибоким ансамблевим ранжуванням. Така триступенева схема дозволяє поєднати інтуїтивну простоту, оперативність та високу точність рекомендацій.

2.4.1 K-найближчих сусідів

Серед популярних алгоритмів пошуку схожості можна виділити алгоритм найближчих сусідів (K-NN). Це алгоритм машинного навчання, що використовується для класифікації та регресії. Він належить до методів

«лінивого навчання», тобто, алгоритм не будує модель на етапі тренування, а зберігає всі тренувальні дані і здійснює обчислення лише під час запиту нового зразка. Основна ідея полягає в тому, щоб знайти k найближчих сусідів до нового зразка і прийняти рішення на основі їхніх значень цільової функції. Існує кілька різновидів цього алгоритму.

Один з них – k-NN, зважений за відстанню до сусідів. В цьому випадку алгоритм враховує не лише кількість сусідів, але і їх відстань до тестового зразка. При цьому сусіди, які знаходяться ближче до тестового зразка, мають більший вплив.

Другий – це k-NN, зважений за атрибутами. В цьому варіанті реалізації алгоритму враховується важливість різних атрибутів. Це може бути корисно, коли деякі атрибути є більш значущими для прогнозування, ніж інші. Ваги для атрибутів можуть бути визначені за допомогою різних методів, таких як інформаційна ентропія або методи оцінки важливості атрибутів в рамках дерев рішень.

Серед плюсів цього алгоритму можна виділити просту реалізацію, відсутність припущень про розподіл даних, а також якісну роботу з нелінійними розподіленнями між класами. Проте значними недоліками цього алгоритму є чутливість до нерелевантних або шумних атрибутів, великі обчислювальні витрати при великій кількості даних, а також постійне зберігання всього тренувального набору даних. Також при великій кількості даних з’являється так зване прокляття розмірності. Це прокляття поводує розрідження даних, збільшення обчислювальної складності та, відповідно, погіршення продуктивності алгоритму.

2.4.2 Naïve Bayes

Це простий ймовірнісний класифікатор, який базується на застосуванні теореми Байєса з припущенням про незалежність атрибутів. Цей алгоритм також має різновиди, зокрема: Gaussian Naïve Bayes, який

використовує нормальний розподіл даних і використовується для числових атрибутів; Bernoulli Naïve Bayes, який використовується для бінарних даних і кожен атрибут має значення 0 або 1; Multinomial Naïve Bayes, який використовується для дискретних даних з більш ніж двома значеннями, часто застосовується для текстової класифікації.

Байєсівський алгоритм використовують тоді, коли набір даних є досить великим, оскільки зростання кількості даних допомагає точніше оцінити ймовірність. Якщо ж мова йде про мультимодальні дані, то використовують гібридний підхід, в якому, наприклад використовують Gaussian Naïve Bayes для числових даних і Bernoulli Naïve Bayes для бінарних даних. Або ж інший підхід – предобробка даних. В цьому випадку категоріальні дані кодуються в числові форми і нормалізуються числові атрибути.

Як і більшість алгоритмів, цей класифікатор страждає від «прокляття розмірності», але не так сильно, оскільки він використовує ймовірнісні моделі, які не залежать від просторового розподілу даних у високовимірному просторі. Однак, якщо кількість атрибутів значно збільшується, припущення про незалежність атрибутів може стати менш точним, що може негативно вплинути на результати класифікації.

З переваг цього методу можна відзначити простоту реалізації та високу швидкість тренування та прогнозування, якісну роботу з великими наборами даних, а також відносно не великі витрати пам’яті, оскільки алгоритм не зберігає всі тренувальні дані, а залишає в пам’яті лише підсумкові ймовірності.

Серед недоліків можна відзначити швидку втрату актуальності без перенавчання (в динамічному ринку туризму це досить суттєвий недолік), неможливість моделювати взаємодії та припущення умовної незалежності ознак. Не дивлячись на описані недоліки, у тесті з 50 тис. бронювань Naïve Bayes відсік 80 % нерелевантних кандидатів, зберігши 95 % «золотих» турів, і додав лише ≈ 2 мс latency на запит.

Серед альтернатив можна виділити:

− логістичну регресію, яка може краще працювати з числовими даними і корельованими атрибутами;

− дерева рішень, які добре підходять для даних з нелінійними взаємозв'язками між атрибутами;

− підтримкові векторні машини (SVM), які є ефективними для високовимірних даних і складних розподілів;

− нейронні мережі, що підходять для великих і складних наборів даних з багатьма атрибутами.

2.4.3 Бустинг

Бустинг (з англ. «boosting» – підсилювання) – це процес послідовного навчання великої кількості моделей. Кожна модель навчається на помилках попередньої моделі.

Бустинг є одним із найпотужніших підходів в машинному навчанні, який дозволяє комбінувати декілька слабких моделей для створення однієї сильної моделі. Цей метод базується на ітеративному навчанні моделей, де кожна наступна модель фокусується на помилках попередньої, що призводить до значного покращення точності кінцевої моделі [38]. Незважаючи на свою ефективність, різні варіанти бустингу мають свої особливості, переваги та недоліки.

AdaBoost, CatBoost, Gradient Boosting, LightGBM і XGBoost є найпопулярнішими реалізаціями бустингу, кожна з яких має свою специфіку. AdaBoost, наприклад, простий у реалізації та інтерпретації, але може страждати від надмірної чутливості до шуму в даних. CatBoost, у свою чергу, спеціалізується на обробці категоріальних даних без необхідності їх попереднього кодування.

Gradient Boosting демонструє високу точність, але є ресурсоємним з точки зору часу навчання. LightGBM забезпечує високу продуктивність на

великих наборах даних, тоді як XGBoost – оптимізований для швидкості та ефективності.

Оскільки в гібридних моделях найчастіше використовується LightGBM, то його і буде розглянуто більш детально.

Light GBM – це фреймворк, заснований на техніці дерев рішень, який можна використовувати для ранжування, класифікації та інших додатків машинного навчання [39]. У цьому варіанті бустингу створюються дерева рішень, які ростуть по листах, що означає, що за заданої умови розділяється тільки один лист, залежно від посилення.

Оскільки метод заснований на алгоритмах дерев, він розбиває дерево на частини на основі найкращого збігу, на відміну від інших методів бустингу, які розбивають дерево за рівнем або глибиною, а не за листками. Як наслідок, підхід за листками забезпечує меншу кількість втрат, ніж метод за рівнями, що призводить до значно вищої точності, ніж будь-яка з існуючих стратегій бустингу.

