МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Сумський державний університет

Факультет електроніки та інформаційних технологій

Кафедра комп'ютерних наук

«До захисту допущено»

В.о. завідувача кафедри

_____ Ігор ШЕЛЕХОВ

(підпис)

22 червня 2023 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня бакалавр

зі спеціальності 122 – Комп'ютерних наук,

освітньо-професійної програми «Інформатика»

на тему: «Інформаційна веб-система нових місць для знайомств.»

здобувача групи ІН-94-1 Шкарупи Даніїла Олексійовича.

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

_____ Даніїл Шкарупа

(підпис)

Керівник

кандидат фізико-математичних наук,

старший викладач кафедри Шовкопляс О.А.

Суми – 2023

Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра комп'ютерних наук

«Затверджую»
В.о. завідувача кафедри
_____ Ігор ШЕЛЕХОВ
(підпис)

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня бакалавра

зі спеціальності 122 - Комп'ютерних наук, освітньо-професійної програми «Інформатика»
здобувача групи ІН-94-1 Шкарупи Даніїла Олексійовича

1. Тема роботи: «Інформаційна веб-система нових місць для знайомств.»
затверджую наказом по СумДУ від *«01» червня 2023 р. № 0475-VI*
2. Термін здачі здобувачем кваліфікаційної роботи *до 22 червня 2023 року*
3. Вхідні дані до кваліфікаційної роботи_____
4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити) *1) Аналіз проблеми предметної області, постановка й формування завдань дослідження. 2) Огляд технологій, що використовуються для побудови веб-системи таких типів. 3) Розроблення системи з побудови соціально-професійних зв'язків. 4) Аналіз результатів.*
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)_____
6. Консультанти до проєкту (роботи), із значенням розділів проєкту, що стосується їх

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання «____» _____ 20 ___ р.

Завдання прийняв до виконання _____ Керівник _____
(підпис) (підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Термін виконання	Примітка
1	*Аналіз проблеми предметної області, постановка й формування завдань дослідження*		
2	*Огляд технологій, що використовуються для підтримки й побудови соціально-професійних зв'язків*		
3	*Розробка системи з побудови й підтримки соціально-професійних зв'язків*		
4	*Аналіз отриманих результатів*		
5	*Оформлення пояснювальної записки до кваліфікаційної роботи*		

Здобувач вищої освіти _____ Керівник _____
(підпис) (підпис)

АНОТАЦІЯ

Записка: 40 стр., 25 рис, 13 використаних джерел.

Обґрунтування актуальності теми роботи – тема роботи є актуальною, оскільки зростає популярність використання Інтернету для знаходження співрозмовників, нових знайомств та можливостей для відпочинку. Віддалений доступ до онлайн-платформ стає зручним інструментом для задоволення потреб сучасного суспільства у спілкуванні та розвитку у своїх інтересах.

Об'єкт дослідження – процес знаходження нових знайомств, ідей, відпочинку та спілкування через веб-додаток в умовах обмежень на зустрічі та події, спричинених вірусною інфекцією та іншими чинниками.

Мета роботи – розробка веб-додатка, який надасть можливість користувачам знаходити нові знайомства, заходи та місця для відпочинку. Даний додаток буде сприяти спілкуванню та розвитку у своєму напрямку, надавати доступ до інформації про події та допомагати знаходити фестивалі, конкурси та чемпіонати.

Методи дослідження – аналіз потреб користувачів, дослідження ринку та конкурентів, проведення фокус-груп, а також збір та аналіз даних веб-аналітики.

Результати – виконано аналіз предметної області та аналогічних сайтів, сформульовано цілі і вирішено проблеми. Розроблено інформаційну систему з функціональним інтерфейсом, яка забезпечує пошук нових знайомств та місць. Проведено успішне тестування, а розроблені веб-ресурси вже розміщені на хостингу.

ІНФОРМАЦІЙНА СИСТЕМА, ГЕОЛОКАЦІЯ, МІСЦЕПОЛОЖЕННЯ,
ANGULAR, TYPESCRIPT, .NET

ЗМІСТ

ВСТУП

Тема роботи є актуальною, оскільки зростає популярність використання Інтернету для знаходження співрозмовників, нових знайомств та можливостей для відпочинку. Віддалений доступ до онлайн-платформ стає зручним інструментом для задоволення потреб сучасного суспільства у спілкуванні та розвитку у своїх інтересах, особливо в умовах обмежень на зустрічі та події, спричинених вірусною інфекцією та іншими чинниками.

Основною метою даної роботи є розробка веб-додатка, який надасть можливість користувачам знаходити нові знайомства, заходи та місця для відпочинку. Даний додаток буде сприяти спілкуванню та розвитку у своєму напрямку, надавати доступ до інформації про події та допомагати знаходити фестивалі, конкурси та чемпіонати. Для досягнення поставленої мети будуть виконані наступні завдання: аналіз потреб користувачів, дослідження ринку та конкурентів, проведення фокус-груп, а також збір та аналіз даних веб-аналітики.

