Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет комп'ютерних наук (або центр післядипломної освіти, або навчально-науковий центр заочної форми навчання)
(повна назва)

Кафедра програмної інженерії
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

Програмна система для рекомендації та підбору актуальних культурних подій. Back-end.

(тема)

Виконав:
студент 4 курсу, групи ПЗПІ-20-3

Хамінов І.О.
(прізвище, ініціали)

Спеціальність 121 – Інженерія програмного забезпечення
(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма Програмна інженерія
(повна назва освітньої програми)

Керівник доц. кафедри ПІ Побіженко І.О.
(посада, прізвище, ініціали)

Допускається до захисту
Зав. кафедри _____ З.В.Дудар
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет комп'ютерних наук
Кафедра програмної інженерії
Рівень вищої освіти перший (бакалаврський)
Спеціальність 121 – Інженерія програмного забезпечення
Тип програми Освітньо-професійна
Освітня програма Програмна Інженерія
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«___» _____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Хамінову Іллі Олександровичу
(прізвище, ім'я, по батькові)

1. Тема роботи Програмна система для рекомендації та підбору актуальних культурних подій. Back-end.

Затверджена наказом по університету від 04.06. 2024р. № 471 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 14.06.2024

3. Вихідні дані до роботи Розробка програмної системи, яка надає рекомендації та персоналізує культурні заходи, з використанням мови програмування C# та фреймворку ASP.NET 6

4. Перелік питань, що потрібно опрацювати в роботі

Вступ, аналіз предметної галузі, формування вимог до програмної системи, архітектура та проектування програмного забезпечення, опис прийнятих програмних рішень, тестування розробленого програмного забезпечення, висновки, додатки.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз предметної галузі	08.04.2024	*виконано*
2	Створення специфікації ПЗ	12.04.2024	*виконано*
3	Проектування ПЗ	15.04.2024	*виконано*
4	Розробка ПЗ	10.05.2024	*виконано*
5	Тестування ПЗ	30.05.2024	*виконано*
6	Оформлення пояснювальної записки	03.06.2024	*виконано*
7	Підготовка презентації та доповіді	11.06.2024	*виконано*
8	Попередній захист	13.06.2024	*виконано*
9	Нормоконтроль, рецензування	15.06.2024	*виконано*
10	Здача роботи у електронний архів	17.06.2024	*виконано*
11	Допуск до захисту у зав. кафедри	20.06.2024	*виконано*

Дата видачі завдання 8 квітня 2024р.

Студент _____ Хамінов І.О.
(підпис)

Керівник роботи _____ доц. кафедри ПІ Побіженко І.О.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи бакалавра, 68 с., 18 рис., 12 джерел.

СИСТЕМА ДЛЯ РЕКОМЕНДАЦІЇ КУЛЬТУРНИХ ПОДІЙ, ПРОГРАМНА СИСТЕМА, СЕРВЕРНА ЧАСТИНА, API, ASP.NET 6, C#, ENTITY FRAMEWORK, SQL SERVER

Об’єкт розробки – серверна частина та база даних для програмної системи для рекомендації та підбору актуальних культурних подій.

Мета розробки – розробка програмної системи для рекомендації та підбору актуальних культурних подій.

Метод рішення – середовище розробки Visual Studio 2022 та SQL Server Management Studio, мова програмування C# та фреймворк ASP.NET 6, база даних MS SQL Server.

У результаті розробки створено серверну частину програмної системи для рекомендації та підбору актуальних культурних подій.

CULTURAL EVENTS RECOMMENDATION SYSTEM, SOFTWARE, BACKEND, API, ASP.NET 6, C#, ENTITY FRAMEWORK, SQL SERVER

The object of development is the server part and database for the software system for recommending and selecting current cultural events.

The purpose of the development is to develop a software system for recommending and selecting relevant cultural events.

The solution method is the Visual Studio 2022 development environment and SQL Server Management Studio, the C# programming language and the ASP.NET 6 framework, and the MS SQL Server database.

As a result of the development, the server part of the software system was created for recommending and selecting relevant cultural events.

Я, Хамінов Ілля Оександрович, студент гр. ПЗПІ-20-3, здобувач вищої освіти на першому (бакалаврському) рівні кафедри «Програмна інженерія», заявляю: моя кваліфікаційна робота на тему «Програмна система для рекомендації та підбору актуальних культурних подій. Back-end.», що буде представлена до екзаменаційної комісії для публічного захисту, виконана самостійно, в ній не містяться елементи плагіату і вона може бути опублікована в електронному архіві відкритого доступу ElAr KhNURE. Усі запозичення з друкованих та електронних джерел мають відповідні посилання.

Я ознайомлений з діючим положенням «Про протидію академічному плагіату в ХНУРЕ», згідно з яким виявлення плагіату є підставою для відмови до допуску кваліфікаційної роботи до захисту та застосування дисциплінарних заходів.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

JWT – JSON Web Token

API – Application Programming Interface

REST – REpresentational State Transfer

HTTPS – HyperText Transfer Protocol Secure

CI/CD – Continuous Integration/Continuous Deployment

JSON – JavaScript Object Notation

ВСТУП

В епоху стрімкого розвитку культурного життя, актуальним стає питання ефективної організації та відбору заходів, які б відповідали особистим уподобанням кожної людини. "Eventify" — це програмний продукт, який вирішує актуальну проблему вибору культурних заходів, адаптованих до індивідуальних уподобань, зацікавленості та геолокації користувачів.

Зі зростанням кількості культурних подій, таких як стендапи, виставки, концерти та фестивалі, виникає потреба в ефективній системі, яка не лише інформує про заходи, але й допомагає користувачам знайти найбільш релевантні події на основі їхніх уподобань та поточного місця розташування. "Eventify" використовує передові алгоритми машинного навчання для надання персоналізованих рекомендацій, роблячи процес вибору заходів зручнішим та інтуїтивно зрозумілим.

"Eventify" відповідає на зростаючі очікування сучасної аудиторії, яка цінує якість та персоналізацію у своєму культурному досвіді. Завдяки інноваційним технологічним рішенням, сервіс оперативно інформує про заходи, дозволяє фільтрувати та шукати події, а також сприяє зручному бронюванню квитків. Взаємодія з ШІ-асистентом для вибору подій робить цей процес ще простішим та приємнішим.

Головні цілі "Eventify" полягають у наданні високоточних та відповідних рекомендацій, ефективному керуванні користувацькими даними, а також дотриманні сучасних стандартів безпеки та конфіденційності. Створення цієї системи є вирішальним для сприяння культурній інтеграції та розвитку, враховуючи індивідуальні потреби користувачів.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Аналіз предметної галузі

Культурні розваги в сучасному суспільстві відіграють важливу роль, задовольняючи різноманітні потреби та впливаючи на розвиток індивідуальності та спільності. Однією з основних потреб, яку вони задовольняють, є потреба в розважальних та культурних формах дозвілля, що допомагають людям відпочити від повсякденних звичних занять та звільнити напругу. Крім того, культурні розваги стимулюють творчість та розвиток особистості, сприяючи вираженню індивідуальної культурної ідентичності. Вони надають можливість відчути себе частиною спільноти та збагачують духовний світ кожної людини, відкриваючи нові горизонти його артистичного та емоційного розвитку.

Сфера діяльності програмної системи "Eventify" охоплює індустрію розваг і культурних заходів, зокрема організацію та пошук різноманітних подій, таких як концерти, стендапи, виставки, ярмарки та фестивалі. Ця галузь відзначається широким географічним розповсюдженням заходів, великою кількістю учасників і різноманітністю інтересів аудиторії.

Учасники культурних подій належать всім віковим групам та соціальним статусам. Кожна вікова категорія має свої унікальні потреби та інтереси, але спільним для всіх є прагнення до отримання позитивних емоцій і нових вражень. Вони зіштовхуються з проблемою надмірної кількості доступних заходів і інформаційним перевантаженням, що ускладнює вибір відповідної події. Це створює зростаючу потребу в розробці ефективного інструменту, який би допомагав фільтрувати, рекомендувати та персоналізувати культурні заходи для користувачів.

Сфера культурних заходів охоплює широкий спектр подій, спрямованих на різні інтереси та смаки аудиторії. До таких заходів належать концерти, виставки, ярмарки, фестивалі, театральні вистави, кінопокази, літературні вечори, майстер-класи та багато інших. Кожен з цих форматів пропонує унікальні можливості для культурного розвитку та соціальної взаємодії.

Організатори заходів і куратори культурних просторів постійно шукають нові методи залучення аудиторії, покращення взаємодії з відвідувачами та підвищення якості їхнього досвіду.

Технологічний прогрес, особливо у сферах мобільних додатків, соціальних мереж та інтерактивних платформ, створює нові можливості для творчого підходу до організації та просування культурних заходів. "Eventify" використовує ці інновації, щоб забезпечити ефективну, інтуїтивно зрозумілу та доступну систему підбору подій.

На ринку представлено багато сервісів для пошуку цікавих подій для користувача, але лише деякі з них пропонують високий рівень персоналізації та інтеграцію з інтерактивними картами та штучним інтелектом, щоб покращити користувацький досвід.

Для аналізу було обрано телеграм канал “Афіша Київ” (див. рис. 1.1). Цей інтернет ресурс надає цікаві рекомендації, але немає можливості персоналізувати під вподобання користувача, тож цікаві події можуть загубитися серед інших.

Рисунок 1.1 – Телеграм канал “Афіша Київ” (за даними [1])

Другою системою аналогом є moemisto.ua (див. рис. 1.2). На сайті є можливість фільтрувати події за категоріями та датою, але немає карти, де можна швидко переглянути місце події, що уповільнює час пошуку, так як потрібно кожного разу переходити на інший сторонній додаток з картою, наприклад google maps і вбивати в пошук адресу та переглядати розташування.

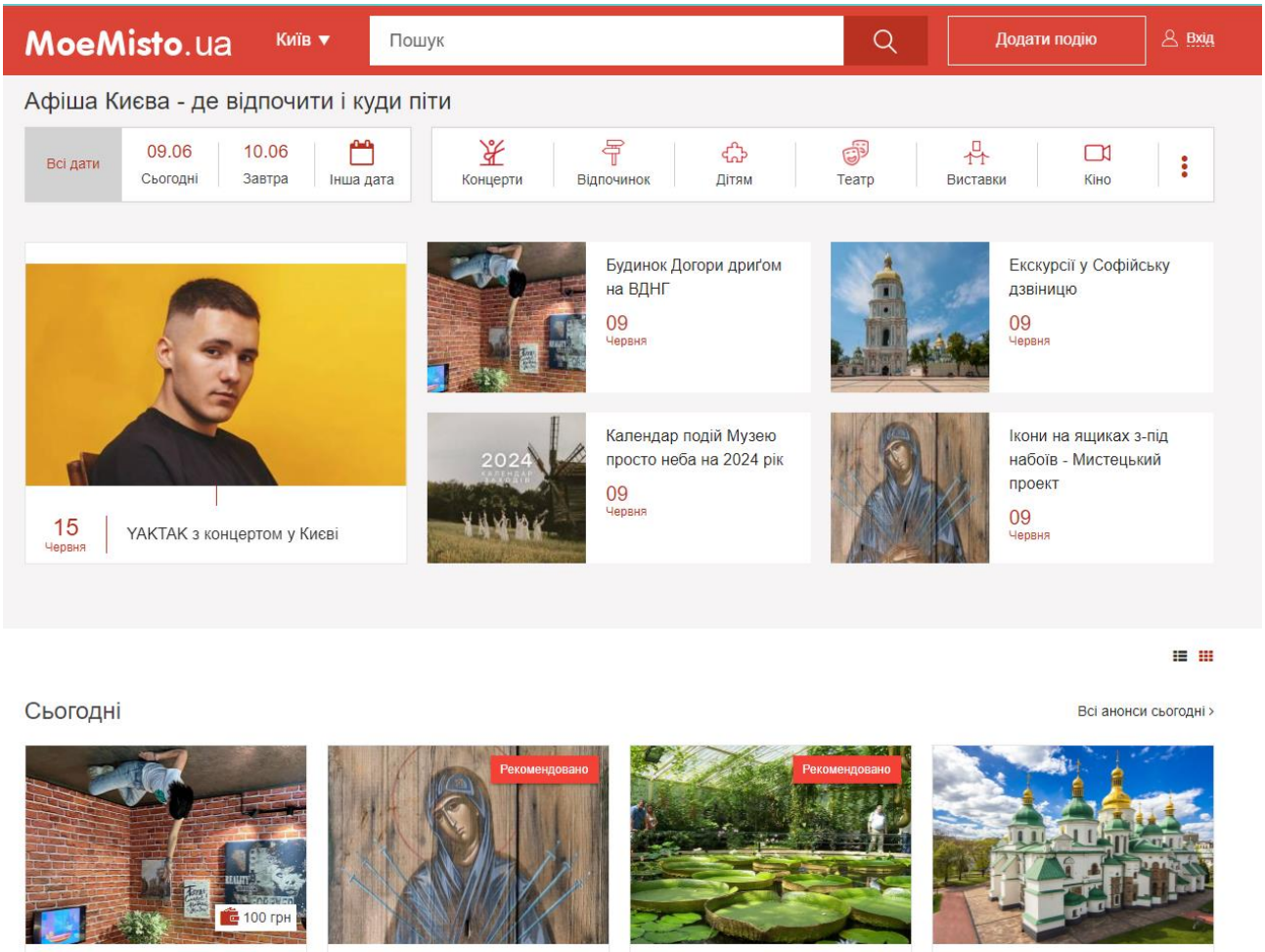


Рисунок 1.2 – Афіша подій moemisto.ua (за даними [2])

Серед основних функцій програмної системи “Eventify” можна виділити наступні:

– бронювання квитків – Інтеграція з платформами, що представляють організаторів подій, дозволяє користувачам легко бронювати квитки та запрошення на різноманітні заходи безпосередньо через інтерфейс системи, забезпечуючи їм зручність і ефективність процесу;

- перегляд заходів на мапі – Функція перегляду подій на інтерактивній карті місцевості спрощує процес вибору заходів, оскільки дозволяє користувачам швидко оцінити події в залежності від їх розташування на карті та зручно обирати ті, які розташовані поблизу;
- спілкування з ШІ-ботом – Використання штучного інтелекту для надання допомоги у виборі заходів, відповіді на запитання та надання додаткової інформації про заходи – це інтеграція інтелектуальних алгоритмів у систему, що допомагає користувачам приймати обґрунтовані рішення щодо вибору заходів, отримувати відповіді на свої запитання та знаходити більше інформації про ці заходи.

2.2 Виявлення проблем та актуалізація рішень

У програмі "Eventify", яка призначена для порад щодо культурних заходів, можна впровадити можливості виявлення проблем у процесі вибору культурних подій та реалізацію відповідних рішень. Ось декілька прикладів таких функціональностей:

- персоналізація досвіду користувача на основі пошуку культурної події – це процес використання зібраних даних про користувача та його інтереси для надання індивідуалізованих рекомендацій щодо культурних заходів;
- визначення та реакція на зміни в уподобаннях користувачів. Система може аналізувати зміни в активності користувачів і автоматично модифікувати пропозиції, пропонуючи нові типи подій відповідно до їх оновлених інтересів;
- взаємодія з інтерактивними картами для визначення місцезнаходження подій. Учасники можуть зазнавати труднощі зі знаходженням місць проведення подій, тому використання карт сприятиме їм у зручному плануванні свого часу та маршрутів;
- аналіз та відсіювання інформаційного перенасичення. Велика кількість подій породжує надмір інформації, яку користувачам

складно обробити. Система може використовувати складні алгоритми фільтрації для відсіювання непотрібної інформації;

- виявлення та реагування на зміни уподобань користувачів. Система може аналізувати зміни в поведінці користувачів і автоматично адаптуватися, пропонуючи нові види заходів згідно з оновленими інтересами;
- підтримка в реальному часі через ШІ-асистента. ШІ-асистент може надавати користувачам відповідаючи на запитання та аналізуючи їх потреби знаходити події, що їх цікавлять.

Ці функції призначені для вирішення певних проблем, з якими стикаються користувачі під час вибору культурних заходів, та сприяють актуалізації рішень для забезпечення покращеного користувацького досвіду.