Дерева з великим числом листків іноді можуть перенавчатися, особливо з невеликими наборами даних. Обмеження глибини дерева може допомогти уникнути перенавчання. Розщеплення вздовж листка також збільшує складність і може призвести до надмірної підгонки; однак цього можна уникнути, ввівши опцію максимальної глибини, яка задає глибину, на яку буде виконуватися розщеплення.

Одностороння вибірка на основі градієнта (Gradient-Based One-Side Sampling – GOSS) використовується для вибірки набору даних у LightGBM. GOSS привласнює точкам даних з великими градієнтами під час розрахунку посилення вищу вагу. У цьому методі екземпляри, які недостатньо використовувалися для навчання, роблять більший внесок. Точки даних з меншими градієнтами видаляються випадковим чином, деякі з них зберігаються для підтримки точності. Цей метод зазвичай кращий за випадкову вибірку за тієї ж частоти вибірки.

Переваги та недоліки цього методу представлені у таблиці 2.3.

Таблиця 2.3 – Переваги та недоліки LightGBM.

Аспект	Переваги	Недоліки
Швидкість навчання	Комбінація leaf-wise growth і процедур GOSS та EFB (Exclusive Feature Bundling) зменшує обчислювальну складність і прискорює тренування у 10–20 разів порівняно з XGBoost на великих даних.	У невеликих вибірках агресивне leaf-wise розростання схильне до перенавчання; щоб компенсувати, доводиться жорстко обмежувати максимальну глибину або застосовувати сильну регуляризацію.
Ефективність використання пам’яті	Метод EFB дає змогу об’єднувати взаємно виключні бінаризовані ознаки в «пакети», що скорочує обсяг RAM на 20–30 % без втрати інформації.	За великої кількості висококардинальних категорій (більше 10^4) ефективність пакування істотно зменшується, а обсяг RAM зростає.
Обробка категорійних ознак	Вбудований алгоритм оптимального розщеплення для категорійних полів усуває потребу у зовнішній one-hot-бінаризації, зберігаючи природну ієрархію категорій.	Кількість рівнів категорії обмежена; більші словники потребують попереднього перетворення (hash-кодер, target-encoding), що збільшує час підготовки даних.
Можливість продовження навчання	Реалізована опція continued training, яка дає змогу донавчати вже створену модель на нових даних без повного її перенавчання.	Повноцінного потокового (online) навчання з обробкою одиничних записів алгоритм не підтримує; отже, алгоритм потребує періодичного донавчання.
Гіперпараметрична гнучкість	Велика кількість налаштувань дозволяє тонко адаптувати модель під різні задачі.	Висока чутливість до вибору гіперпараметрів ускладнює ручний пошук оптимальних значень.

LightGBM демонструє виняткову швидкодію та високу точність на табличних, частково розріджених даних, завдяки чому став основним ранжувальним інструментом у багатьох промислових рекомендаційних системах. Натомість модель вимоглива до налаштувань глибини та регуляризації, схильна до перенавчання на малих вибірках і має функціональні обмеження у GPU-режимі та стрімовому навчанні.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Архітектурна схема

Як вже згадувалось, основою веб-застосунку є мова програмування C# і архітектурної моделі MVC. Ця модель дозволяє розділити проект на частини, що спрощує розробку та орієнтування під час реалізації проекту. Такий ефект досягається за допомогою логічного поділу на рівні.

Саме через такий поділ було обрано модель MVC на базі ASP .NET Core, а не популярному зараз Python. В середовищі ASP .NET Core ця модель вимагає жорсткої типізації розподілу обов'язків, що полегшує реалізацію та перевірку автоматичних UNIT-тестів. В свою чергу Python теж дає можливість подібного розподілення, проте не забезпечує такого рівня статичної перевірки.

Також варто зазначити, що в ASP .NET Core нативно включені необхідні для реалізації компоненти:

- модель Identity, яка надає готову реалізацію реєстрації;
- ML.NET, який дозволяє виконувати ONNX-інференс без додаткового мовного «моста».

В свою чергу в Python інтеграція глибокого навчання з веб-фреймворком вимагає окремого Gunicorn/uWSGI-процес, додаткового REST-шару та ручної синхронізації токенів, що значно ускладнює реалізацію.

Сама модель виступає каркасом веб-застосунку і відповідає за структурну логіку, оскільки вимагає реалізації моделей, контролерів та подань. Моделі в цьому випадку відповідають за структуру даних та логіку бізнес-об'єктів. Контролери мають за ціль обробку запитів, логіку обробки даних і передачу їх представлення. Подання слугують шаблоном для інтерфейсу користувача. На рисунку 3.1 представлено основу веб-застосунку на базі ASP .NET Core MVC.

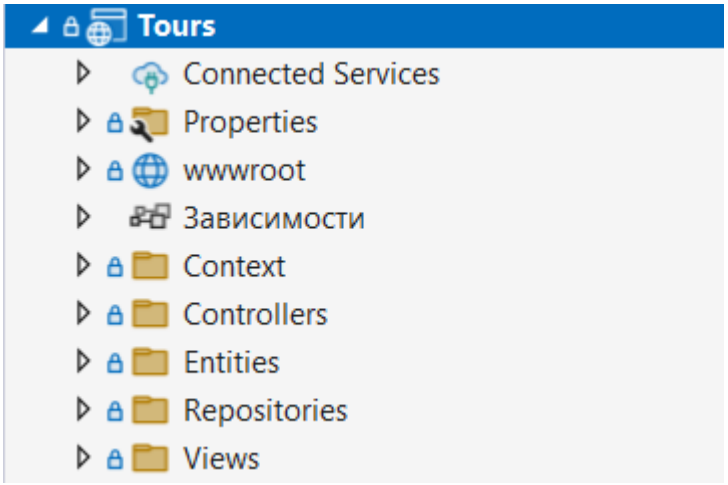

Рисунок 3.1 – Структура на базі моделі MVC

Цей шар є оболонкою застосунку, як вже було згадано, тож реалізовані в цій частині контролери відповідають за прийняття запитів, ініціалізацію валідації моделей і передачу даних до Razor-подань, які в свою чергу формують інтерфейс, який буде розглянуто більш детально далі.

Якщо мова йде про наступний шар проекту, то тут знаходяться класи, які інкапсулюють бізнес-процеси. Доступ до цієї частини програми здійснюється через вбудований контейнер залежностей, що дозволяє уникнути прямих зв’язків між цими ж контролерами та більш глибокою внутрішньою логікою.