У процесі роботи було проведено аналіз предметної області та аналогічних сайтів, сформульовано цілі і вирішено проблеми. Результатом є розроблена інформаційна система з функціональним інтерфейсом, яка забезпечує пошук нових знайомств та місць для відпочинку. Крім того, було проведено успішне тестування, а розроблені веб-ресурси вже розміщені на хостингу.

Виконання даної роботи дозволить задовольнити потреби сучасного суспільства у знаходженні нових знайомств, розвитку та забезпеченні можливостей для відпочинку через веб-додаток.

Більшість цікавих та амбіційних стартапів, а саме вісімдесят відсотків, не досягли свого успіху через невміння ефективно розпоряджатися ресурсами.

Таким чином, успішність проєкту безпосередньо залежить від того, наскільки добре розроблений і реалізований бізнес-план.

1 ІНФОРМАЦІЙНІЙ ОГЛЯД

1.1 Огляд проблемної області

Люди хочуть бачити поряд із собою однодумців та спілкуватися з людьми на одній хвилі, бо вони нас зрозуміють з півслова, тому що з ними можна бути собою і не треба вдавати, тому що з ними весело і цікаво. Кожен може перерахувати ще багато різних "бо". Тут все й так зрозуміло – однодумці потрібні та важливі.

Зустрічі з людьми онлайн та пошук нових місць не призначені тільки для додатків для знайомств. Насправді, це один із найпростіших способів почати пошук нових друзів та інтересів.

Є багато способів налагодити зв’язки в Інтернеті. Навіть якщо ви не зацікавлені в тривалих стосунках з онлайн-друзями, ви завжди можете використовувати Інтернет, щоб заводити друзів у реальному світі, дізнаючись про місцеві події та інші заходи у вашому регіоні, які можуть допомогти вам зустрітися з людьми, але насправді пошук нових знайомств та місць – це не легка задача для сучасного користувача, бо всі ці сервіси для знаходження однодумців виконують цю функцію некоректно, тобто вимагають грошей, розповсюджують фейки, мають застарілі та непрацюючі компоненти і тому подібне (рис. 1.1).

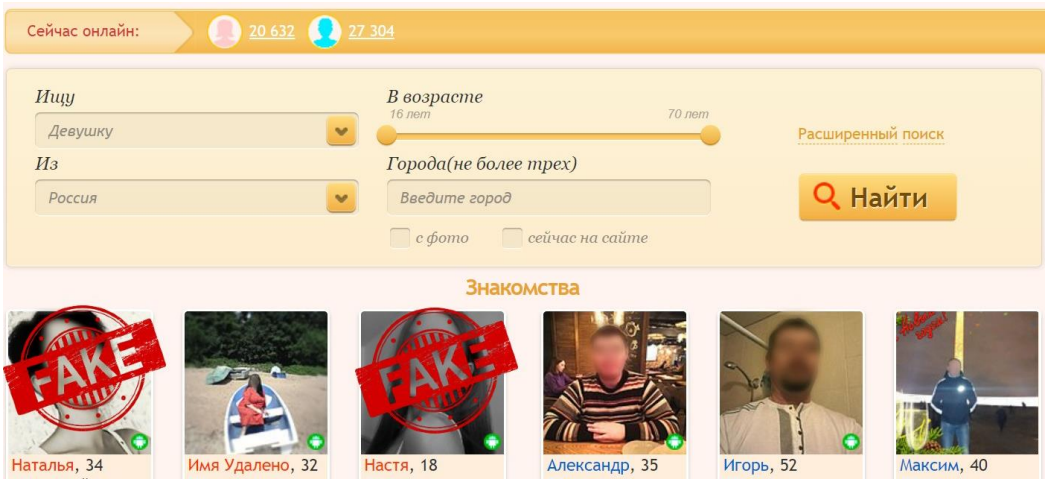

Рисунок 1.1 – Сторінка сайту для знайомств

Проте це зовсім не означає, що варто боятися всіх, хто сидить по той бік монітора. Порядних людей там набагато більше, ніж ошуканців. Чоловіки та жінки з різних країн справді зацікавлені у пошуку нових місць та знайомств, багато хто з них прийшов із серйозними намірами. Попри все пам'ятати про власну безпеку потрібно завжди.

1.2 Огляд Web-ресурсів

На сьогоднішній день існує невелика кількість подібних веб-ресурсів, щоб можна було запропонувати своє місце на карті з помітками, знайти потрібний заклад з певними заходами та однодумцями.

Перед розробкою веб-додатку був проведений огляд вже існуючих сайтів, таких як:

– cityscale.com.ua;
– vibemap.com;
– tinder.com.