Ось пропозиції щодо кроків для актуалізації рішень у системі "Eventify" для покращення взаємодії користувачів із сервісом та забезпечення більш ефективного вибору культурних заходів:

- персоналізовані рекомендації – розширення можливостей системи для збору даних про відвідувачів, їхні уподобання та минулий досвід, щоб надавати більш персоналізовані рекомендації щодо культурних заходів;
- аналіз поведінки користувачів – використання аналітики для ретельного вивчення поведінки користувачів на платформі, щоб зрозуміти їхні вподобання та потреби;
- оптимізація інтерфейсу користувача – розробка інтуїтивно зрозумілого користувацького інтерфейсу з легким доступом до функцій фільтрації, пошуку та перегляду заходів на карті;
- взаємодія зі спільнотою користувачів – створення форумів або можливості обміну думками і враженнями про культурні заходи серед користувачів, щоб стимулювати активну спільноту та взаємодію;

- інтеграція з інтерактивними мапами – забезпечення можливості перегляду подій на інтерактивній карті, що спростить процес вибору заходів відповідно до місця розташування користувача;
- розробка системи сповіщень – створення системи повідомлень, що автоматично інформує користувачів про нові культурні заходи або зміни в запланованих подіях, враховуючи їхні індивідуальні уподобання;
- застосування ШІ-асистента – використання розмовного ШІ-асистента у реальному часі для надання користувачам миттєвих відповідей на їх запитання, що сприяє вибору подій або вирішенню їхніх проблем.

2.3 Постановка задачі

Метою цієї роботи є розробка програмної системи "Eventify", яка надає рекомендації та персоналізує культурні заходи. Система має ефективно збирати, зберігати, обробляти та аналізувати дані, що стосуються інтересів та поведінки користувачів у контексті відвідування подій.

Основні завдання роботи включають:

- автоматичне збирання даних про уподобання користувачів, їхнє місцезнаходження та історію відвідувань заходів;
- розробка надійної бази даних для ефективного зберігання великого обсягу інформації про користувачів та події;
- обробка зібраних даних для визначення відповідності заходів уподобанням користувачів;
- створення алгоритмів для персоналізації рекомендацій, що враховують історичні дані та поточні запити користувачів;
- впровадження функціоналу для візуалізації заходів на інтерактивній карті, що полегшить користувачам планування відвідувань;
- інтеграція ШІ-асистента, який може в реальному часі консультувати користувачів, відповідати на їхні запитання та допомагати у виборі заходів;

– забезпечення заходів захисту даних для запобігання несанкціонованому доступу та збереження конфіденційності інформації.

Метою роботи є створення ефективного та надійного інструменту, який дозволить користувачам знаходити культурні заходи, що відповідають їхнім інтересам та потребам, тим самим підвищуючи загальну задоволеність та взаємодію з культурним контентом.

2 ФОРМУВАННЯ ВИМОГ ДО ПРОГРАМНОЇ СИСТЕМИ

Мова програмування та технології:

- серверна частина написана на мові C# [3] з використанням фреймворку ASP.NET 8 [4];
- використання Entity Framework Core [5] – об'єктно-орієнтована технологія доступу до бази даних;
- використання Code First [6] для створення бази даних;
- використання бази даних MS SQL Server [7] для зберігання даних;
- використання JWT [8] для аутентифікації запитів;
- використання MS Identity для авторизації та аутентифікації користувачів.

Взаємодія з веб додатком:

- розробка API (Application Programming Interface) [9] для забезпечення зв'язку між серверною частиною та веб-додатком;
- використання архітектурного стилю REST [10] для обробки та передачі ресурсів за найпоширенішим протоколом HTTPS [11];
- масштабованість та продуктивність;
- розміщення сервера та бази даних на хмарному сховищі Azure [12], що дозволяє додатку масштабуватися вертикально та горизонтально;
- оптимізація швидкості роботи системи для обробки великого обсягу даних та запитів до бази даних.

Тестування та документація:

- написання модульних та інтеграційних тестів для перевірки правильності роботи додатка;
- написання документації де зазначається вся інформація про API і які функціональні можливості серверна частина має.

Безпека та конфіденційність:

- хешування паролю користувача перед збереженням в базі даних;
- використання мережевого протоколу HTTPS для шифрування даних;

- використання JWT токенів для передачі даних для аутентифікації користувача в системі;
- використання параметризованих запитів для запобігання sql ін’єкцій;
- використання api key для аутентифікації та авторизації програми або користувача.

Враховуючи ці вимоги, розробка серверної частини на C# з використанням фре потужну та надійну програмну систему для рекомендацій

3. АРХІТЕКТУРА ТА ПРОЕКТУВАННЯ

3.1 UML проектування ПЗ

Діаграма варіантів використання (Use Case Diagram) є ключовою UML діаграмою, що використовується для моделювання функціональних вимог системи з точки зору зовнішніх акторів. Вона дозволяє визначити основні взаємодії між акторами та системою, виокремлюючи можливості системи, які повинні бути реалізовані.

У відношенні до системи "Eventify", діаграми варіантів використання допомагають уточнити потреби користувачів та описати функціональність системи. Давайте розглянемо кілька таких діаграм, які ілюструють взаємодію різних типів користувачів з системою (див. рис. 3.1, 3.2, 3.3).

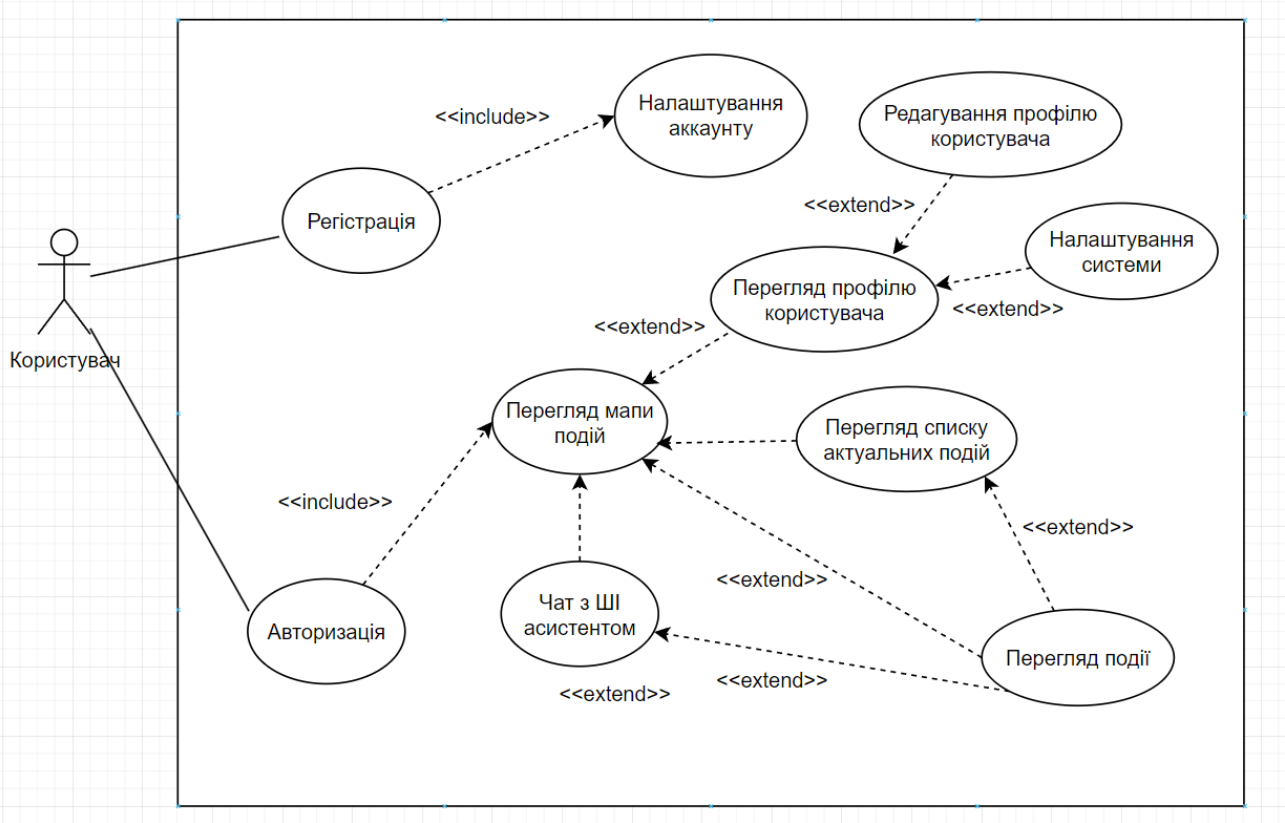


Рисунок 3.1 – Use-Case діаграма для Користувача (рисунок виконано самостійно)

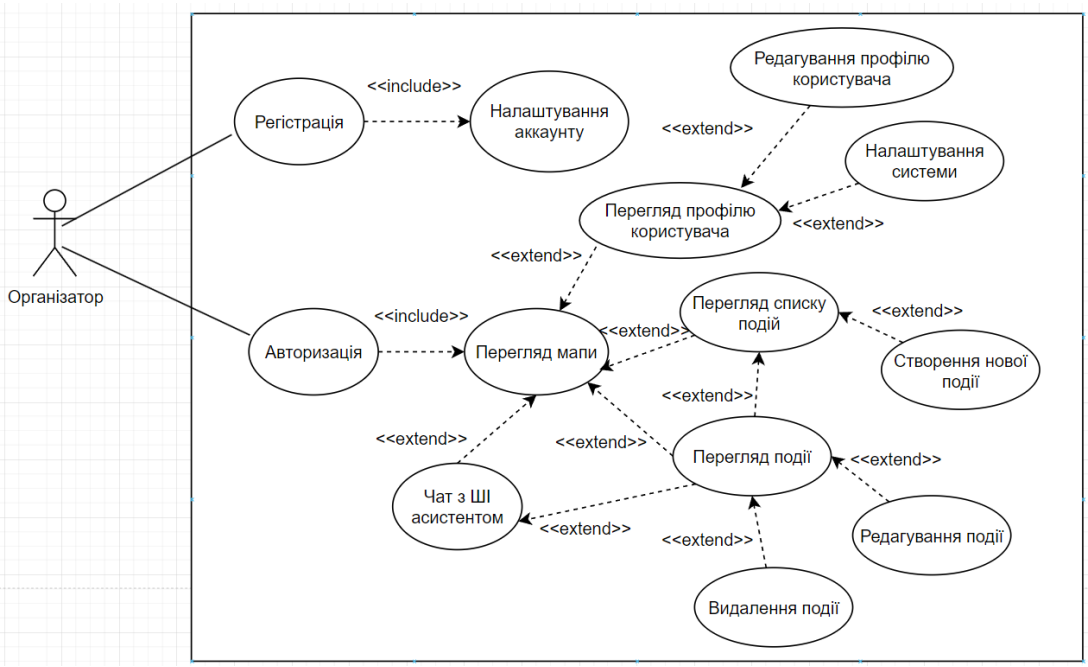


Рисунок 3.2 – Use-Case діаграма для Організатора (рисунок виконано самостійно)

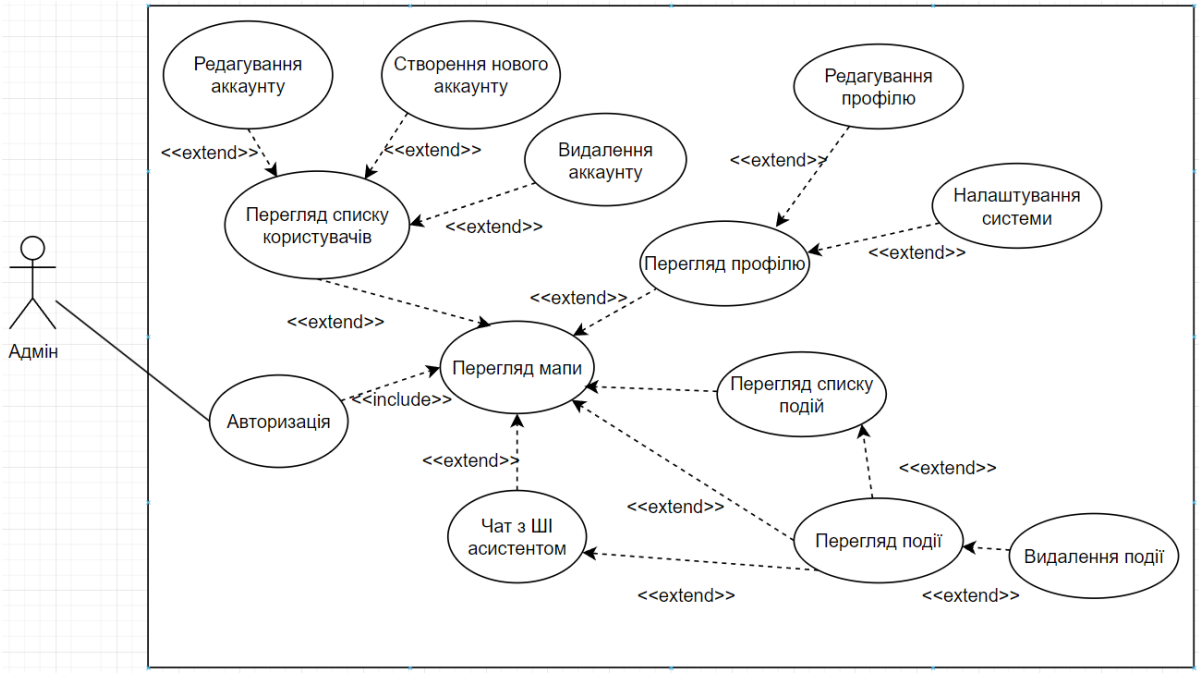


Рисунок 3.3 – Use-Case діаграма для Адміна (рисунок виконано самостійно)

Розглянемо акторів програмної системи “Eventify”:

а) Користувачі системи – це прості відвідувачі, які шукають інформацію про події та можуть взаємодіяти з різними функціями для планування

свого відпочинку. Ось деякі можливі способи використання користувачем системи:

1) Пошук подій – користувачі можуть шукати події за різними критеріями, такими як дата, місце, тип події тощо.
2) Перегляд деталей подій – користувачі можуть отримувати детальну інформацію про події, таку як опис, час проведення, місце, вартість квитків тощо.
3) Перегляд мапи та списку подій – користувачі можуть переглядати події на мапі поряд із собою чи окремий список.
4) Реєстрація та авторизація – користувач може створити та увійти в обліковий запис.
5) Спілкування з ШІ асистентом – спілкування з віртуальним помічником для отримання порад чи допомоги.

б) Організатори – це особи або групи, які відповідають за організацію подій і використовують систему для управління цим процесом, а також для аналізу відвідуваності. Ось можливі способи використання користувачем системи:

1) Додавання нових подій – створення і додавання до системи нових подій;
2) Редагування подій – внесення певних змін в існуючі події, які були створені цим обліковим записом;
3) Видалення подій – видалення існуючих подій, що були створені цим обліковим записом.

в) Адміністратори – відповідають за управління всією системою, що включає в себе керування користувачами, подіями а також забезпечення стабільності роботи додатку. Ось можливі способи використання користувачем:

1) Керування обліковими записами – створення, оновлення, перегляд та видалення облікових записів;
2) Перевірка правдивості та правильності створених подій.

3.2 Проектування архітектури ПЗ

В системі "Eventify" використовується клієнт-серверна архітектура, яка виступає основою для розподілення функцій між серверною (backend) та клієнтською (frontend та mobile) частинами. Спілкуються ці частини за допомогою мережевого протоколу HTTPS. До архітектури також входить вбудований компонент ШІ, що забезпечує розуміння природної мови та персоналізоване обслуговування користувачів, додаючи до системи функції розширеного аналізу та адаптивних рекомендацій.

Серверна частина (backend):

– написаний на мові програмування C# з використанням фреймворку ASP.NET 8. Він відповідає за основну бізнес логіку додатку та обробку даних, а також за поєднання з базою даних;
– як реляційна база даних використовується MSSQL Server (Microsoft SQL Server). Це сучасне, надійне та гнучке рішення для забезпечення високої продуктивності та широких можливостей для управління та аналізу даних.

Клієнтська частина (frontend):

– веб-додаток написаний за допомогою бібліотеки React.js та мови програмування JavaScript. Він забезпечує інтерактивний користувацький інтерфейс, взаємодіє з користувачем, приймає його введення та відображає дані, отримані через API;
– мобільний додаток написаний на мові Swift та забезпечує інтерактивний користувацький інтерфейс для мобільних пристроїв.

Штучний інтелект:

– Цю функціональність втілено за допомогою потужного і гнучкого фреймоворку LangChain, який розроблений на мові програмування Python. Він використовується для створення помічника, що буде вести діалог з користувачами, відповідати на запитань та надавати рекомендації щодо заходів, які би могли зацікавити.

Комунікація у системі:

– спілкування між клієнтською і серверною частиною відбувається за допомогою HTTPS – це розширення протоколу HTTP, яке підтримує захист даних при транспортуванні за допомогою шифрування інформації відповідно до стандартів SSL і TLS. Такий захист потрібний в комерційних ресурсах, де використовується інформація про конфіденційні або розрахункові дані користувача;
– клієнтська частина надсилає HTTPS запит на сервер, який використовує RESTful API для обробки та повертає данні у форматі JSON.