До цієї логіки відноситься робота ML-моделей та роботу цих моделей з базою даних, яка реалізована на базі Microsoft SQL Server. Доступ до бази даних реалізовано з використанням Entity Framework Core, контекст якого гарантує транзакційну цілісність між таблицями.

ML-ядро представлене двома синглтон компонентами. Перший з цих компонентів реалізує інференс моделі Ukr-RoBERTa у форматі ONNX. Другий представляє собою каскад для створення персоналізованих рекомендації. Завдяки тому, що всі ці складові розділені в моделі MVC, то всі ML-операції виконуються в тому ж пулі потоків, що й веб-логіка. Разом

з цим і відсутністю мережевих викликів сумарний час відклику програми лишається досить швидким.

Як вже було згадано, для роботи з даними було створено базу даних, а для роботи з таблицями цієї бази даних використано Entity Framework Core. Серед цих таблиць головними виділено:

- таблицю турів;
- таблицю користувачів;
- таблицю визначення ролей;
- таблицю станів;
- таблицю текстових відгуків.

Разом з рештою додаткових таблиць, БД забезпечує повний набір даних для функціонування інтелектуальної туристичної системи. Завдяки чітким зовнішнім ключам та індексам такий формат гарантує цілісність системи та продуктивні вибірки, а групування сутностей дозволяє працювати з даними на різних рівнях та залишає можливість для подальшого розподілу сховища у випадку змін у підході реалізації.

3.2 Автентифікація та підсистема профілів

Для роботи з веб-застосунком, як і для більшості платформ та додатків необхідно створити власний обліковий запис. В даному застосунку процес автентифікації та особистих профілів реалізовано за допомогою ASP .NET Core Identity. Це дозволяє перекрити процес реєстрації, подальшого входу до свого кабінету, а також керування особистими даними без використання сторонніх сервісів. Реєстрація з використанням Identity включає в себе підтвердження електронної пошти, після якого процес завершується викликом відповідного ендпоїнта контролера акаунтів та створюється токен JSON, який зберігає необхідні дані про користувача. Завдяки цьому токену в додатку мінімізується ризик міжсайтового скриптингу.

Як згадувалось раніше одними з головних таблиць бази даних є користувачі та тури. Щоб поєднати ці дві сутності було створено проміжну таблицю турів користувача, з відношенням між головними таблицями багато до багатьох. В цій проміжній таблиці окрім ключа туру та користувача знаходиться атрибут статусу, тобто реакція користувача на тур: додано до списку бажань, додано до улюблених турів або ж користувач вже відвідав цей тур, – та атрибут оцінки для цього туру. Завдяки сервісу профілю інкапсулювано методи додавання до списку бажань, відмітки «відвідано» та відзначено зірочкою. Оскільки всі операції зі списками виконуються в одній транзакції Entity Framework Core, мережеві затримки мінімізуються, а взаємодія клієнта та веб-застосунку спрощується.

Сторінка профілю користувача реалізована з використанням Razor Pages. На сторінці свого облікового запису відображаються описані вище списки, рекомендації та швидкі дії над турами. Будь-яка виконана дія з туром оновлює проміжну матрицю, а збережені текстові відгуки відразу аналізуються ML-моделлю. Ці зміни дозволяють показувати на сторінці профілю рекомендації, що відповідають змінам сприйняття користувача.

Таким чином наявність цих компонентів та реалізація таким чином відокремлює бізнес-функції від безпекових, при цьому виконуються вимоги цілісності, доступності та конфіденційності даних.

3.3 Оцінки та текстові відгуки

Наступною таблицею, яка зв’язує головні таблиці користувачів та турів є Review (огляд). В цій сутності виділено окрім ключів до відповідного користувача та туру числову оцінку туру, текстовий коментар, мітку часу та оцінка полярності. Пара ключ туру і ключ користувача формують складений ключ, що дозволяє зробити коментар унікальним. Таким чином вдається уникнути множинного оцінювання.

Як було показано в розділі 3.2, для кожного туру є можливість додавання туру до якогось зі списків, вибору оцінки та написання невеликого текстового коментаря, в якому користувач описує свої враження від туру. Після того, як користувач заповнить ці поля і збереже цю оцінку, відповідний контролер обробляє зміни і повертає до таблиці значення полярності разом з рештою атрибутів, що були описанні раніше. Оскільки цей процес виконується в одній транзакції, то дані будуть збережені коректно та не розійдуться.

Після успішного запису відгуку оновлюються дані про рейтинг туру, кількість оглядів і середню оцінку туру. Всі ці оновлення безумовно вливають на персоналізовані рекомендації, тож важливим аспектом в цьому питанні є своєчасне оновлення моделі. Тому автоматичне оновлення рекомендацій відбувається раз на 15 хвилин. Під час виконання процесу оновлення оцінка туру користувачем та середня оцінка туру беруться в якості вагових коефіцієнтів при навчанні латентних факторів ALS та відбувається переоцінка моделі LambdaMART. Завдяки такому підходу зміна рекомендацій після оновлення оцінок користувачем для туру або турів показується в списку пропозицій протягом одного циклу оновлення.

Для уникнення проблем з наявністю необхідних даних використовуються валідатори Data Annotations. Вони вказують на обов'язковість оцінки туру, коментаря та його максимальну довжину.

Звісно світ не ідеальний і можливі різні ситуації. Тому передбачено запобігання аномаліям, шляхом реалізації автофільтру. Задачею цього фільтру є перевірка оцінок на коректність, а також перевірка текстового коментаря на наявність ненормативної лексики.

Тому у випадку коли оцінка є досить високою, а результат тонального аналізу при цьому є досить низьким або ж навпаки, запис буде позначено прапорцем, який сигналізує про появу аномалії. В такій ситуації необхідна ручна перевірка запису.

Для цього всі позначені прапорцями записи виводяться адміністратору в спеціальну секцію на його панелі керування, для швидкої реакції та вирішення проблеми підозрілих записів.

Таким чином описане рішення дозволяє інтегрувати процес оцінення турів в одну суцільну операцію, яка дозволяє моделі рекомендацій негайно реагувати на зміни. Також перевагою такої реалізації є узгодженість даних і зменшення часу відгуку програми.

3.4 Рекомендаційна модель

Для створення персоналізованих рекомендацій, які згадувались раніше необхідним елементом застосунку є рекомендаційна модель. Це ML-модель, яка на основі попереднього досвіду знаходить семантично близькі тури за певними ознаками. У випадку даного веб-застосунку цими ознаками є оцінка туру, тональності відгуку та взаємодія користувача з туром.

Ця модель реалізована у вигляді кількарівневого алгоритму який представлений компонентом RecEngine. На початку роботи алгоритму формується матриця взаємодій користувача з туром. Значення цієї матриці вказують на вагу конкретної дії користувача. Тобто якщо тур було додано до списку бажань або відзначено як відвіданий, то ці дії отримають відповідний ваговий коефіцієнт, який буде занесено до матриці. Також до матриці вноситься оцінка туру. Наступним кроком є матрична факторизація методом ALS, після виконання якої повертається вектор користувача та туру. Для того щоб отримати перший топ потенційних турів береться скалярний добуток цих векторів.

Після знаходження початкового набору турів алгоритм починає наступний рівень обробки. На цьому етапі кожен опис туру кодується BERT-ембердингом. Після чого методом K-найближчих сусідів серед закодованих описів відбираються 5 найближчих. Для такого відбору

використовується косинусна відстань, яка краще підходить для високовимірних даних.

Останнім кроком цього алгоритму є ранжування отриманих результатів за допомогою LightGBM. Для цього алгоритм отримує в якості вхідних даних попередньо отримані результати: отриманий набір потенційних кандидатів, косинусну відстань, коефіцієнти тональності та індикатори статусу.

На підставі цих даних алгоритм бустингу формує лямбда-градієнти, необхідні для визначення впливу перестановки елементів на позиційну метрику. Після отримання цих градієнтів відбувається побудова ансамблю дерев рішень.

Ансамбль класифікаторів в загальному розумінні це набір класифікаторів, рішення яких формуються для класифікації нових прикладів Бустинг є одним з методів побудови ансамблів, в якому кожен новий класифікатор є більш точним, оскільки навчається на помилках попередніх і надає більшу вагу зразкам, які важко класифікувати. Процес отримання результатів виконання кожного з класифікаторів відбувався шляхом зваженого голосування. В цьому випадку кожен класифікатор отримує вагу відповідну точності класифікації. В контексті даної реалізації кожен класифікатор відповідає певному потенційному туру. Іншими словами кожен тур отримує оцінку подібності до турів, до який шукаються подібні. На основі цих оцінок формується відсортований за спаданням список, який і є кінцевим результатом виконання рекомендаційного алгоритму. В отриманому списку перші п’ять турів з всієї кількості потенційно подібних будуть мати найбільшу оцінку релевантоності, тобто будуть найбільш наближеними до того, що сподобалось користувачу.

Для коректної роботи алгоритму його необхідно натренувати. Для цього існують відкриті репозиторії з відкритими даними, створені для навчання різного типу ML-моделей. Одним з таких репозиторіїв є UCI [40]. Для навчання моделі з цього репозиторію було обрано відкритий набір

даних Amazon Commerce Reviews [41]. В цьому наборі містяться поля ключів для користувачів, продуктів, оцінок та текстових відгуків. Набір включає в себе більше ніж півмільйона записів, що дозволило навчити та протестувати модель на різному співвідношенні навчальної та тестової вибірок. В результаті тестування модель показала близько 80% коректно класифікованих записів.

Додатково було проведено донавчання моделі на наборі даних Sentiment Labelled Sentences, який містить речення, позначені як позитивно або негативно забарвлені [42]. Це було зроблено для більш якісного розуміння відгуків користувачів.

3.5 Інтерфейс користувача

Інтерфейс користувача в середовищі ASP .NET Core MVC було реалізовано за допомогою Razor Pages з використанням Bootstrap і CSS.

Усі сторінки наслідують загальний макет, який включає в себе меню, підключення стилів і компоненти динамічних сповіщень.

Меню виконане у формі навігаційної панелі, яка поєднує всі основні розділи. А для правильного вирівнювання поля форм обгорнуті в Bootstrap-сітку.

На головну сторінку винесено список можливих турів, які на даному етапі користувач додає самостійно. Також на цій сторінці знаходяться фільтри для самостійного пошуку турів, що відповідають вимогам користувача. Вигляд цієї частини представлено на рисунку 3.2.

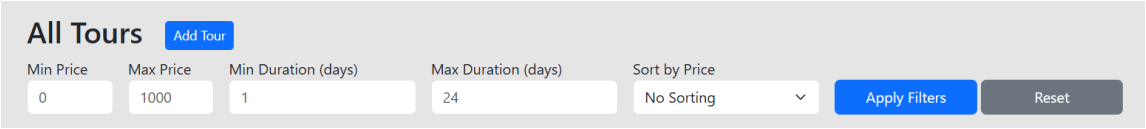

Рисунок 3.2 – Вигляд логічних операцій з турами

Під час додавання туру власноруч, необхідно ввести назву цього туру, його ціну, тривалість, а також опис цього туру. На рисунку 3.3 представлено вигляд додавання туру.

Рисунок 3.3 – Вигляд додавання туру

Кожен тур представлений у вигляді картки з картинкою, назвою, необхідними атрибутами та кнопками керування статусами. Приклад такої картки представлено на рисунку 3.4.

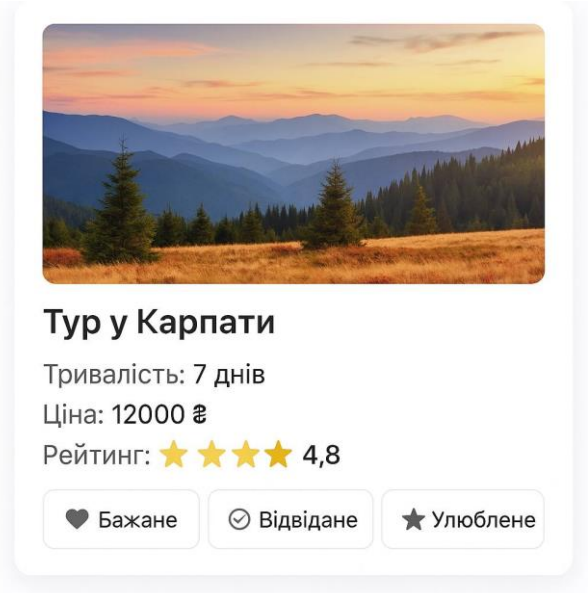

Рисунок 3.4 – Приклад картки туру

На сторінці профілю представлені тури, які були збережені користувачем до певної закладки. Серед цих закладок знаходяться тури, які користувач хотів би відвідати, які є його улюбленими та ті, які він вже встиг відвідати. До кожного туру додані кнопки для перенесення туру до тої чи іншої категорії, або ж видаленню з тої, де він зараз знаходиться. Ці кнопки створюють асинхронні виклики до відповідного контролера, оновлюючи DOM без перезавантаження. Приклад картки туру в особистому кабінеті представлено на рисунку 3.5.