На серверній частині реалізована архітектура Clean Architecture - це підхід до проектування програмного забезпечення, який акцентує увагу на розподілі відповідальності та незалежності компонентів. Вона має модульну структуру, де бізнес-логіка відокремлена від зовнішніх факторів, таких як бази даних, веб-сервіси чи користувацький інтерфейс. Це робить систему більш гнучкою і дозволяє легко вносити зміни в окремі частини програми без впливу на інші компоненти.

Розробимо діаграму розгортання системи "Eventify", яка буде відображати структуру та взаємозв'язки між основними компонентами системи (див. рис. 3.4).

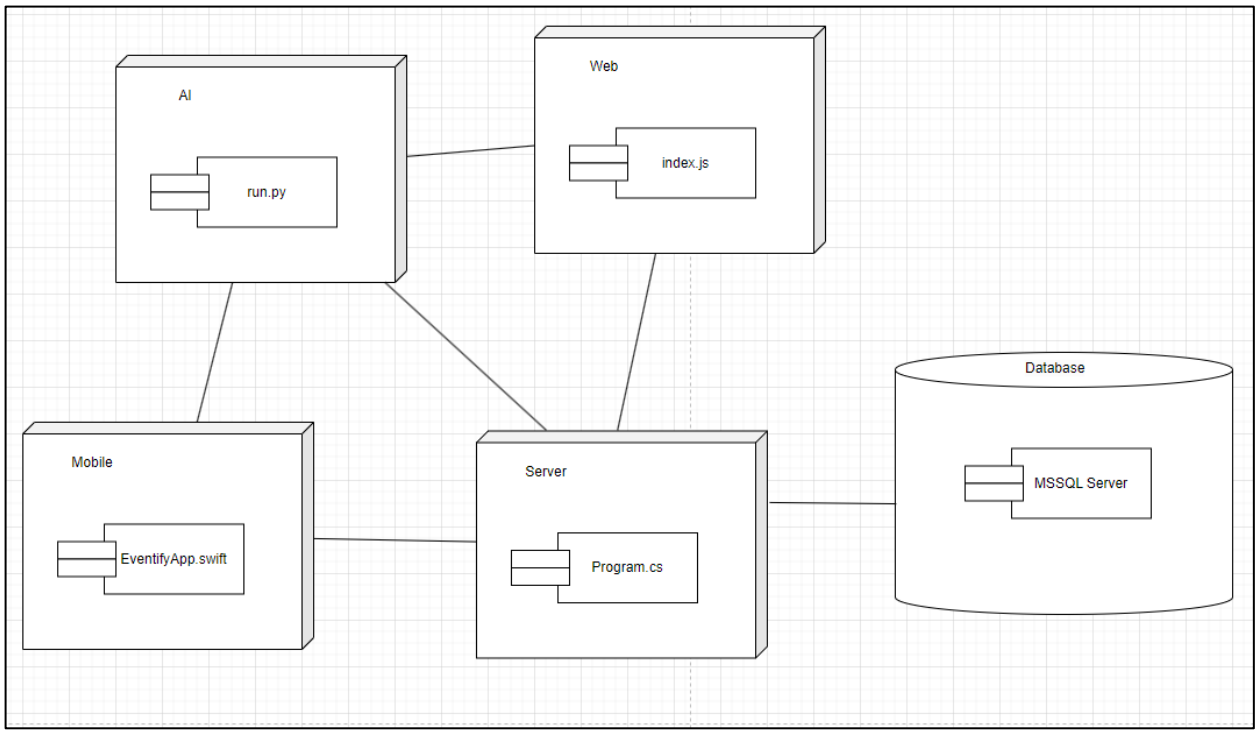


Рисунок 3.4 – Діаграма розгортання (рисунок виконано самостійно)

3.3 Проектування бази даних

Проектування бази даних для програмної системи рекомендацій подій вимагає глибокого розуміння бізнес-процесів, які потрібно відобразити в структурі бази даних. Вибір MSSQL, або Microsoft SQL Server, в ролі основного DBMS відображає потребу в сучасному, надійному та гнучкому рішенні для забезпечення високої продуктивності та широких можливостей для управління та аналізу даних. MS SQL відомий своєю високою надійністю, швидкістю та інтеграцією з іншими продуктами Microsoft, що робить його особливо зручним для використання в середовищі Windows. Окрім того, він володіє потужними інструментами для роботи з даними, включаючи засоби для управління, аналізу та звітності. Це, разом з високим рівнем підтримки та широкою спільнотою користувачів, робить MS SQL великим кандидатом для використання в проектах будь-якого масштабу.

SQL Server Management Studio (SSMS) - це інтегроване середовище для управління будь-якими компонентами SQL Server, від баз даних до звітів і аналізу. Це програмне забезпечення, розроблене Microsoft, яке надає зручний інтерфейс для адміністрування і роботи з базами даних на платформі SQL Server. З його допомогою можна створювати та редагувати бази даних, виконувати SQL-запити, розгортати бази даних, керувати безпекою, налаштовувати резервне копіювання та відновлення даних, а також проводити моніторинг та оптимізацію продуктивності. SSMS дуже популярний серед адміністраторів баз даних і розробників, які працюють з продуктами SQL Server.

В нашій системі використовується Code First - це один із підходів до роботи з Entity Framework. В цьому підході ви спочатку створюєте класи для сутностей вашої доменної моделі на C#, а потім використовуєте Entity Framework для автоматичного створення бази даних на основі цих класів. Це особливо корисно в Domain Driven Design, коли ви більше фокусуєтеся на домені вашого додатка, а не на проектуванні бази даних заздалегідь.

Таблиці бази даних, що використовуються сервером:

- таблиця «Event» зберігає інформацію основну інформація про подію, таку як назву, час, хто створив, опис, обмеження за віком;
- таблиця «Tag» містить інформацію про теги, що використовуються щоб відрізнити між собою події. Також надають можливість користувачам підписатися на появу нових події;
- таблиця «Location» зберігає інформацію про місце знаходження події на карті;
- таблиця «ViewHistory» використовується для зберігання реакції на подію користувача;
- таблиця «Settings» зберігає налаштування системи для окремого користувача;
- таблиця «AspNetRoles» містить данні про ролі в системі. Вони допомагають виокремити доступи в залежності від користувача;
- таблиця «AspNetsUser» використовується для зберігання основної інформації про користувача в системі. Вона містить його особисті та облікові данні;
- таблиця «AspNetsUserRoles» проміжна таблиця для «AspNetRoles» та «AspNetsUser» для зв’язку багато до багатьох.
- таблиця «Tag-Event» виступає проміжною таблицею між «Tag» і «Event», так як в нас подія може відповідати багатьом тегам і навпаки один тег відноситися до багатьох подій;
- таблиця «User-Tag» є проміжною таблицею для «User» та «Tag» так як в нас один користувач може бути підписаний на декілька тегів і навпаки на один тег може бути підписано багато користувачів.

Ось представлена ER діаграма бази даних додатку “Eventify” (див. рис. 3.5):

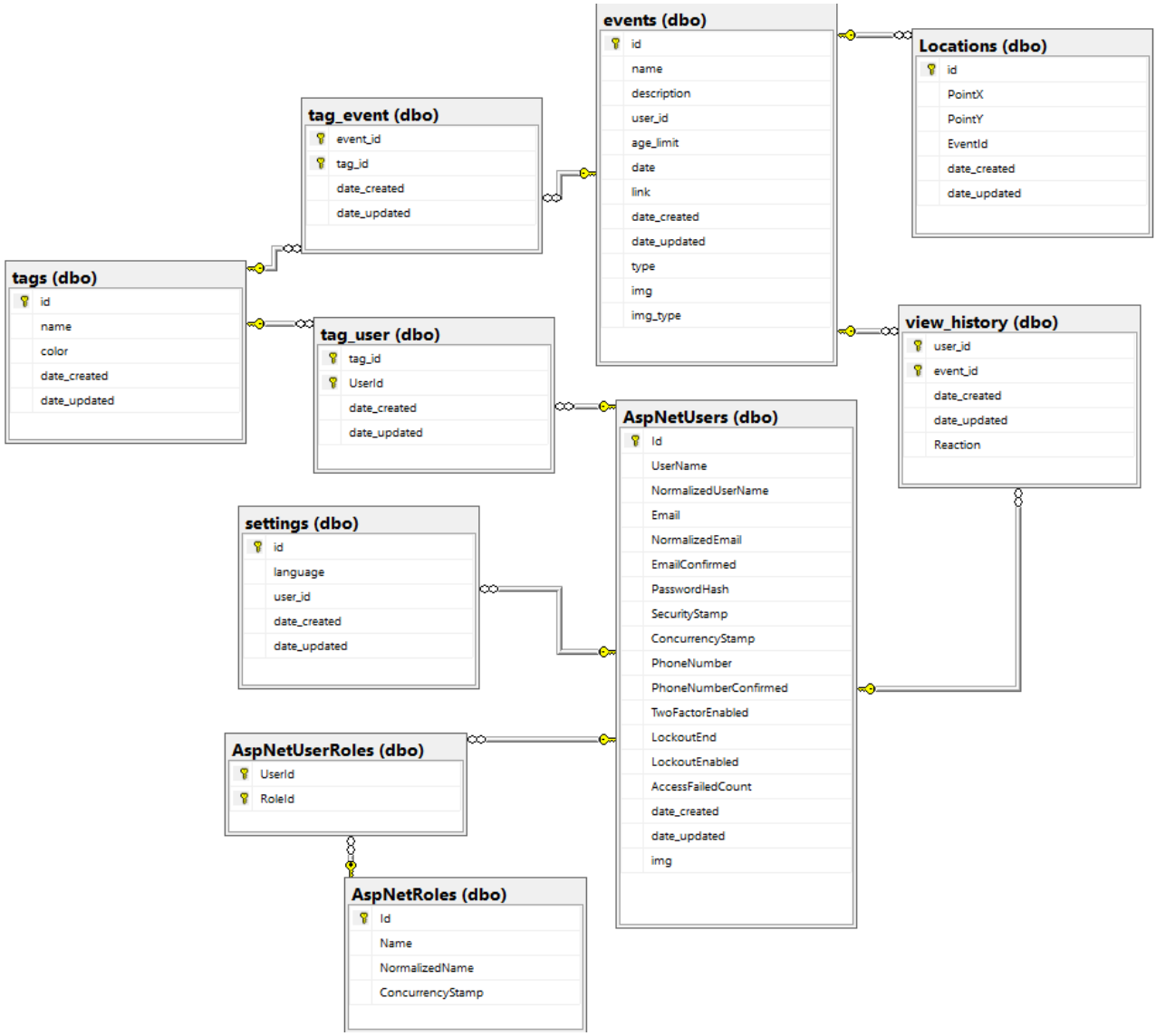


Рисунок 3.5 – ER діаграма (рисунок виконано самостійно)

База даних знаходиться в третій нормальній формі, що підтверджує її нормалізованість. Однією з переваг такої структури є відсутність повторюваної інформації, що робить зберігання та обробку даних більш зручними. Це сприяє ефективному використанню пам'яті, особливо при роботі з великими обсягами даних.

3.4 Огляд алгоритмів та методів

Як для мене, найцікавіший та найскладніший алгоритм у системі “Eventify” є алгоритм надання рекомендації подій для користувачів. Після запиту користувача, ключовим етапом є визначення та рекомендація події, яка

найкраще відповідає його інтересам. Алгоритм детально описує процес вибору події:

- аналіз запиту – це перший крок, де чат-бот обробляє запит користувача, розпізнає ключові слова і на основі цих даних створює запит до серверу. Наприклад, надходить запит “я хочу відвідати виставку десь в центрі Києва”. Чат-бот виділяє – “виставка”, “центр Києва” як основні критерії пошуку;
- взаємодія з сервером – другим кроком сервер приймає запит і з отриманими даними формує критерії і надсилає запит в базу даних, отримуючи результат прораховує відстань від заданої локації і надає найбільш підходящі вимогам користувача події;
- зміна критеріїв – якщо сервер нічого не повернув, система розширює критерії пошуку і знову надсилає запит з новими даними;
- взаємодія з сервером – сервер приймає запит і з новими отриманими даними ще раз взаємодіє з базою для знаходження потрібних подій;
- надання рекомендації – чат-бот обробляє отриману відповідь і пропонує користувачу всі доступні варіанти, з наданням деталей. Також надає можливість забронювати або отримати більше інформації про обрану ним подію.

4 ОПИС ПРИЙНЯТИХ ПРОГРАМНИХ РІШЕНЬ

4.1 Налаштування основної частин серверу

Для роботи з базою даних використуву’ється EF Core - сучасний об'єктно-реляційний об'єктно-орієнтований фреймворк (ORM), розроблений Microsoft, який дозволяє розробникам .NET працювати з базами даних за допомогою об'єктів .NET. Щоб його налаштувати потрібно встановити потрібні NuGet пакети (див. рис. 4.1).

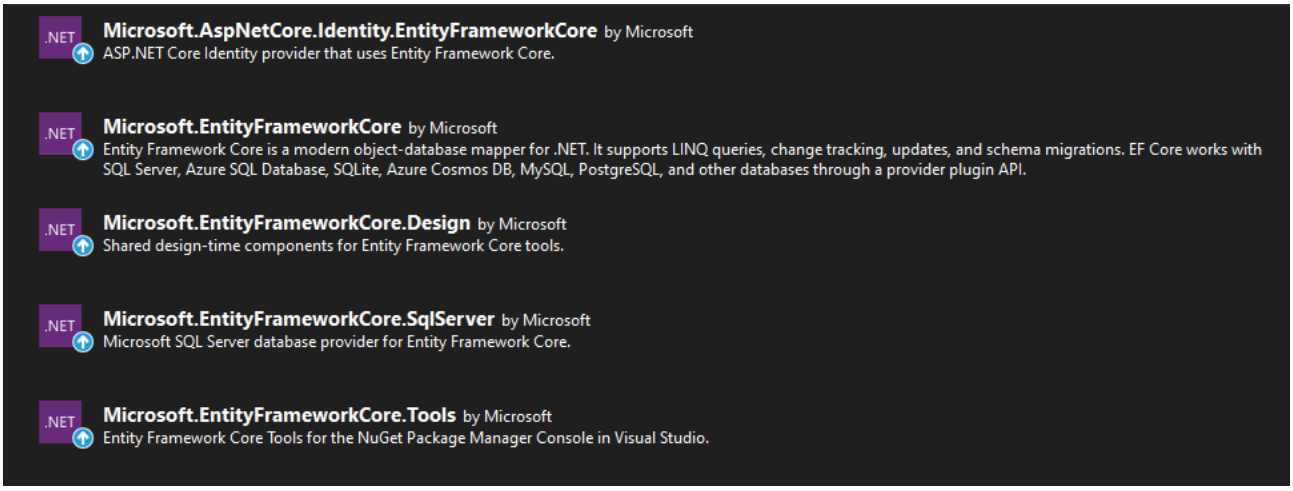
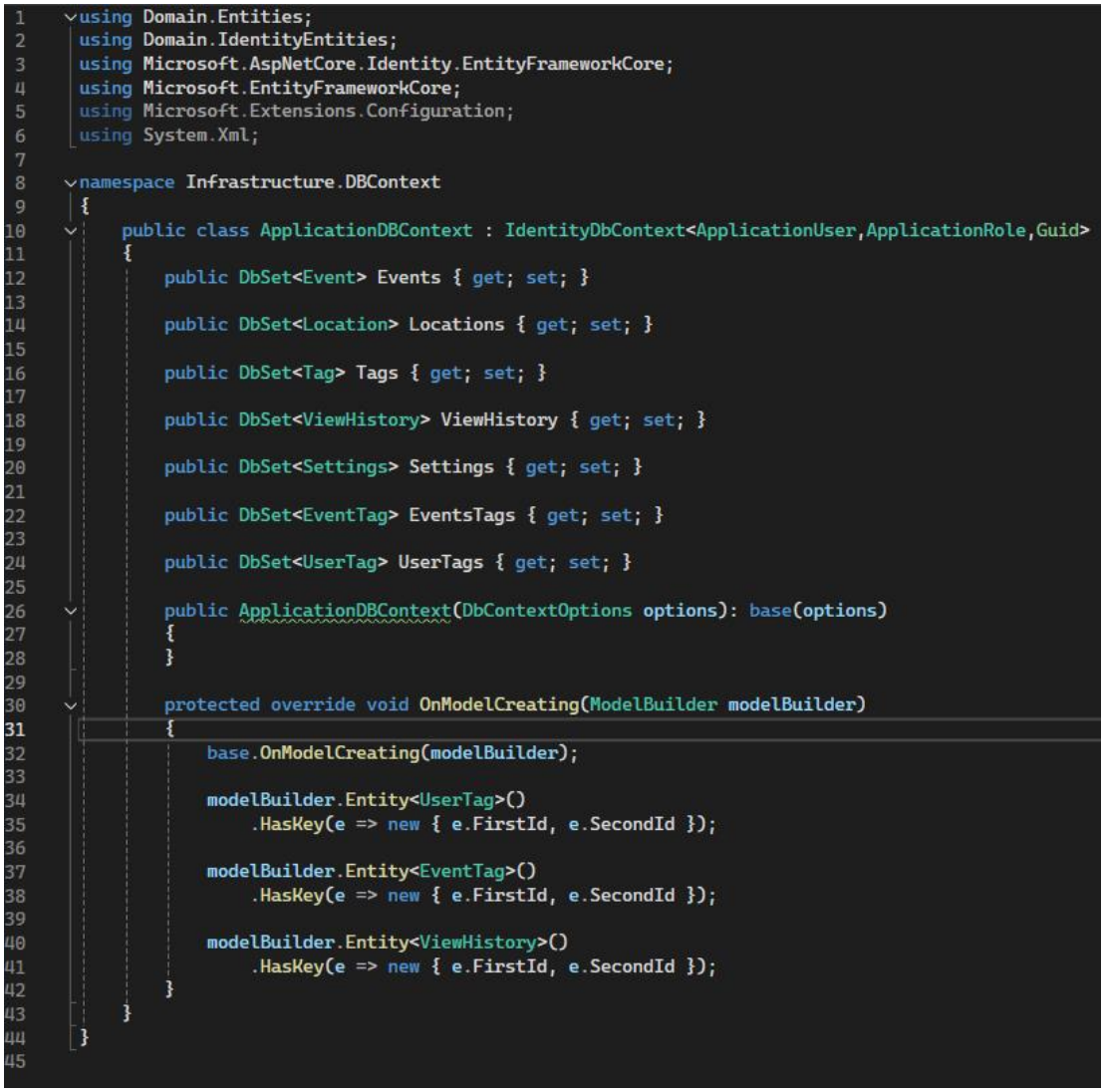