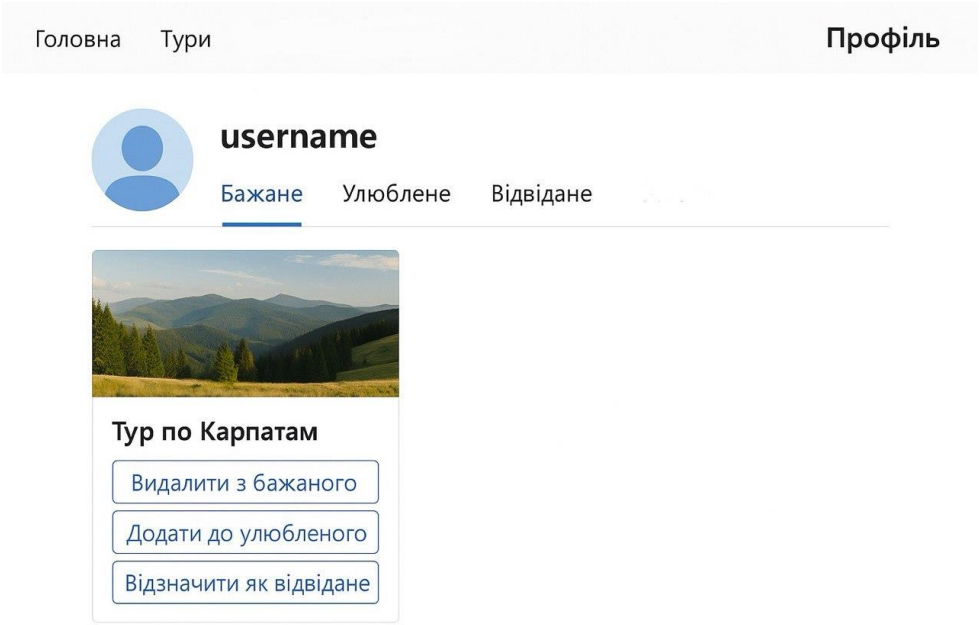

Рисунок 3.5 – Картка туру в особистому кабінеті

Як вже згадувалось раніше, до кожного туру користувач залишає оцінку туру та текстовий коментар. Перейшовши на сторінку відвіданого туру окрім базової інформації на початку буде представлено оцінку туру, коментар та сформовану оцінку цього коментаря, яка буде підсвічуватись відповідним кольором. Якщо тональність коментаря була негативною, то відповідно буде використано червоний колір, якщо ж тональність буде оцінена як позитивна, тоді буде використано зелений колір. Приклад реалізації представлено на рисунку 3.6.

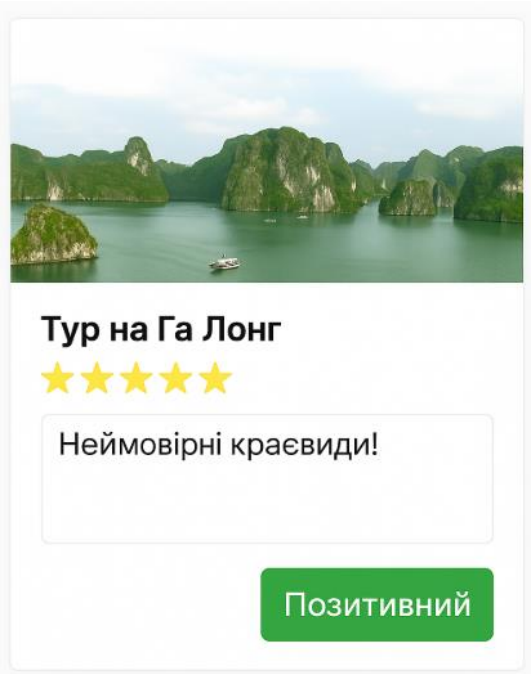

Рисунок 3.6 – Приклад представлення оцінки

Головними критеріями для створення візуального інтерфейсу були простота, зрозумілість та доступність. Таким чином представлений інтерфейс дозволяє швидко зрозуміти, що необхідно зробити на тому чи іншому етапі, переглянути відповідні результати, а використання відповідності кольорів та візуальності оцінок дозволяє без зайвих пошуків пригадати, які враження були від туру.

Також інтерфейс дозволяє швидко переключатись між ключовими розділами за допомогою уміщених на сторінках клікабельних підписів. А за допомогою лаконічності вдалося мінімізувати перевантаження зайвою інформацією.

3.6 Можливості розширення

Враховуючи всі наведені вище описи складається цілком приємна картина інтелектуальної рекомендаційної системи. Проте така реалізація окрім наведених раніше переваг кожного з використаних елементів є цілком

вагомі недоліки. До цих недоліків можна віднести одну з переваг – структуру проєкту. Незважаючи на те, що використана монолітна реалізація дозволяє швидкий запуск програми, зменшення можливих проблем роботи з даними та швидкість передачі проміжних результатів між різними компонентами програми, така структура вимагатиме значних змін, щоб стати повноцінною системою для загального використання.

Тому в перспективі необхідно відійти від використання моноліту до мікро сервісного підходу. Головним завданням такого переходу є винесення до окремих компонентів моноліту в самостійні сервіси.

До таких компонентів відносяться перш за все ML-моделі. При зростаючій кількості користувачів будуть збільшуватись і кількість даних, відгуків і оцінок, які необхідно постійно оновлювати, перенавчатись та банально виконувати всі дії на великих кількостях потоків. Якщо все це буде відбуватись в монолітному додатку, то час відгуку системи буде постійно збільшуватись разом із ресурсними витратами. Винесення цих моделей в окремі сервіси дозволить використати GPU-прискорення для LightGBM. Також з'явиться можливість горизонтального масштабування саме частини машинного навчання відповідно до навантажень. Цей процес виконується у середовищі Kubernetes за використанням механізму Horizontal Pod Autoscaler (HPA). При такому підході навантаження цієї частини програми завжди регульоване, оскільки HPA періодично перевіряє необхідні метрики і в залежності від отриманих результатів або збільшує кількість контейнерів, або видаляє зайві. Таким чином при великому навантаженні модулю стабільність продуктивної роботи і мінімальні затримки при відповіді гарантовані. При цьому якщо навантаження буде знижуватись, то буде зменшуватись кількість працюючих контейнерів, що відповідно зменшить витрати обчислювальних ресурсів. Додатковою перевагою винесення окремо ML-моделей є можливість їх зміни без необхідності повної перебудови застосунку загалом.