Рисунок 4.1 – NuGet пакети для EF (рисунок виконано самостійно)

Також створити DB Context (див. рис. 4.2), та так як використовується підхід Code First – прописати всі моделі за якими створюються таблиці в базі даних. В цих моделях за допомогою атрибутів «Key» та «ForeignKey» налаштовуються зв’язки один до багатьох. Правильно налаштовані зв'язки відіграють вирішальну роль у забезпеченні цілісності даних і ефективності виконання запитів до бази даних. Далі за допомогою команд “Add-Migration” створити міграції - механізм, який дозволяє автоматично створювати, змінювати і підтримувати схему бази даних на основі змін у модельних класах та “Update-Database” застосовувати останні міграції до бази даних.



```
using Domain.Entities;
using Domain.IdentityEntities;
using Microsoft.AspNetCore.Identity.EntityFrameworkCore;
using Microsoft.EntityFrameworkCore;
using Microsoft.Extensions.Configuration;
using System.Xml;

namespace Infrastructure.DBContext
{
    public class ApplicationDBContext : IdentityDbContext<ApplicationUser,ApplicationRole,Guid>
    {
        public DbSet<Event> Events { get; set; }

        public DbSet<Location> Locations { get; set; }

        public DbSet<Tag> Tags { get; set; }

        public DbSet<ViewHistory> ViewHistory { get; set; }

        public DbSet<Settings> Settings { get; set; }

        public DbSet<EventTag> EventsTags { get; set; }

        public DbSet<UserTag> UserTags { get; set; }

        public ApplicationDBContext(DbContextOptions options): base(options)
        {
        }

        protected override void OnModelCreating(ModelBuilder modelBuilder)
        {
            base.OnModelCreating(modelBuilder);

            modelBuilder.Entity<UserTag>()
                .HasKey(e => new { e.FirstId, e.SecondId });

            modelBuilder.Entity<EventTag>()
                .HasKey(e => new { e.FirstId, e.SecondId });

            modelBuilder.Entity<ViewHistory>()
                .HasKey(e => new { e.FirstId, e.SecondId });
        }
    }
}

```

Рисунок 4.2 – Application EF Context (рисунок виконано самостійно)

Головним файлом програми є Program.cs (див. рис. 4.3) – відповідає за ініціалізацію та конфігурацію додатка. Тут вказується всі налаштування додатку, в тому числі взаємодію EF Core з MS Identity.

```
using ReactApp.Server.SturtupExtensions;
var builder = WebApplication.CreateBuilder(args);

var authConfig = new AuthConfig();
builder.Configuration.Bind("Jwt", authConfig);

builder.Services.AddControllers();

builder.Services.AddCors(o =>
    o.AddDefaultPolicy(b =>
        b.AllowAnyOrigin().AllowAnyMethod().AllowAnyHeader()));

builder.Services.AddEndpointsApiExplorer();

builder.Services.AddDbContext<ApplicationDBContext>(options =>
{
    options.UseSqlServer(builder.Configuration.GetConnectionString("DefaultConnection"),
        sqlServerOptionsAction: sqlOptions =>
        {
            sqlOptions.EnableRetryOnFailure(
            maxRetryCount: 5, // Number of retry attempts
            maxRetryDelay: TimeSpan.FromSeconds(30), // Delay between retries
            errorNumbersToAdd: null // Optional: Additional SQL error codes to retry on
        );
        });
});

builder.Services.ConfigureServicesForAuth(builder.Configuration, authConfig);
builder.Services.ConfigureServicesForDI(builder.Configuration);

var app = builder.Build();

// Configure the HTTP request pipeline.
if (app.Environment.IsDevelopment())
{
    app.UseSwagger();
    app.UseSwaggerUI();
}

app.UseHttpsRedirection();

app.UseRouting();

app.UseAuthentication();
app.UseAuthorization();

app.UseCors();

app.UseEndpoints(endpoints =>
{
    endpoints.MapControllerRoute(
        name: "default",
        pattern: "{controller}/{action=Index}/{id?}");
});

app.MapControllers();

app.Run();
```

Рисунок 4.3 – код файлу Program.cs (рисунок виконано самостійно)

По переше, реєструється контекст бази даних з MS Identity (див. рис. 4.4 та 4.5).

```
builder.Services.AddDbContext<ApplicationDBContext>(options =>
{
    options.UseSqlServer(builder.Configuration.GetConnectionString("DefaultConnection"),
        sqlServerOptionsAction: sqlOptions =>
        {
            sqlOptions.EnableRetryOnFailure(
            maxRetryCount: 5, // Number of retry attempts
            maxRetryDelay: TimeSpan.FromSeconds(30), // Delay between retries
            errorNumbersToAdd: null // Optional: Additional SQL error codes to retry on
        );
        });
```

Рисунок 4.4 – Реєстрація контексту бази даних (рисунок виконано самостійно)

```
services.AddIdentity<ApplicationUser, ApplicationRole>(options =>
{
    options.Password.RequiredUniqueChars = 1;
    options.Password.RequireDigit = true;
    options.Password.RequiredLength = 8;
})
    .AddEntityFrameworkStores<ApplicationDBContext>()
    .AddDefaultTokenProviders();
```

Рисунок 4.5 – Реєстрація MS Identity (рисунок виконано самостійно)

Також потрібно додати в такій послідовності середовище аутентифікації (див. рис. 4.6).

```
app.UseAuthentication();
app.UseAuthorization();
```

Рисунок 4.6 – Середовище аутентифікації (рисунок виконано самостійно)

Ці рядки додають обробку аутентифікації та авторизації до конвеєра нашого додатка, що дозволяє використовувати функціонал Identity для перевірки користувачів.

JWT (JSON Web Token) — це компактний, самодостатній токен у форматі JSON, який використовується для безпечної передачі інформації між сторонами як JSON-об'єкт. Він застосовується для аутентифікації та авторизації користувачів у багатьох сучасних веб-додатках. Налаштування JWT у .NET 6 передбачає конфігурацію сервісів аутентифікації та налаштування політик

авторизації. Правильне налаштування цих сервісів знаходиться у класі Program (див. рис. 4.7).

```
services.AddAuthentication(options =>
{
    options.DefaultAuthenticateScheme = JwtBearerDefaults.AuthenticationScheme;
    options.DefaultChallengeScheme = JwtBearerDefaults.AuthenticationScheme;
    options.DefaultScheme = JwtBearerDefaults.AuthenticationScheme;
})
    .AddJwtBearer(jwt =>
    {
        jwt.SaveToken = true;
        jwt.RequireHttpsMetadata = false;
        jwt.TokenValidationParameters = new TokenValidationParameters()
        {
            ValidateIssuer = true,
            ValidateAudience = true,
            ValidAudience = authConfig.Audience,
            ValidIssuer = authConfig.Issuer,
            ValidateLifetime = true,
            IssuerSigningKey = new SymmetricSecurityKey(Encoding.UTF8.GetBytes(authConfig.Key)),
            ValidateIssuerSigningKey = true,
        };
    });
```

Рисунок 4.7 – Налаштування JWT (рисунок виконано самостійно)

Це необхідним для того, щоб додаток правильно визначав і обробляв JWT токени, забезпечуючи безпечний обмін даними та відповідність усім вимогам безпеки.

4.2 Розміщення додатку на Azure

Для деплою «Eventify» на Azure використовується GitHub Actions - платформа автоматизації, вбудована в GitHub, яка дозволяє користувачам створювати, налаштовувати та виконувати конвеєри CI/CD (Continuous Integration/Continuous Deployment) безпосередньо у своїх репозиторіях.

Щоб налаштувати GitHub Actions потрібно створити репозиторій на GitHub, куди вивантажити програму (див. рис. 4.8).

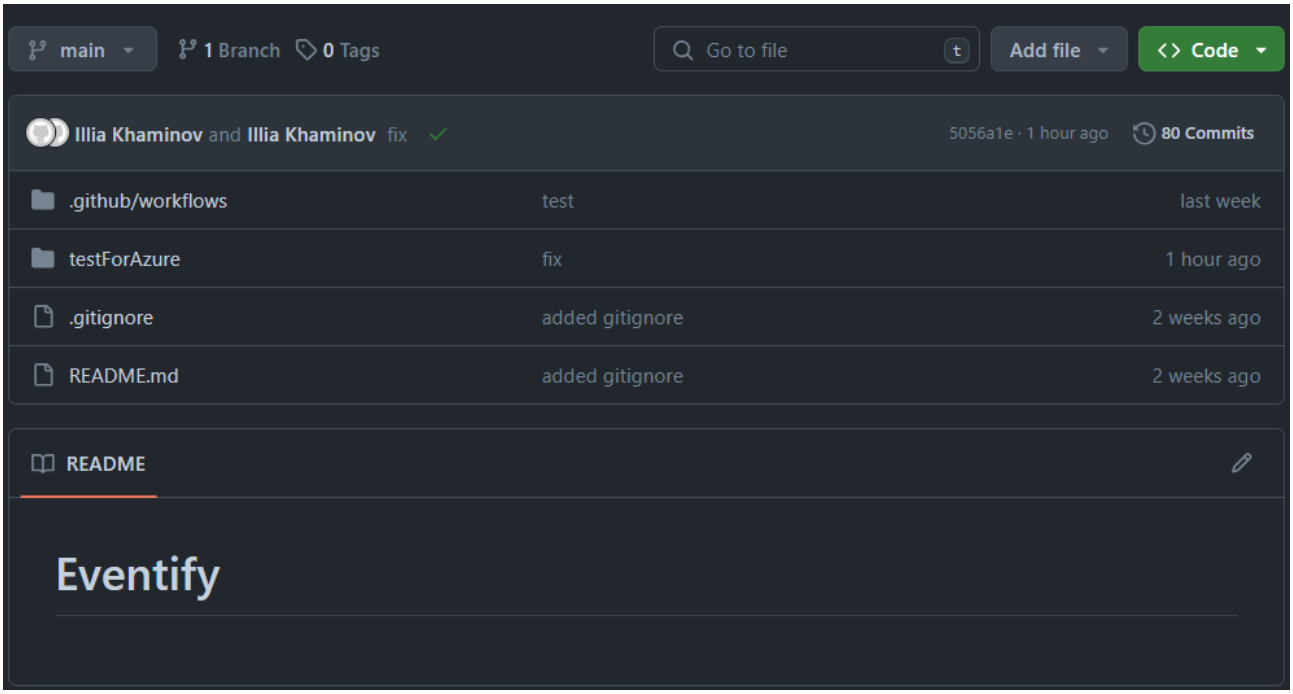


Рисунок 4.8 – Репозиторій на GitHub (рисунок виконано самостійно)

Після цього потрібно створити Azure Web App (див. рис. 4.9), де під час створення вказати який GitHub репозиторій підв’язати під цей веб додаток.

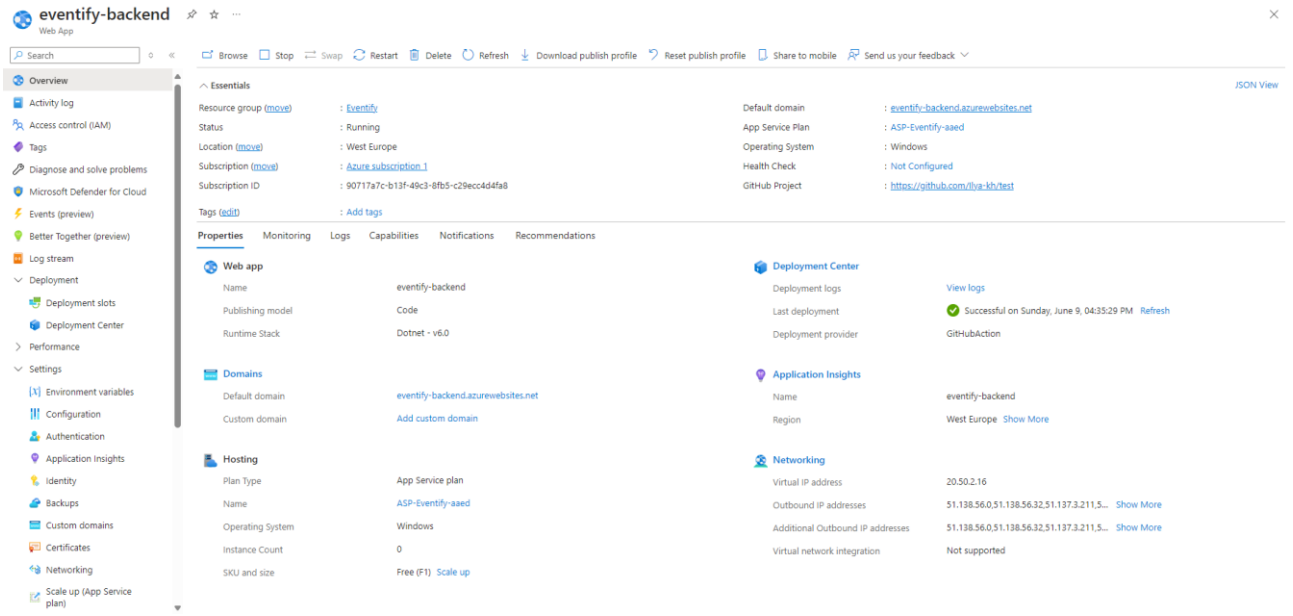


Рисунок 4.9 – Azure Web App (рисунок виконано самостійно)

Тепер потрібно створити yaml скрипт (див. рис. 4.10), що налаштовує роботу GitHub Actions.

```yaml
on:
  push:
  workflow_dispatch:

jobs:
  build:
    runs-on: windows-latest

    steps:
      - uses: actions/checkout@v4

      - name: Set up .NET Core
        uses: actions/setup-dotnet@v1
        with:
          dotnet-version: '6.0.x'
          include-prerelease: true

      - name: Build with dotnet
        run: dotnet build --configuration Release
        working-directory: testForAzure

      - name: dotnet publish
        run: dotnet publish -c Release -o ${{env.DOTNET_ROOT}}/myapp
        working-directory: testForAzure

      - name: Upload artifact for deployment job
        uses: actions/upload-artifact@v3
        with:
          name: .net-app
          path: ${{env.DOTNET_ROOT}}/myapp

  deploy:
    runs-on: windows-latest
    needs: build
    environment:
      name: 'Production'
      url: ${{ steps.deploy-to-webapp.outputs.webapp-url }}
    permissions:
      id-token: write #This is required for requesting the JWT

    steps:
      - name: Download artifact from build job
        uses: actions/download-artifact@v3
        with:
          name: .net-app

      - name: Login to Azure
        uses: azure/login@v1
        with:
          client-id: ${{ secrets.AZUREAPPSERVICE_CLIENTID_E770B828728E49019FDAF54CA1EF6774 }}
          tenant-id: ${{ secrets.AZUREAPPSERVICE_TENANTID_B625357F0BF447B4A148E60A7F6333E1 }}
          subscription-id: ${{ secrets.AZUREAPPSERVICE_SUBSCRIPTIONID_8AFFEAF14D8243498B6207BD02A777AF }}

      - name: Deploy to Azure Web App
        id: deploy-to-webapp
        uses: azure/webapps-deploy@v2
        with:
          app-name: 'eventify-backend'
          slot-name: 'Production'
          package: .
```

Рисунок 4.10 - Yaml скрипт (рисунок виконано самостійно)

Після кожного змінення в «main» гілці репозиторія, автоматично запускається скрипт, що розгортає додаток. Ось так виглядає робота GitHub

Actions (див. рис. 4.11). Тепер є оновлений веб додаток, який розміщений на Azure. .