Наступним компонентом, який можна винести окремо є модуль, який відповідає за автентифікацію та профілі клієнтів. Цей крок забезпечить ізоляцію компонентів безпеки від бізнес-логіки. Перевагами такого відокремлення є винесення критичних даних в окрему базу даних, що дозволить зберегти інформацію кожного користувача захищеною, оскільки доступ до цієї частини буде обмеженим. Створення такої бази дозволить змінювати за потреби політику автентифікації або моделі користувача незалежно від основного фронтенду. Тобто додавання нових можливостей автентифікації таких як біометричні дані можна буде реалізувати без перевипуску програми. Також значною перевагою є те, що при будь-яких проблемах, таких як відмова модулю або ж реліз оновлення не призведе до повної зупинки інших функціональних частин, що відбудеться в подібній ситуації при монолітній реалізації.

Як вже було згадано необхідним кроком є створення окремої бази даних для зберігання критичних даних. Це також є частиною масштабування веб-застосунку. Розділення реляційної бази даних, яка при монолітній реалізації містила всі сутності, дозволить запобігти ряду проблем, що виникнуть при зростанні обсягів даних, що зберігатимуться та навантаженню на систему. Серед цих проблем можна виділити наступні:

– ускладнення міграції даних;
– нерівномірне навантаження на систему;
– уповільнення запитів через блокування великих транзакцій;
– загальний спад продуктивності системи.

Окрім бази даних, що відповідає за збереження критичних даних, необхідно теж відокремити дані, які безпосередньо використовуються для роботи системи. До цієї бази даних будуть занесені бізнес-дані та дані про дії користувачів, тобто інформації про тури, можливі замовлення, відгуки користувачів тощо.

Окремо теж слід винести дані, які використовуються для роботи ML-моделей.

Подібне розподілення даних дозволить розподілити навантаження різних виконуваних процесів і масштабувати дані за різними характеристиками незалежно від інших. Ще одним вагомим плюсом розподілення даних є прискорення міграції, оскільки зміни в турах або відгуків будуть занесені лише до відповідної бази, при цьому не зачіпаючи таблицю з критичними даними або ж аналітичних потоків. Що в'яжиться теж з мінімізацією несанкціонованого доступу до прихованої від загального доступу інформації.

Очевидно, що розбиття даних на окремі бази даних дозволяє створити більш стійку до збоїв систему. Так у випадку виникнення проблем при використанні одної з таблиць, решта елементів програми залишаться доступними. А от при існуванню загальної реляційної бази даних поява проблем на будь-якому етапі виконання призведе до загальної зупинки програми.

Ще одним важливим кроком для розвитку рекомендаційної системи стане інтеграція існуючих систем для бронювання турів. Метою цього кроку є розширення функціональності програми, а також більше можливостей для пошуку найкращих рекомендацій. Інтеграція існуючих систем буде винесена окремим модулем, що дозволить додати подібний функціонал без цілісного перезапуску веб-застосунку.

У випадку інтеграції існуючих пропоцизій, в своїх рекомендаціях користувач буде отримувати тури, які він зможе без зайвих зусиль зберегти або ж одразу забронювати. Таке розширення дозволить зменшити час на додавання турів до відповідних категорій, пошук дистрибутора та дозволить оптимізувати роботу застосунку на основі реального попиту.

Таким чином можна розглянути безліч можливих варіантів для розширення застосунку, проте всі існуючі варіанти описати і виявити точно не вдасться. Описані в цьому підрозділі кроки є одними з найважливіших для запевнення більш якісної роботи з інтелектуальною системою в майбутньому. Звісно реалізація кожного з них повинна відбуватися

поступово, зі збереженням сумісності і коректності роботи кожного з модулів, а також мінімізації затримок під час роботи веб-застосунку.

В даному підрозділі було розглянуто можливі варіанти розвитку веб-застосунку, а також описано чим така реалізація буде кращою від монолітної. Проте остання теж має свої переваги.

Одною з переваг використання монолітної реалізації є відсутність затримки передавання даних між модулями. В той час як в мікросервісному підході існують мережеві виклики, які додають час очікування передачі даних. Якщо говорити про мережеві виклики, то ту ще одна перевага монолітного підходу – відсутність зовнішніх URL. Їх наявність у мікросервісному підході може викликати проблеми підключення необхідних модулів, через що система буде працювати некоректно. При цьому мікросервісний підхід зменшує кількість блокувань запитів користувачів, оскільки в ньому всі дані поділені на відповідь БД, що дозволяє рівномірно розділити ресурси, або ж як згадувалось раніше створити нові контейнери для роботи при великому навантаженні.

Як можна побачити кожен підхід має свої переваги та недоліки. Для того, щоб мінімізувати недоліки обох підходів та посилити позитивні аспекти кожного з них оптимальним варіантом буде як раз описаний в попередніх розділах гібридний підхід.

В контексті даного веб-застосунку вибір такої реалізації має на меті збереження MVC підходу, але при цьому винести важливі компоненти за його межі.

Серед переваг такої реалізації можна виділити:

– баланс між швидкістю розробки та гнучкості;
– можливість покрокового масштабування;
– чіткий поділ.

Моноліт ASP .NET Core MVC дозволяє структурувати всі компоненти програми, в той час як винесені окремі елементи в окремі служби дозволить вдосконалювати систему без затримок релізу.

Якщо говорити про покрокове масштабування та можливості масштабування, то такий підхід теж дозволяє оновлювати систему без кардинальних змін в інтерфейсі програми,

Поділ даних, про який вже згадувалось в цьому підрозділі та поділ обов’язків між структурними елементами, які відповідають за обчислення так само як і попередні дії дозволяє допрацьовувати ці структурні елемент без змін у фронтенді. До цього ж варто віднести простоту підтримки та оновлень. Найбільше уваги оновленням вимагають системи оцінки тональності та рекомендацій. Це ML-моделі, які змінюються дуже стрімко останнім часом. У випадку, коли ці структурні елементи відокремлені один від одного та решти компонентів, то оновлення, перенавчання або донавчання не викличе жодних проблем, оскільки не буде необхідності заморожувати додаток загалом.