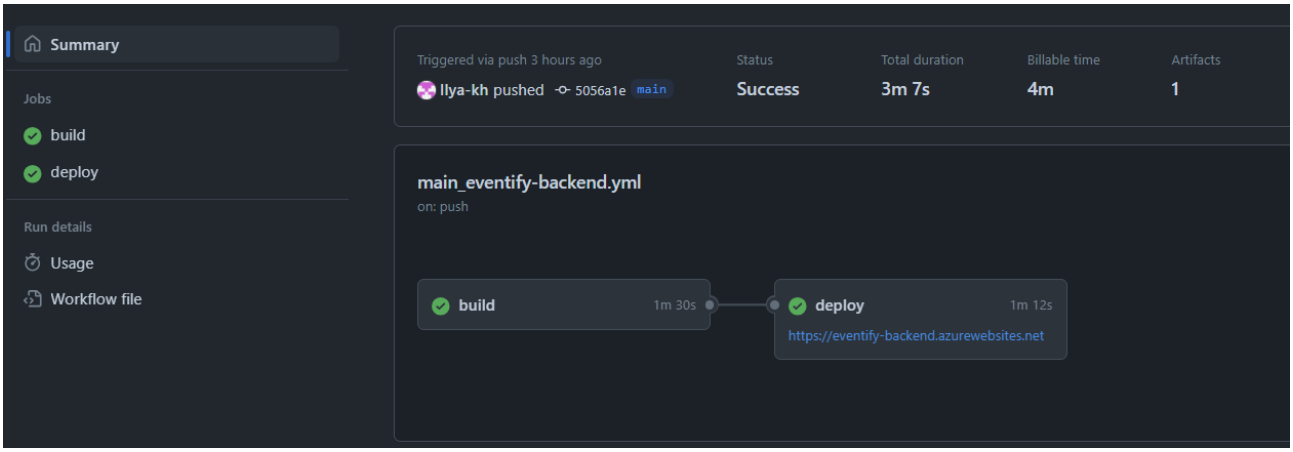


Рисунок 4.11 – Робота GitHub Actions (рисунок виконано самостійно)

Тут ми можемо побачити кроки роботи, де build – збірка додатку та deploy його розгортання в середовищі Azure.

5 ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

5.1 Тестування серверу

Тестування є критично важливим етапом у процесі розробки програмного забезпечення з кількох ключових причин: забезпечення якості (допомагає виявити помилки, дефекти та невідповідності в програмному забезпеченні ще до його випуску), підвищення надійності (ретельно протестоване програмне забезпечення є більш стабільним та надійним), економія часу та ресурсів (виявлення та виправлення помилок на ранніх етапах розробки є менш витратним) та покращення користувацького досвіду (якісне програмне забезпечення, яке працює без збоїв і відповідає очікуванням користувачів, забезпечує кращий користувацький досвід).

У контексті Eventify, використовується Swagger для проведення тестування. За допомогою Swagger можно легко переглядати та взаємодіяти з різними ресурсами API, а також виконувати різноманітні запити, такі як GET, POST, PUT, DELETE та інші. Одна з основних переваг використання Swagger полягає в тому, що є можливість взаємодіяти з API візуально, використовуючи інтерфейс користувача Swagger UI, який генерується автоматично з документації вашого API. Це робить процес тестування більш інтуїтивно зрозумілим та ефективним, оскільки не потрібно запам'ятовувати або вводити URL-адреси, заголовки або параметри запиту вручну. Крім того, Swagger дозволяє легко оглядати різні ресурси та їхні методи, а також переглядати вихідні дані та відповіді, що надходять від сервера. Це дозволяє швидко перевіряти правильність роботи API та виявляти будь-які проблеми чи невідповідності у відповідях сервера.

На рисунку 5.1 відображається в Swagger UI запит на отримання подій, який містить вхідні дані на основі яких поверне лист подій. Поля “sortBy” та “sortAscending” відповідає за яким полем сортувати та в якому напрямку відповідно. Поля “page” та “pageSize” відповідають за номер сторінки, та кількість подій на одній сторінці відповідно. “types” відповідає за фільтрацію по типам а “tags” по тегам.

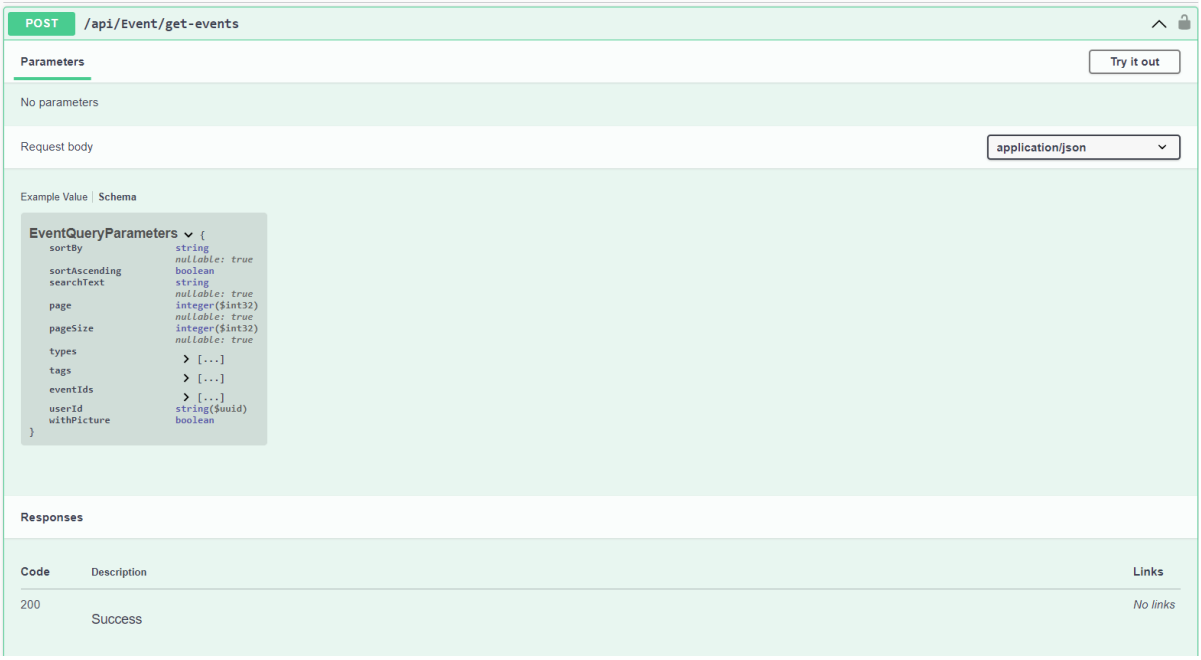


Рисунок 5.1 – Тестування отримання подій (рисунок виконано самостійно)

Наприклад було надіслано номер сторінки, розмір однієї сторінки та прапор чи потрібна картинка для події (див. рис. 5.2). Після цього було отримано список подій (див. рис. 5.3)

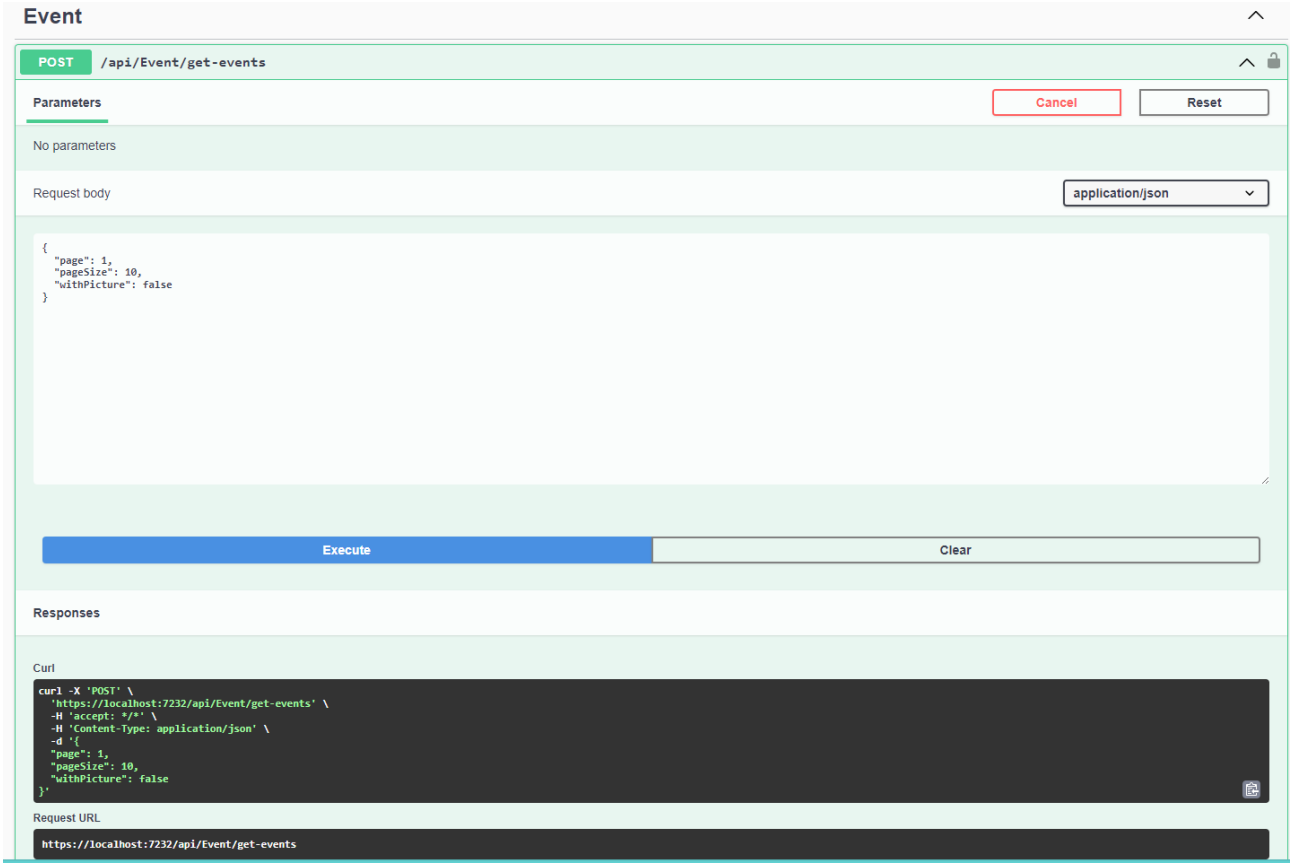


Рисунок 5.2 – Відправка запиту (рисунок виконано самостійно)

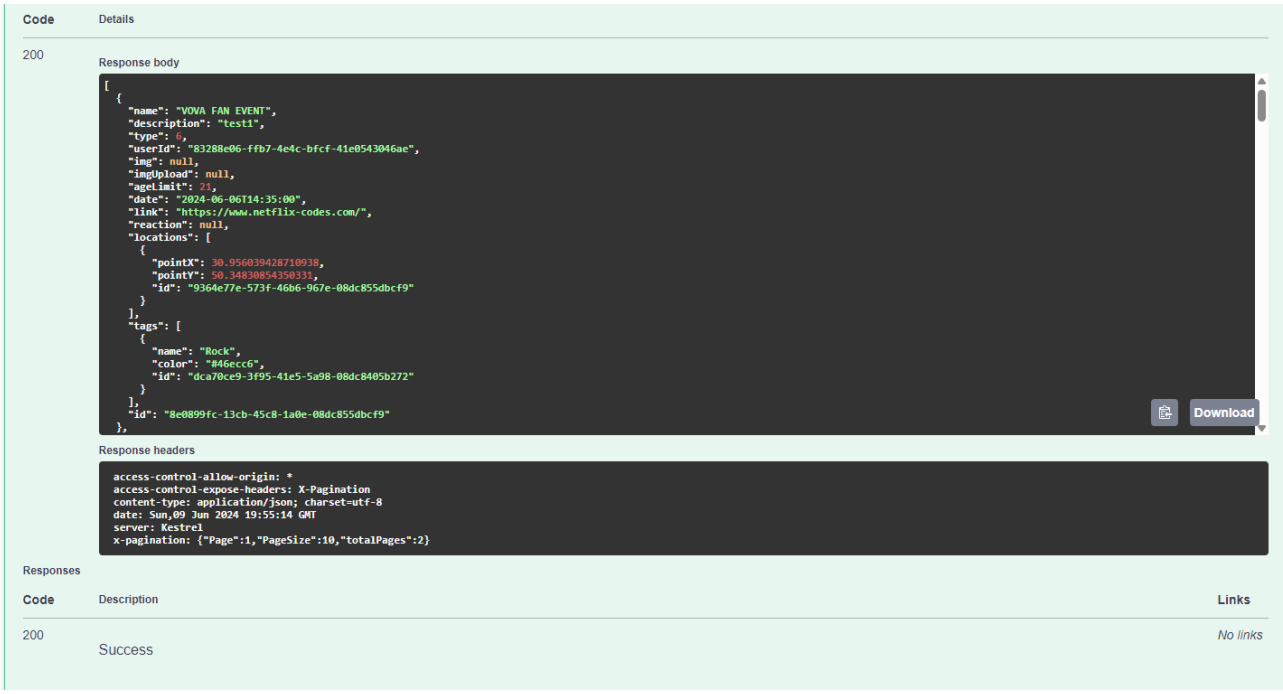


Рисунок 5.3 – Отримання результату (рисунок виконано самостійно)

Таким чином, процес передачі даних між клієнтом і сервером виглядає як послідовність кроків, де сервер отримує дані від клієнта, обробляє їх, формує відповідь у форматі JSON і надсилає її назад клієнту.

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи бакалавра було проведено аналіз предметної області, наявних програмних рішень та їхніх недоліків. На основі цього аналізу визначили напрямки для покращення існуючих рішень та встановили основні вимоги і завдання для розробки нової системи. Зокрема, були визначені ключові вимоги.

Для реалізації серверної частини системи було обрано C# з використанням фреймворку ASP.NET 6. Використання цих технологій сприяло створенню надійної архітектури, здатної ефективно взаємодіяти з фронтендом через API, забезпечуючи безпеку, швидкість обробки даних і легку інтеграцію з іншими системами та сервісами.

Для зберігання даних використовується MSSQL Server, потужна та надійна реляційна база даних, яка пропонує широкі можливості для зберігання й оптимізації інформації.

Веб-система розроблена за допомогою бібліотеки React.js та мови програмування JavaScript, що забезпечує інтерактивність та високу швидкість відгуку завдяки використанню підходу SPA. Це оптимізує процес завантаження сторінок і взаємодії з сервером, забезпечуючи плавну та зручну навігацію без необхідності перезавантаження сторінки, покращуючи загальний користувацький досвід.

Мобільна частина системи розроблена з використанням мови програмування Swift, що забезпечує високу продуктивність та оптимальну адаптацію до потреб мобільних користувачів. Це дозволяє користувачам залишатися на зв'язку з системою "Eventify" у будь-який час і з будь-якого місця, забезпечуючи швидкий доступ до інформації про події та можливість реагування на оновлення.