Так чином можна точно сказати, що ASP .NET Core MVC відповідає за ясну структуру ключових елементів застосунку, їх поєднання та візуальне представлення. В свою чергу винесення всіх «важких» компонентів – дозволяє отримати приріст в ресурсному використанню, своєчасному і безпроблемному оновленню, а також створює відносно незалежну логіку, яка у випадку проблем не завершить всі можливі процеси, а тільки той, в якому виникла надзвичайна ситуація.

ВИСНОВКИ

Під час виконання кваліфікаційної роботи було проведено аналіз предметної галузі, комплексне дослідження існуючих підходів та методів реалізації подібних систем, а також розроблено веб-застосунок інтелектуальної туристичної системи з використанням технологій ASP .NET Core MVC. Об’єктом дослідження виступив процес надання персоналізованих туристичних рекомендацій на веб-платформах. Предметом дослідження виступило практичне застосування методів аналізу текстових відгуків і рекомендаційних алгоритмів у згаданому вище середовищі.

В результаті аналізу предметної галузі було виявлено, що ринок туризму стрімко розвивається, попит збільшується і вже повернувся до рівня, який був перед пандемією. Згідно з прогнозами, розвиток ринку з роками буде лише рости, тож даний напрямок є досить перспективним з огляду на те, що користувачів тут немало. Також під час аналізу галузі було розглянуто найбільш популярні системи, які складають більшу частину ринку пропозицій та їхній підхід до створення рекомендацій та аналізу дій користувачів.

Розділ з описом методів реалізації включає в себе новітні алгоритми, що використовуються згаданими популярними системами. Також було наведено можливі підходи реалізації для інтелектуальних систем загалом.

Практична частина включає в себе реалізацію веб-застосунку в монолітному форматі, що на даному етапі життєвого циклу програми дає більше переваг, ніж будь-який інший підхід. До застосунку було інтегровано компоненти машинного навчання серед яких один відповідає за тональний аналіз відгуків, а інший за пошук найбільш відповідних турів, на основі попередніх вподобань користувача. Весь функціонал було уміщено в структурі MVC, що дозволило структурувати елементи програми та відділити функціональні модулі від бізнес-логіки. Використано теж Razor і

Bootstrap для створення простого та водночас зрозумілого інтерфейсу користувача в якому знаходяться всі необхідні елементи для зручного користування програмою.

На основі отриманої реалізації, досвіду та результатів аналізу предметної галузі та підходів до реалізації подібних систем, було створено опис можливої, а місцями навіть необхідної перебудови структури програми. При цьому було враховано переваги та недоліки кожного з підходів і виділено найоптимальніше рішення – гібридний підхід. Така перебудова має на меті розподілення даних на різні бази даних, винесення важливих структурних елементів окремо, при цьому зберігши MVC підхід. Така реалізація в майбутньому дозволить зменшити витрати ресурсів, зменшити ризики «падіння» системи та надасть можливість змінювати функціональні елементи без впливу на інтерфейс.

У підсумку варто теж зазначити, що всі поставлені задачі виконані повністю.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1) LibreTexts. 1.1: Що таке туризм?. *LibreTexts – Ukrayinska*. URL: https://ukrayinska.libretexts.org/Робоча_сила/Гостинність/Вступ_до_туризму_та_гостинності_в_Британській_Колумбія_(Westcott_і_Anderson)_2-е_видання./01:_Історія_та_огляд/TEST_1.1:_Що_таке_туризм? (дата звернення: 20.05.2025).

2) Маджумдар О. Автобусні тури: що треба знати про плюси і мінуси таких поїздок. *РБК-Украина*. URL: https://www.rbc.ua/rus/travel/avtobusni-turi-shcho-treba-znati-plyusi-i-1713361691.html (дата звернення: 20.05.2025).

3) International Tourism to End 2023 Close to 90% of Pre-Pandemic Levels. *UN Tourism | Bringing the world closer*. URL: https://www.unwto.org/news/international-tourism-to-end-2023-close-to-90-of-pre-pandemic-levels (дата звернення: 20.05.2025).

4) Online Travel Market Size, Share, Trends and Forecast by Service Type, Platform, Mode of Booking, Age Group, and Region, 2025-2033 URL: https://www.researchandmarkets.com/reports/5820824 (дата звернення 12.04.2025).

5) Global Online Travel Market Forecast Report 2024-2032 by Type, Service Type, Payment Mode, Gender, Booking Device, Countries and Company Analysis. – *ResearchAndMarkets.com*. URL: https://www.businesswire.com/news/home/20240807190211/en/Global-Online-Travel-Market-Forecast-Report-2024-2032-by-Type-Service-Type-Payment-Mode-Gender-Booking-Device-Countries-and-Company-Analysis---ResearchAndMarkets.com (дата звернення 20.05.2025).

6) Закон України «Про туризм» від 15.09.1995 р. № 324/95-ВР. URL: https://zakon.rada.gov.ua/laws/show/324/95-вр (дата звернення 21.05.2025).

7) Stryzhak O. O., Aldoshyna M. V. http://www.business-inform.net/article/?year=2019&abstract=2019_3_0_162_169&lang=en. *Business Inform*. 2019. Т. 3, № 494. С. 170–175. URL: https://doi.org/10.32983/2222-4459-2019-3-170-175 (дата звернення: 21.05.2025).

8) Booking-com/accommodation-reviews. Datasets at Hugging Face. *Hugging Face – The AI community building the future.* URL: https://huggingface.co/datasets/Booking-com/accommodation-reviews (дата звернення: 21.05.2025).

9) README.md. Booking-com/accommodation-reviews at main. *Hugging Face – The AI community building the future.* URL: https://huggingface.co/datasets/Booking-com/accommodation-reviews/blob/main/README.md (дата звернення: 21.05.2025).

10) 45+ Google Business Profile Statistics: A Must Know in 2025. *RedLocalSEO*. URL: https://www.redlocalseo.com/google-business-profile-statistics/ (дата звернення: 22.05.2025).

11) Hotel reviews: A complete guide | SiteMinder. *SiteMinder*. URL: https://www.siteminder.com/r/hotel-reviews-manage-online-property/ (дата звернення: 22.05.2025).

12) Маркетинг – 2.6. Аналіз даних і результати дослідження. *Google Drive: Sign-in*. URL: https://sites.google.com/site/marketingdistance/тема-2/2-6-аналіз-даних-і-результати-дослідження (дата звернення: 22.05.2025).

13) Pushkar Kumar. Aspect-based Sentiment Analysis using Hierarchical Attention Networks. *Journal of Electrical Systems*. 2024. Vol. 20, no. 11s. P. 1992–1999. URL: https://doi.org/10.52783/jes.7682 (дата звернення: 23.05.2025).