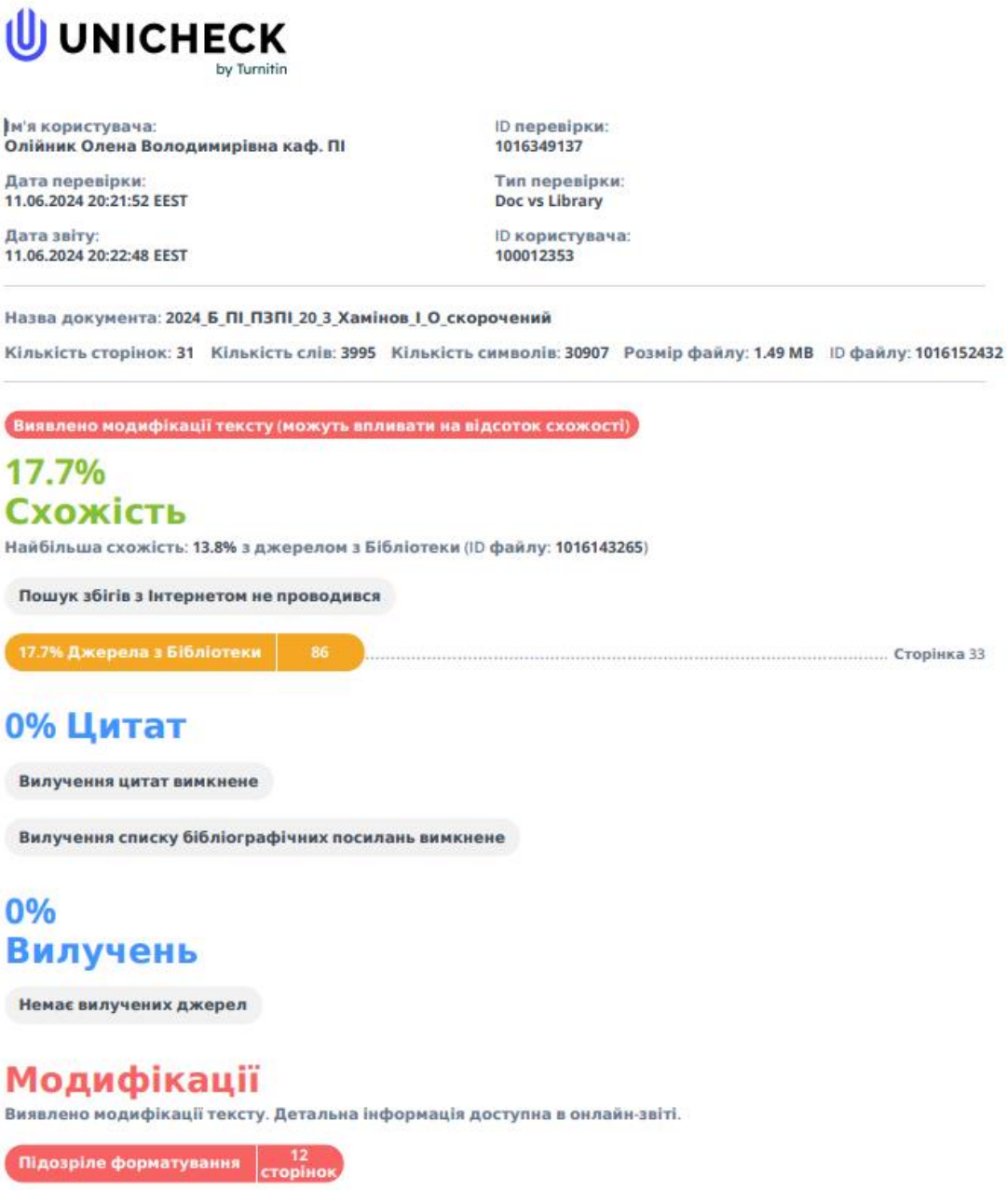
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Телеграм канал “Афіша Київ” [Електронний ресурс] – URL: https://t.me/afisha_kyiva (дата звернення: 09.06.2024).
2. Афіша подій moemisto.ua [Електронний ресурс] – URL: https://moemisto.ua/vn/kontserti (дата звернення: 09.06.2024).
3. C# docs - get started, tutorials, reference. Microsoft Learn: Build skills that open doors in your career. [Електронний ресурс] – URL: https://docs.microsoft.com/en-us/dotnet/csharp/ (дата звернення: 02.05.2024).
4. Kumar V., Kumar V. ASP. NET Web API //Beginning Windows 8 Data Development: Using C# and JavaScript. – 2013. – С. 123-146.
5. Schwichtenberg H., Schwichtenberg H. Introducing entity framework core //Modern Data Access with Entity Framework Core: Database Programming Techniques for. NET,. NET Core, UWP, and Xamarin with C#. – 2018. – С. 1-14.
6. Parent C., Spaccapietra S. Issues and approaches of database integration //Communications of the ACM. – 1998. – Т. 41. – №. 5es. – С. 166-178.
7. Мазурова О. А., Широкопетлева М. С., Черепанова Ю. Ю. Информационные системы. [Електронний ресурс] – URL: https://dl.nure.ua/pluginfile.php/509/mod_resource/content/2/01.pdf (дата звернення: 13.05.2024).
8. Jones M., Campbell B., Mortimore C. Json web token (jwt) profile for oauth 2.0 client authentication and authorization grants. – 2015. – №. rfc7523.
9. Лавріщева К. М. Програмна інженерія / К. М. Лавріщева. – К. : Академперіодика, 2008. – 319 с.
10. Ehsan A. et al. RESTful API testing methodologies: Rationale, challenges, and solution directions //Applied Sciences. – 2022. – Т. 12. – №. 9. – С. 4369.

11. Callegati F., Cerroni W., Ramilli M. Man-in-the-Middle Attack to the HTTPS Protocol //IEEE Security & Privacy. – 2009. – T. 7. – №. 1. – C. 78-81.
12. Chappell D. et al. Introducing the Azure services platform //White paper, Oct. – 2008. – T. 1364. – №. 11.

ДОДАТОК А

Звіт результатів перевірки на унікальність тексту в базі ХНУРЕ



UNICHECK
by Turnitin

Ім'я користувача:
Олійник Олена Володимирівна каф. ПІ

ID перевірки:
1016349137

Дата перевірки:
11.06.2024 20:21:52 EEST

Тип перевірки:
Doc vs Library

Дата звіту:
11.06.2024 20:22:48 EEST

ID користувача:
100012353

Назва документа: 2024_Б_ПІ_ПЗПІ_20_3_Хамінов_І_О_скорочений

Кількість сторінок: 31 Кількість слів: 3995 Кількість символів: 30907 Розмір файлу: 1.49 MB ID файлу: 1016152432

Виявлено модифікації тексту (можуть впливати на відсоток схожості)

17.7%
Схожість

Найбільша схожість: 13.8% з джерелом з Бібліотеки (ID файлу: 1016143265)

Пошук збігів з Інтернетом не проводився

17.7% Джерела з Бібліотеки 86 ..... Сторінка 33

0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

0%
Вилучень

Немає вилучених джерел

Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Підозріле форматування 12 сторінок

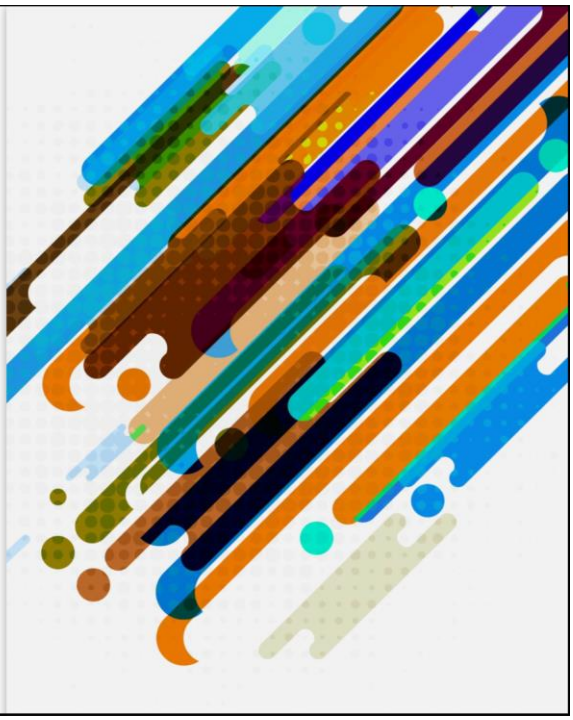
ДОДАТОК Б

Слайди презентації

ПРОГРАМНА СИСТЕМА ДЛЯ РЕКОМЕНДАЦІЇ ТА ПІДБОРУ АКТУАЛЬНИХ КУЛЬТУРНИХ ПОДІЙ. BACK-END.

- Виконав:
- ст. гр. ПЗПІ-20-3
- Хамінов І. О.

- Керівник:
- доц. кафедри ПІ Побіженко І.О.



Актуальність

- Зростання кількості культурних заходів;
- Ефективна організація та відбір заходів, які б відповідали особистим уподобанням кожної людини;
- Значення геолокації та індивідуальних уподобань

2

Об'єкт та мета роботи

Об'єкт роботи: Програмна система “Eventify” для рекомендацій культурних подій.

Мета роботи: Розробка серверної частини системи для персоналізованого вибору культурних заходів на основі уподобань та геолокації користувачів.

3

Чому ASP.NET

1. Висока продуктивність і масштабованість
2. Безпека
3. Простота у використанні та розробці
4. Широкий набір інструментів і бібліотек
5. Підтримка та спільнота

4

SQL Server

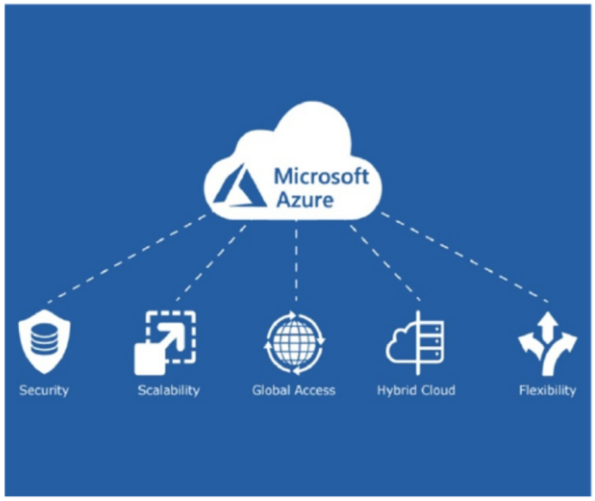
Переваги:

- Продуктивність і доступність
- Безпека
- Інтеграція та сумісність
- Сумісність з різними мовами програмування

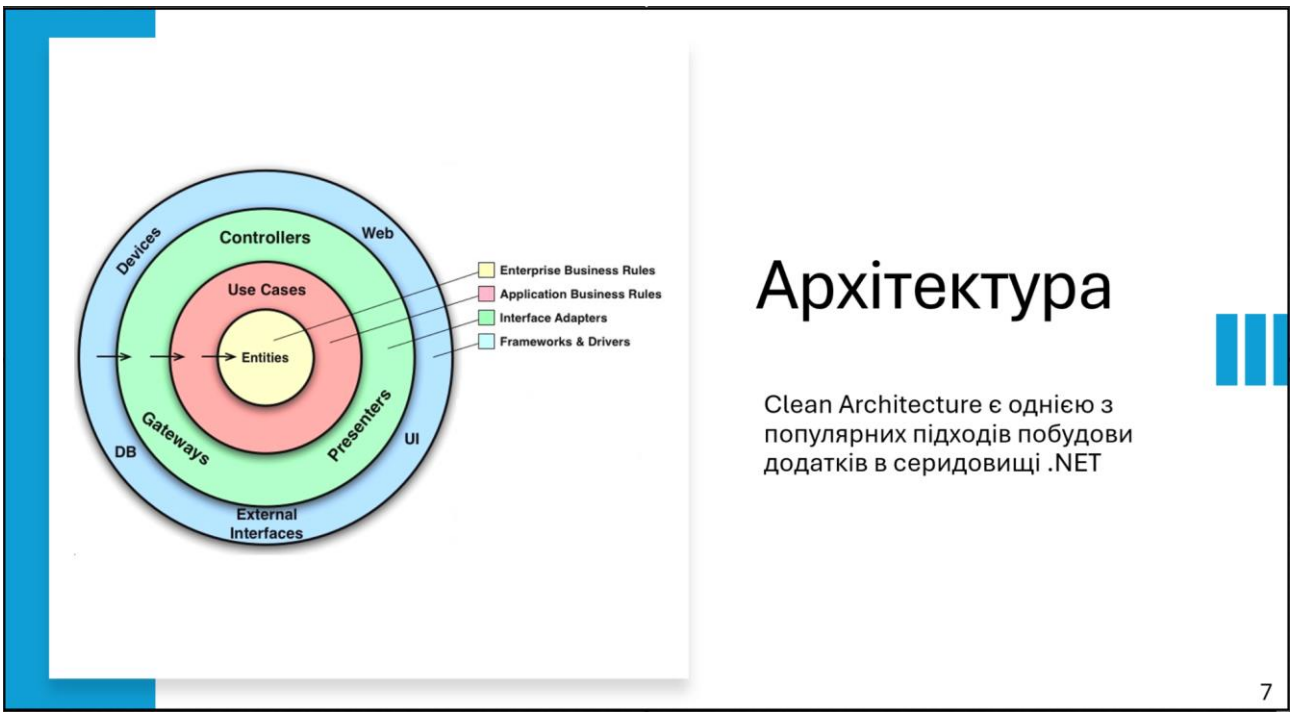
5

Azure

- Microsoft Azure — це платформа хмарних обчислень, яку компанії можуть використовувати для розробки додатків, розширення можливостей власних центрів обробки даних або в якості єдиного IT-середовища.



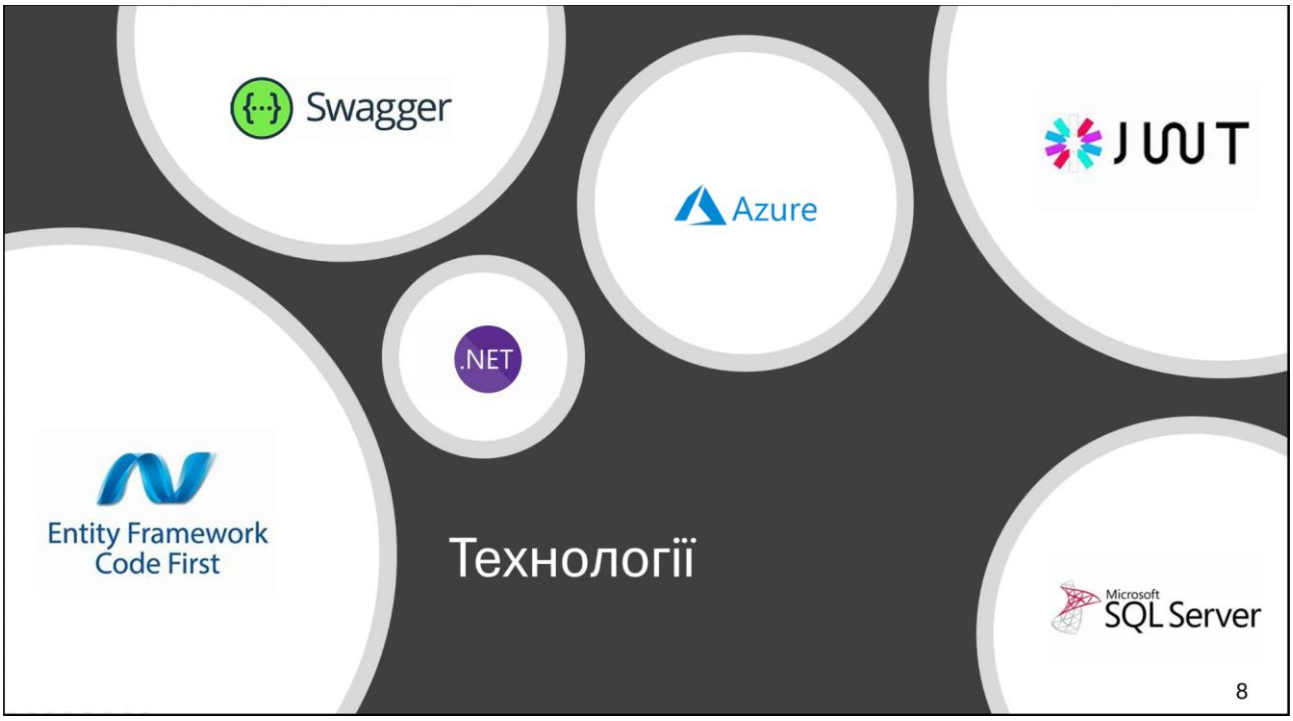
6



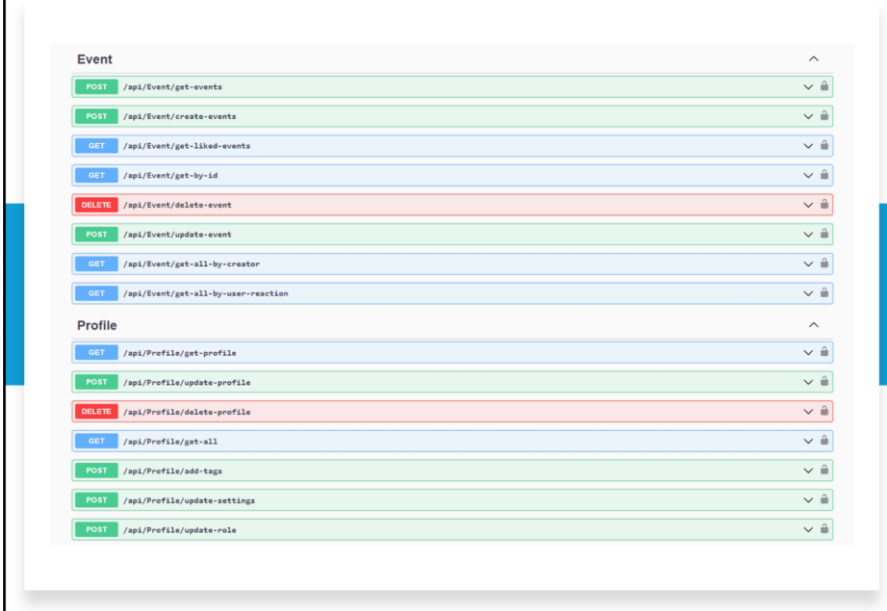
Архітектура

Clean Architecture є однією з популярних підходів побудови додатків в серидовищі .NET

7



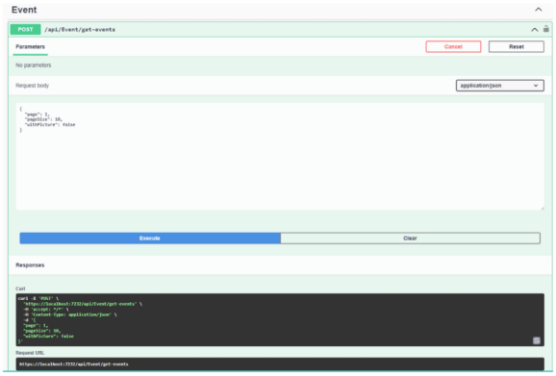
8

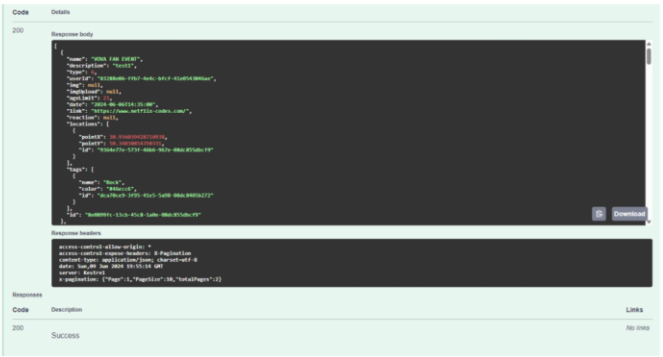


API ендпоінти

9

Тестування ендпоінтів





10

Апробація

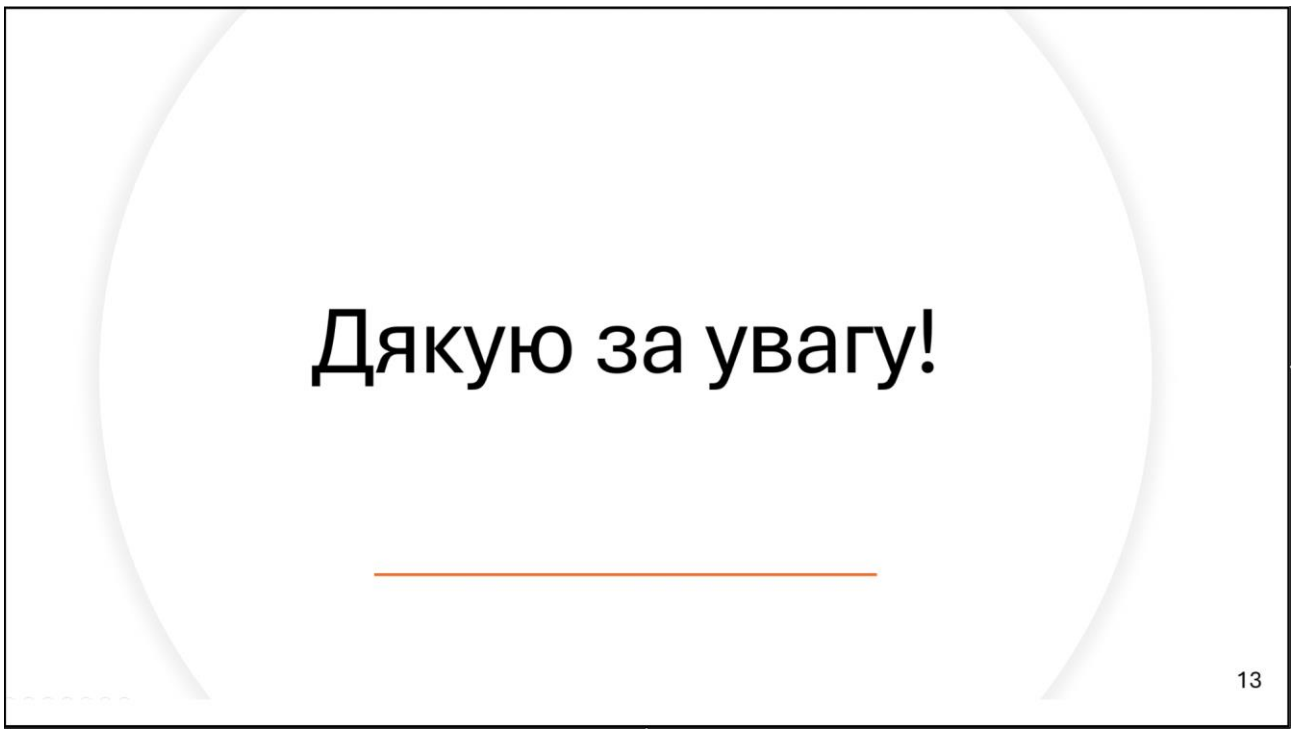
- Сафошин В. В., Хамінов І. О., Дегтяр В. Е., Побіженко І. О. Eventify: інноваційний підхід до організації та пошуку заходів з використанням ШІ. XXVIII міжнародний молодіжний форум «Радіоелектроніка та молодь у XXI столітті», конференція «Інформаційні інтелектуальні системи» - с. 544.

11

Висновки

- В результаті роботи було:
  - Визначено мету теми та сформовані завдання
  - Встановлені та налаштовані технології, що використовувалися
  - Розроблено серверну частину додатку та проведено тестування

12


Дякую за увагу!
13

ДОДАТОК В

Специфікація вимог до програмного продукту

ВСТУП

1.1 Огляд продукту

Сервіс "Eventify" з інтегрованим back-end рішенням представляє собою комплексний інструмент для користувачів, зацікавлених у відвідуванні та організації різноманітних заходів, включаючи концерти, виставки, ярмарки, фестивалі. Цей продукт включає в себе широкий спектр функцій для ефективного пошуку, відбору та участі в заходах.

Автентифікація та авторизація користувачів є ключовими елементами безпеки сервісу, що гарантує захист особистої інформації та контроль доступу до функцій залежно від ролі користувача. Ці можливості дозволяють користувачам створювати персональні профілі, налаштовувати інтереси та отримувати персоналізовані рекомендації.

"Eventify" спрощує процес відкриття та реєстрації на заходи завдяки інтуїтивно зрозумілому інтерфейсу, де користувачі можуть переглядати детальну інформацію про події, включаючи час, дату, місце проведення, а також реєструватися на них в кілька кліків. Це забезпечує зручний доступ до культурних та соціальних заходів.

Система також має функцію керування подіями, де організатори можуть створювати нові заходи, редагувати існуючі, управляти реєстраціями та відгуками відвідувачів. Це дає змогу ефективно організовувати та планувати заходи, а також збирати зворотний зв'язок для покращення майбутніх подій.

Додатково, сервіс включає можливість спілкування з ШІ асистентом, який допомагає користувачам у виборі подій на основі їхніх переваг, історії пошуку та оцінок, підвищуючи особистісний досвід використання платформи.

Загалом, "Eventify" є інноваційним рішенням для тих, хто шукає активне культурне життя, надаючи зручні інструменти для відкриття, планування та

участі в заходах, а також для організаторів, щоб ефективно управляти та просувати свої події.

1.2 Мета

Основною метою проекту "Eventify" є створення універсального сервісу, який спрощує процес пошуку, вибору та участі в різноманітних заходах, таких як концерти, виставки, ярмарки, фестивалі, для користувачів. Цей сервіс надає можливість перегляду актуальних подій на карті місцевості, їх фільтрації за інтересами та пошуку, а також рекомендації заходів, які можуть бути цікавими для конкретного користувача.

Додатково, "Eventify" забезпечує верифікацію користувачів, можливість для переходу на сторінку для замовлення квитків та запрошень на заходи, виставлення оцінок. Сервіс також включає інтеграцію зі штучним інтелектом, який виступає в ролі асистента для користувачів, допомагаючи у виборі заходів на основі їх переваг.

Таким чином, мета проекту "Eventify" полягає у забезпеченні зручного, інтуїтивно зрозумілого інструменту для відкриття нових можливостей дозвілля, сприяння культурному розвитку та соціалізації користувачів через участь у заходах, що відбуваються в їхній місцевості або в інших регіонах.

1.3 Межі

У контексті проекту "Eventify", межі визначаються функціональними та технічними аспектами системи. Проект зосереджений на наданні користувачам інструментів для пошуку, фільтрації, рекомендації та участі у заходах, а також на інтеграції з платформами для замовлення квитків та спілкуванні з ШІ асистентом. Важливими аспектами є верифікація користувачів, виставлення оцінок подіям.

Технічні обмеження системи включають обмеження щодо ресурсів сервера, які можуть вплинути на її масштабованість та продуктивність при великій кількості одночасних запитів. Це також стосується обмежень на

кількість одночасних користувачів та подій, які можуть бути оброблені системою без зниження її ефективності.

З точки зору безпеки, проект має забезпечити захист даних користувачів, включаючи аутентифікацію, авторизацію, шифрування даних та захист від несанкціонованого доступу. Це вимагає використання сучасних протоколів безпеки та методів шифрування, а також постійного моніторингу та оновлення системи безпеки.

Проект "Eventify" не займається безпосередньою організацією заходів, а лише надає платформу для їх пошуку та рекомендації. Тому, відповідальність за точність інформації про заходи, їх виконання та якість лежить на організаторах подій та представниках заходів.

1.4 Посилання

Проект "Eventify" прагне автоматизувати та оптимізувати процеси вибору та участі в культурних заходах, мінімізуючи необхідність ручного пошуку актуальних подій у різних джерелах. Інтегровані функції верифікації користувачів, бронювання квитків, виставлення оцінок, а також відображення подій на карті надають користувачам зручний і інтуїтивно зрозумілий доступ до необхідної інформації. Це значно спрощує процес взаємодії з подіями та їх відвідування, покращуючи загальний досвід користувача.

"Eventify" також забезпечує організаторам ефективний канал для залучення нової аудиторії та просування своїх заходів. Сервіс дозволяє їм швидко оновлювати інформацію про події, управляти реєстраціями та отримувати зворотний зв'язок від учасників. Це створює можливості для покращення організації заходів та підвищення їх якості.

Таким чином, "Eventify" пропонує цілісний підхід до взаємодії з культурними подіями, сприяючи культурному розвитку користувачів та ефективному управлінню подіями, що покращує досвід як відвідувачів, так і організаторів

1.5 Означення та абревіатури

Backend – це серверна частина програмного забезпечення, яка відіграє ключову роль у обробці даних, виконанні бізнес-логіки та забезпеченні безпеки. Вона управляє взаємодією з базою даних, виконує процеси аутентифікації та авторизації користувачів і забезпечує виконання операцій, важливих для функціонування системи.

Frontend - це клієнтська частина програмного забезпечення, відповідальна за інтерфейс та взаємодію з користувачем. Вона включає в себе дизайн, розмітку та скрипти сторінок, які користувачі бачать та з якими взаємодіють в браузері або через мобільні додатки.

API (Application Programming Interface) – це інтерфейс програмування застосунків, що дозволяє різним програмним компонентам спілкуватися між собою. API виступає як місток, що визначає правила та методи, за допомогою яких можна доступатися до функціоналу або даних однієї програми з іншої.

ORM (Object-Relational Mapping) – технологія, що дозволяє мапінг реляційних баз даних на об'єктно-орієнтовану модель програми. Завдяки ORM, робота з базою даних стає більш інтуїтивно зрозумілою для розробників, оскільки вони можуть оперувати об'єктами замість SQL-запитів.

HTTP (HyperText Transfer Protocol) – основний протокол передачі даних у Всесвітній мережі, який використовується для завантаження веб-сторінок з сервера на клієнтський браузер. HTTP визначає спосіб, яким повинні бути сформовані та передані запити та відповіді між клієнтом та сервером.

CRUD (Create, Read, Update, Delete) – це концепція, що описує чотири основні дії, які використовуються при роботі з даними: створення нових записів, читання (вибірка) існуючих даних, оновлення (модифікація) даних та видалення записів. Ці операції є фундаментальними для будь-яких систем, які працюють з базами даних.

LLM (Large Language Model) – це велика мовна модель, що використовує алгоритми глибокого навчання для обробки та генерації природної мови, здатна виконувати широкий спектр задач, від автоматичного перекладу до створення текстового контенту. LLM використовуються для покращення здатності комп'ютерних систем розуміти та взаємодіяти з людською мовою, надаючи можливості для створення більш продвинутих і зручних користувацьких інтерфейсів, а також для аналізу великих обсягів текстових даних.

2. ЗАГАЛЬНИЙ ОПИС

2.1 Перспективи продукту

Перспективи програмної системи “Eventify” – надання рекомендацій та підбору актуальних заходів (концертів, виставок, ярмарок, фестів) є обнадійливими і обіцяють багато можливостей для подальшого розвитку та розширення.

З урахуванням зростаючого інтересу до культурних подій та розваг, а також зростаючої потреби у персоналізованих та зручних інструментах для вибору та участі в заходах, "Eventify" може знайти своє місце в різних галузях та сприяти розвитку сфери розваг та культурно-масових заходів.

Перспективи продукту включають розширення функціональності та додавання нових модулів, щоб задовольнити специфічні потреби організацій та користувачів. Наприклад, можливості планування і розподілу ресурсів, система відгуків, інтеграція з іншими платформами або соціальними мережами, розширена аналітика та звітність.

Крім того, вдосконалення аналітичних засобів для оцінки успішності різних заходів та рекламних кампаній. Розробка розширених звітів для організаторів та партнерів для підвищення ефективності подій. Розширення географічного охоплення для підтримки подій у різних регіонах та країнах. Активне залучення до міжнародних культурних заходів та фестивалів для розширення аудиторії.

Загалом, "Eventify" має великий потенціал стати ключовим гравцем у сфері організації та участі в культурних та розважальних заходах. Розширення функціональності, посилення географічного охоплення та впровадження новітніх технологій можуть вивести продукт на новий рівень та забезпечити йому стабільне місце на ринку.

2.2 Функції продукту

Основні функції цього продукту включають:

– FE-1: управління користувачами;

- FE-2: управління заходами;
- FE-3: безпека та доступ до даних;
- FE-4: пошук і фільтрація заходів;
- FE-5: взаємодія з ШІ асистентом;
- FE-6: перегляд подій на карті.

2.3 Характеристика користувачів

Характеристики користувачів програмної системи “Eventify” – надання рекомендацій та підбору актуальних заходів (концертів, виставок, ярмарок, фестів) можуть бути різноманітними, оскільки система взаємодіє з різними типами користувачів. Основні характеристики користувачів включають:

- користувачі є основними в системі. Це люди будь-якого віку та статі. Їх характеристики можуть включати їхні інтереси, їх вік та локацію де вони мешкають;
- менеджери компаній відповідають за керування подіями. Вони мають розширені права доступу та можуть створювати, редагувати та видаляти події, керувати користувачами, аналізувати звітність та здійснювати інші адміністративні функції;
- адміністратори системи відповідають за керування користувачами та менеджерами.

2.4 Загальні обмеження

У програмній системі “Eventify” – надання рекомендацій та підбору актуальних заходів (концертів, виставок, ярмарок, фестів) існують загальні обмеження, які можуть впливати на її функціонування та використання. Деякі з цих обмежень включають.

Система повинна дотримуватися вимог безпеки, щоб захистити дані користувачів та запобігти несанкціонованому доступу. Обмеження безпеки можуть включати криптографічні вимоги, захист від атак, правила доступу до даних та контроль прав користувачів.

Також система повинна бути здатна масштабуватись для впорядкування зростаючого обсягу користувачів, проектів та завдань. Обмеження масштабованості можуть впливати на продуктивність та швидкодію системи при збільшенні навантаження.