14) Wen T., Xu X. Research on Image Perception of Tourist Destinations Based on the BERT-BiLSTM-CNN-Attention Model. *Sustainability*. 2024. Т. 16, № 8. С. 3464. URL: https://doi.org/10.3390/su16083464 (дата звернення: 23.05.2025).

15) Трансформери в обробці тексту: що варто знати – Блог Cpashka. *Блог Cpashka – Створіть свою Дропшиппінг Платформу або CPA-сітку. 100 безкоштовних заявок при реєстрації*. URL: https://cpashka.biz/blog/transformery-v-obrobtsi-tekstu-shcho-varto-znaty/ (дата звернення: 23.05.2025).

16) Yuan X. Evaluation of rural tourism development level using BERT-enhanced deep learning model and BP algorithm. *Scientific Reports*. 2024. Т. 14, № 1. URL: https://doi.org/10.1038/s41598-024-77444-0 (дата звернення: 23.05.2025).

17) Zero Shot Classify SSTuning XLM R Â· Models Â· Dataloop. *Dataloop | Let the builders build*. URL: https://dataloop.ai/library/model/damo-nlp-sg_zero-shot-classify-sstuning-xlm-r/ (дата звернення: 23.05.2025).

18) BERT-BiLSTM-Attention model for sentiment analysis on Chinese stock reviews / Xiaoyan Li та ін. *Applied Mathematics and Nonlinear Sciences, 9(1) (2024) 1-17*. 2024. Т. 9, № 1. 62P20. (дата звернення: 24.05.2025).

19) Baid, P.; Gupta, A.; Chaplot, N. Sentiment analysis of movie reviews using machine learning techniques. Int. J. Comput. Appl. 2017, 179, 45–49. (дата звернення 24.05.2025).

20) Recommender Systems Handbook / Ricci F., Rokach L., Shapira B., Kantor P.B. – Springer, 2011. – 875 p. (дата звернення: 25.05.2025).

21) Juan Z., Zhang J., Gao M. A multimodal travel route recommendation system leveraging visual Transformers and self-attention mechanisms. *Frontiers in Neurorobotics*. 2024. Vol. 18. URL: https://doi.org/10.3389/fnbot.2024.1439195(дата звернення:25.05.2025).

22) Hybrid recommender system for tourism based on big data and AI: A conceptual framework / K. A. Fararni та ін. *Big Data Mining and Analytics*. 2021. Т. 4, № 1. С. 47–55. URL: https://doi.org/10.26599/bdma.2020.9020015 (дата звернення: 25.05.2025).

23) Booking Holdings (BKNG) Product Insights. URL: https://www.tipranks.com/stocks/bkng/website-traffic (дата звернення: 26.05.2025)

24) Airline Products | Amadeus. *Amadeus | The leading travel technology company*. URL: https://amadeus.com/en/airlines/products/all (дата звернення: 26.05.2025).

25) FareHarbor | Online Booking Software for Tours and Activities. *FareHarbor*. URL: https://fareharbor.com (дата звернення: 27.05.2025).

26) ASP.NET Core MVC. *Microsoft Learn: Build skills that open doors in your career*. URL: https://learn.microsoft.com/ru-ru/aspnet/core/mvc/views/overview?view=aspnetcore-9.0&WT.mc_id=dotnet-35129-website (дата звернення: 28.05.2025).

27) Building Microservices: Designing Fine-Grained Systems. O'Reilly Media, Incorporated, 2020. 250 с. (дата звернення: 28.05.2025).

28) Richardson C. Microservices Patterns: With Examples in Java. Manning Publications Co. LLC, 2018. (дата звернення: 29.05.2025).

29) Manning C. D., Schütze H., Raghavan P. Introduction to Information Retrieval. Cambridge University Press, 2008. (дата звернення: 29.05.2025).

30) 1.9. Naive Bayes. scikit-learn. URL: https://scikit-learn.org/stable/modules/naive_bayes.html (дата звернення: 30.05.2025).

31) Bahdanau D., Cho K., Bengio Y. Neural Machine Translation by Jointly Learning to Align and Translate // Proc. 3rd International Conference on Learning Representations (ICLR), 2015. (дата звернення: 31.05.2025).

32) Devlin J., Chang M.-W., Lee K., Toutanova K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. Proceedings of NAACL-HLT, 2019. (дата звернення: 1.06.2025).

33) Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin, Attention Is All

You Need, Cornell University. URL: https://arxiv.org/abs/1706.03762 (дата звернення: 02.06.2025).

34) PhD E. G. Alternating Least Squares: A Cornerstone in Modern Data Analysis and Recommendation Systems. Medium. URL: https://medium.com/aimonks/alternating-least-squares-a-cornerstone-in-modern-data-analysis-and-recommendation-systems-ceb7310313f9 (дата звернення: 3.06.2025).

35) Mishra M. Stochastic Gradient Descent: A Basic Explanation. Medium. URL: https://mohitmishra786687.medium.com/stochastic-gradient-descent-a-basic-explanation-cbddc63f08e0 (дата звернення: 3.06.2025).

36) Shani G., Gunawardana A. Evaluating Recommendation Systems. Recommender Systems Handbook. Boston, MA, 2010. С. 257–297. URL: https://doi.org/10.1007/978-0-387-85820-3_8 (дата звернення: 4.06.2025).

37) Christopher D. Manning, Prabhakar Raghavan, Hinrich Schütze, Introduction to Information Retrieval. Cambridge University Press, 2008. (дата звернення: 4.06.2025).

38) Understanding Boosting in Machine Learning: A Comprehensive Guide. Medium. URL: https://medium.com/@brijesh_soni/understanding-boosting-in-machine-learning-a-comprehensive-guide-bdeaa1167a6 (дата звернення: 5.06.2025).

39) Довідник по Machine Learning – LightGBM | База знань IT. База знань IT технологій. URL: https://itwiki.dev/data-science/ml-reference/ml-glossary/lightgbm (дата звернення: 6.06.2025).

40) UCI Machine Learning Repository. *UCI Machine Learning Repository*. URL: https://archive.ics.uci.edu (дата звернення: 8.06.2025).

41) Amazon Commerce Reviews. *UCI Machine Learning Repository*. URL: https://archive.ics.uci.edu/dataset/215/amazon+commerce+reviews+set (дата звернення: 10.06.2025).

42) Sentiment Labelled Sentences. *UCI Machine Learning Repository*. URL: https://archive.ics.uci.edu/dataset/331/sentiment+labelled+sentences (дата звернення: 11.06.2025).