2.5 Припущення й залежності

Наявність стабільного та надійного Інтернет-з'єднання є важливою умовою для оптимальної роботи системи. Система взаємодіє з мережею для обміну даними, автентифікації, надсилання сповіщень тощо, тому постійне з'єднання з Інтернетом є необхідним. Впевненість в стабільності та швидкості Інтернет-з'єднання є важливою умовою для ефективної роботи системи.

Доступ до бази даних: Система налагоджується від доступу до бази даних для зберігання та отримання інформації про користувачів, проекти, завдання та інші дані. Наявність та налаштування бази даних виконуються через Microsoft SQL Server. Ця база даних є критичною для функціонування системи, тому важливо впевнитися в її належному функціонуванні та оптимальному налаштуванні.

3 КОНКРЕТНІ ВИМОГИ

3.1 Вимоги до зовнішніх інтерфейсів

3.1.1 Інтерфейс користувача

Інтерфейс користувача програмної системи рекомендації та відбору актуальних подій з back-end реалізацією розроблений з метою забезпечення зручності, ефективності та легкості використання для користувачів. Він пропонує інтуїтивно зрозуміле та привабливе середовище, яке дозволяє взаємодіяти з системою та виконувати необхідні завдання.

Інтерфейс користувача може бути представлений у вигляді веб-додатку або мобільного додатка, залежно від потреб користувачів та доступних платформ. Він має чітку структуру, навігаційні елементи, що дозволяють користувачам легко орієнтуватися та виконувати різноманітні дії.

Основні характеристики інтерфейсу користувача включають:

- інтуїтивність: Забезпечення легкого розуміння та використання для користувачів без додаткового навчання.
- ефективність: Максимізація продуктивності користувачів та зменшення часу на виконання завдань.
- привабливість: Створення привабливого та естетичного вигляду інтерфейсу для залучення користувачів.
- чітка структура: Організація інформації та функціональності для легкого доступу та сприяння логічному розташуванню елементів.
- навігаційні можливості: Доступні та зрозумілі елементи навігації для швидкого переміщення користувачів по системі.

Ці аспекти сприяють позитивному враженню від використання системи та підвищують задоволення користувачів взаємодією з нею.

3.1.2 Апаратний інтерфейс

Апаратний інтерфейс програмній системі рекомендації та відбору актуальних подій з back-end реалізацією включає необхідні апаратні компоненти та засоби, які використовуються для забезпечення

функціональності системи. Оскільки ця програма зазвичай працює в онлайн-середовищі та доступна через веб-браузер або мобільний додаток, система вимагає наявності серверів для зберігання та обробки даних. Це можуть бути фізичні сервери, віртуальні сервери або хмарні платформи, які забезпечують необхідні обчислювальні та ресурси для зберігання роботи системи.

3.1.3 Програмний інтерфейс

Програмний інтерфейс (API) програмної системи рекомендації та відбору актуальних подій з back-end реалізацією визначає набір правил та протоколів, за допомогою яких різні компоненти програми можуть взаємодіяти між собою. Він дозволяє зовнішнім системам або розробникам використовувати функціональні можливості та отримувати доступ до даних програмної системи.

Програмний інтерфейс може бути реалізований у вигляді RESTful API , який базується на використанні HTTP-протоколу для передачі даних між клієнтами та сервером. Це дозволяє здійснювати стандартизовану та просту взаємодію з системою за допомогою HTTP-запитів (GET, POST, PUT, DELETE) та обміну даними у форматі JSON або XML.

3.1.4 Комунікаційний протокол

Комунікаційний протокол в програмної системи рекомендації та відбору актуальних подій з back-end реалізацією визначає правила та формати обміну даними між різними компонентами системи. Він гарантує, що передача інформації між цими компонентами відбувається швидко, надійно та безпомилково.

Один з найпоширеніших комунікаційних протоколів, який може бути використаний в такій системі, - це HTTP (Hypertext Transfer Protocol). HTTP використовується для передачі даних через мережу Інтернет та побудований на базі клієнт-серверної архітектури.

HTTP визначає формат запитів та відповідей між клієнтом та сервером. Клієнт (наприклад, веб-браузер або мобільний додаток) робить запит до

сервера, а сервер обробляє цей запит та повертає відповідь з необхідними даними.

3.1.5 Обмеження пам’яті

Обсяг оперативної пам'яті, доступний для виконання програм та зберігання даних, може бути обмеженим. Залежно від розміру доступної пам'яті, система може вміщувати обмежену кількість користувачів та обробляти лише обмежену кількість даних.

Наявність дискового простору для зберігання даних також може бути обмеженою. Дисковий простір використовується для зберігання файлів, баз даних, резервних копій та інших ресурсів. Обмеження дискового простору може впливати на масштабність системи та можливість зберігання великих обсягів даних.

3.1.6 Операції

Користувачі можуть шукати події та заходи що їх цікавлять. Це включає визначення назви, типу, місця, часу та інших важливих атрибутів події.

Система може надавати можливості для створення компаніям або угрупуванням, що мають права на заходи, нові події та заходи. Також система надає можливість утворення звітності для організаторів.

3.2 Атрибути програмного продукту

3.2.1 Надійність

Надійність в програмній системі рекомендації та відбору актуальних подій з back-end реалізацією є важливою характеристикою, оскільки вона забезпечує безперебійну та стабільну роботу системи. Надійність означає, що система працює так, як очікується, і надійно обробляє запити користувачів, запобігаючи можливим збоям чи відмовам.

Для досягнення надійності, система повинна мати вбудований механізм для обробки помилок та винятків, які можуть виникати під час роботи. Це

допомагає виявляти та вирішувати проблеми, що впливають на роботу системи, забезпечуючи її стабільну та надійну функціональність.

Також важливо мати механізми резервного копіювання, що забезпечують збереження та захист даних системи. Це може включати регулярне резервне копіювання даних, створення резервних копій баз даних та інших важливих ресурсів, що гарантує можливість відновлення системи в разі втрати чи пошкодження інформації.

3.2.2 Доступність

Система повинна мати реактивний дизайн, який автоматично адаптується до різних типів пристроїв, таких як комп'ютери, планшети або смартфони. Це дозволяє користувачам зручно взаємодіяти з системою на будь-якому пристрої, незалежно від їхнього розміру екрану. Такий підхід сприяє покращенню зручності використання та задоволенню користувачів, забезпечуючи оптимальний інтерфейс на різних пристроях.

3.2.3 Безпека

Система повинна володіти механізмами аутентифікації для перевірки ідентичності користувачів та авторизації їх доступу до різних функцій системи. Це може включати використання паролів, рівнів доступу та управління правами користувачів.

З метою захисту конфіденційності даних, особливо під час їх передачі між клієнтом та сервером, використовується шифрування. Використання протоколів шифрування, таких як SSL (Secure Sockets Layer) або TLS (Transport Layer Security), сприяє забезпеченню безпеки під час передачі даних.

Система також повинна мати заходи безпеки для захисту від зламу та зловмисницьких атак, таких як введення шкідливого коду, SQL-ін'єкції, переповнення буфера тощо. Це може включати застосування механізмів фільтрації введення користувачів, валідацію даних та захист від переповнення, що сприяє забезпеченню високого рівня безпеки системи.

3.2.4 Супроводжуваність

Система повинна мати належну документацію, яка точно описує її архітектуру, конфігурацію, процедури установки, налаштування та супроводження. Це сприяє швидкому розумінню та вирішенню проблем розробниками та адміністраторами системи, а також дозволяє внесення необхідних змін.

Крім того, система повинна мати механізми для регулярних оновлень та встановлення патчів з метою виправлення відомих помилок, усунення вразливостей та покращення функціональності. Це забезпечує надійність та стійкість системи, а також дозволяє впровадження нових функцій та вдосконалень.

3.2.5 Переносимість

Переносимість в програмній системі, щодо рекомендацій та відбору актуальних подій у back-end реалізації, передбачає здатність системи працювати на різних платформах та середовищах без значних змін. Це досягається шляхом використання платформо-незалежних технологій, стандартизованих фреймворків та уникання глибокої залежності від конкретних платформ або СКБД. Такий підхід гарантує гнучкість та ефективність у розгортанні системи на різних середовищах без великих зусиль.

3.2.6 Продуктивність

Висока продуктивність у програмній системі рекомендації та відбору актуальних подій з реалізацією back-end є суттєвим компонентом і включає в себе оптимізацію коду, масштабування та ефективне використання ресурсів. Розробка коду, який оптимізовано виконує завдання, разом з масштабуванням системи для ефективного розподілу навантаження та правильного використання ресурсів, сприяє високій продуктивності системи та задоволенню потреб користувачів.

3.2.7 Вимоги бази даних

Для програмної системи рекомендації та підбору актуальних подій із back-end реалізацією вимоги до бази даних включають в себе належну структуру, нормалізацію даних та підтримку необхідних запитів та операцій. Забезпечення безпеки даних є пріоритетом і вимагає використання механізмів авторизації, шифрування та контролю доступу. Оптимізація бази даних для досягнення високої швидкодії та ефективності виконання запитів також є ключовою вимогою.

ДОДАТОК Г

Тези XXVIII міжнародного молодіжного форуму «Радіоелектроніка та молодь у XXI столітті»

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
РАДІОЕЛЕКТРОНІКИ

МАТЕРІАЛИ XXVIII МІЖНАРОДНОГО МОЛОДІЖНОГО ФОРУМУ

«РАДІОЕЛЕКТРОНІКА ТА МОЛОДЬ У XXI СТОЛІТТІ»

16 – 18 квітня 2024 р.

Том 6

КОНФЕРЕНЦІЯ
«ІНФОРМАЦІЙНІ ІНТЕЛЕКТУАЛЬНІ СИСТЕМИ»
INFORMATION INTELLIGENT SYSTEMS

Харків 2024

28-й Міжнародний молодіжний форум «Радіоелектроніка та молодь у XXI столітті». Зб. матеріалів форуму. Т. 6., – Харків: ХНУРЕ. 2024. – 958 с.

У збірнику представлено матеріали доповідей учасників 28-го Міжнародного молодіжного форуму «Радіоелектроніка та молодь у XXI столітті».

Для науковців, викладачів, парктичних працівників, студентів, а також широкого кола читачів, які цікавляться цією проблематикою.

Відповідальність за зміст поданого матеріалу несе його автор.

Видання підготовлено факультетом комп'ютерних наук Харківського національного університету радіоелектроніки

61166, Україна, Харків, просп. Науки, 14
тел./факс: (057) 7021397

E-mail: mref21@nure.ua

ISBN ISBN 978-966-659-396-5
DOI 10.30837/IYF.IIS.2024

Програмний комітет конференції

Федорович О.Є.	д.т.н., проф., зав. каф. Комп'ютерних наук та інформаційних технологій (КНІТ), Національний аерокосмічний університет ім. М.Є. Жуковського "Харківський авіаційний інститут", Лауреат Державної премії України.
Субботін С.А.	д.т.н., проф., зав. каф. Програмних засобів, Запорізький національний технічний університет, Україна.
Петренко М.Г.	д.т.н., проф., Інститут кібернетики імені В.М. Глушкова НАН України.
Стасюк О.І.	д.т.н., проф. Державний економіко-технологічний університет транспорту, Україна.
Єрохін А.Л.	проф., декан ф-ту ХНУРЕ, м. Харків, Україна.
Філатов В.О.	проф., зав. каф. ХНУРЕ, м. Харків, Україна.
Петров К.Е.	проф., зав. каф. ХНУРЕ, м. Харків, Україна.
Дудар З.В.	проф., зав. каф. ХНУРЕ, м. Харків, Україна.
Гребеннік І.В.	проф., зав. каф. ХНУРЕ, м. Харків, Україна.
Дейнеко Ж.В.	проф., зав. каф. ХНУРЕ, м. Харків, Україна.

УДК 004.89

EVENTIFY: ІННОВАЦІЙНИЙ ПІДХІД ДО ОРГАНІЗАЦІЇ ТА ПОШУКУ ЗАХОДІВ З ВИКОРИСТАННЯМ ШІ

Сафошин В. В., Хамінов І. О., Дегтяр В. Е.
Науковий керівник – к.т.н., доц. Побіженко І. О.
Харківський національний університет радіоелектроніки, каф. ПІ
м. Харків, Україна
e-mail: volodymyr.safoshyn@nure.ua, illia.khaminov@nure.ua,
e-mail: vladyslav.dehtiar@nure.ua

In an era of rapid cultural evolution, the challenge of effectively organizing and selecting events that cater to individual preferences has become paramount. "Eventify" revolutionizes event organization in the digital age by leveraging artificial intelligence to tailor cultural event recommendations to individual preferences. This platform not only makes finding events effortless but also signifies a new chapter in personalized cultural experiences, ensuring users are matched with events that resonate with their interests. "Eventify" exemplifies the transformative potential of artificial intelligence [1] (AI) in enriching cultural engagement and streamlining the event selection process.

В епоху стрімкого розвитку культурного життя, актуальним стає питання ефективної організації та відбору заходів, які б відповідали особистим уподобанням кожної людини. Доступними рішеннями є різні телеграм канали, які повідомляють користувачів про різні події. В таких джерелах інформація про події не згрупована за видом та розташуванням, тож користувач може загубити цікаві йому події поміж багатьох інших. Також є різні сайти, що пропонують тільки специфічний спектр подій і не охоплюють всі вподобання користувача, що призводить до того що потрібно використовувати багато різних джерел. В цьому контексті сервіс "Eventify" пропонує революційний підхід, що базується на використанні передових технологій штучного інтелекту (ШІ) для аналізу великих обсягів даних про культурні події, їх учасників та переваги користувачів. Удосконалюючи свою місію щодо пошуку та організації культурних заходів, платформа "Eventify" залучатиме передові методи штучного інтелекту, які включають великі мовні моделі [2] (LLMs), ChatGPT[3].

Дослідження спрямоване на створення інноваційної програмної системи, яка за допомогою веб- та мобільного додатків пропонуватиме користувачам персоналізовані рекомендації щодо різноманітних заходів, виходячи з їхніх переваг. Це дозволить користуватися однією платформою для пошуку всіх видів культурних заходів, а не багатьма різними сторонніми ресурсами.

Впродовж дослідження аналогічний рішень нашого додатку, було виявлено те, що жоден з них не використовує ніякі засоби для оптимізації та комфорту у виборі культурних заходів. Враховуючи це, ми прийшли до

висновку, що використання штучного інтелекту стане ідеальним рішенням для покращення користувацького досвіду.

Великі мовні моделі представляють собою клас алгоритмів машинного навчання, здатних аналізувати та генерувати природну мову [4] з високим рівнем складності та точності. У контексті "Eventify" використовуються LLMs для обробки об'ємних текстових даних, пов'язаних із культурними заходами.

ChatGPT – це одна з передових LLMs, розроблена OpenAI. Ця модель базується на технології трансформерів і здатна генерувати природні відповіді в контексті діалогу з користувачем [5]. Під час розробки нашого додатку ми обрали цю LLM як ключовий компонент, оскільки він пропонує унікальні переваги для покращення користувацького досвіду у виборі культурних подій. ChatGPT відіграє ключову роль у нашому додатку, оскільки здатний до глибокого аналізу діалогів, що дозволяє вловлювати нюанси запитів користувачів та відповідати на них з надзвичайною точністю. Також він здатний до поглибленої взаємодії з користувачем, що буде сприяти більш детальному розумінню їхніх уподобань, що дозволяє надавати влучні рекомендації щодо заходів, які відповідають їхнім інтересам.

Отже, наш проект підкреслює значення інтеграції штучного інтелекту в області організації заходів, демонструючи, як сучасні технології можуть радикально змінити взаємодію між організаторами та учасниками. Також за результатами нашого дослідження було проаналізовано усі доступні рішення цієї проблеми та створено інноваційний підхід її вирішення шляхом розробки додатку “Eventify” з використанням передових технологій штучного інтелекту.

Список використаних джерел:

1. Barstow, David. "Artificial intelligence and software engineering." Exploring artificial intelligence. Morgan Kaufmann, 1988. 641-670.

2. Chang, Yupeng, et al. "A survey on evaluation of large language models." ACM Transactions on Intelligent Systems and Technology (2023).

3. Wu, Tianyu, et al. "A brief overview of ChatGPT: The history, status quo and potential future development." IEEE/CAA Journal of Automatica Sinica 10.5 (2023): 1122-1136.

4. Erdem, Erkut, et al. "Neural natural language generation: A survey on multilinguality, multimodality, controllability and learning." Journal of Artificial Intelligence Research 73 (2022): 1131-1207.

5. Sharonova, N., Kyrychenko, I., Gruzdo, I., Tereshchenko, G. (2022) Generalized Semantic Analysis Algorithm of Natural Language Texts for Various Functional Style Types CEUR Workshop Proceedings, 2022, 3171, pp. 16–26.