Кожен із наведених вище сайтів має свої характерні риси, які виділяють його від інших.

Наприклад, cityscale.com.ua за короткий проміжок часу користувач може знайти на карті необхідну інформацію про заходи, оренди, аптеки та злочинності. Проте не може дати широкого вибору місць та заходів, а також має недоліки у вигляді застарілих даних (рис. 1.2).

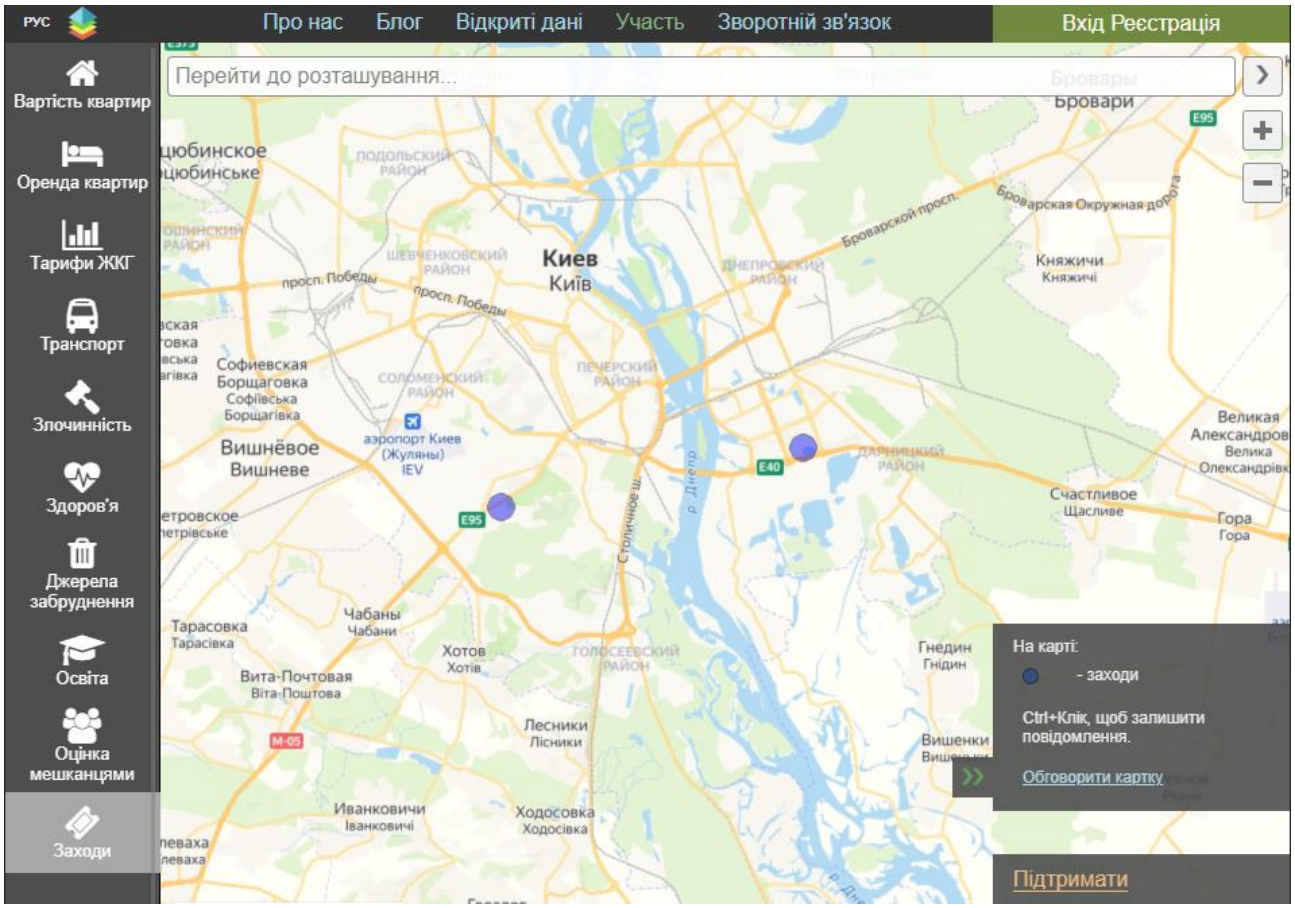

Рисунок 1.2 – Сторінка сайту cityscale.com.ua

Другий додаток vibemap.com схожий на cityscale.com.ua, але відрізняється, по-перше тим, що інформація розповсюджується по всьому світу, по-друге він має привабливий дизайн, що зацікавлює користувачів, також є більш зручним, а по-третє сайт містить більше інформації та гнучкий до пошуку і це значить, що він відповідає користувачам, хто точно знає що йому треба, але має великий недолік – швидкість обробки даних дуже велика (рис. 1.3).

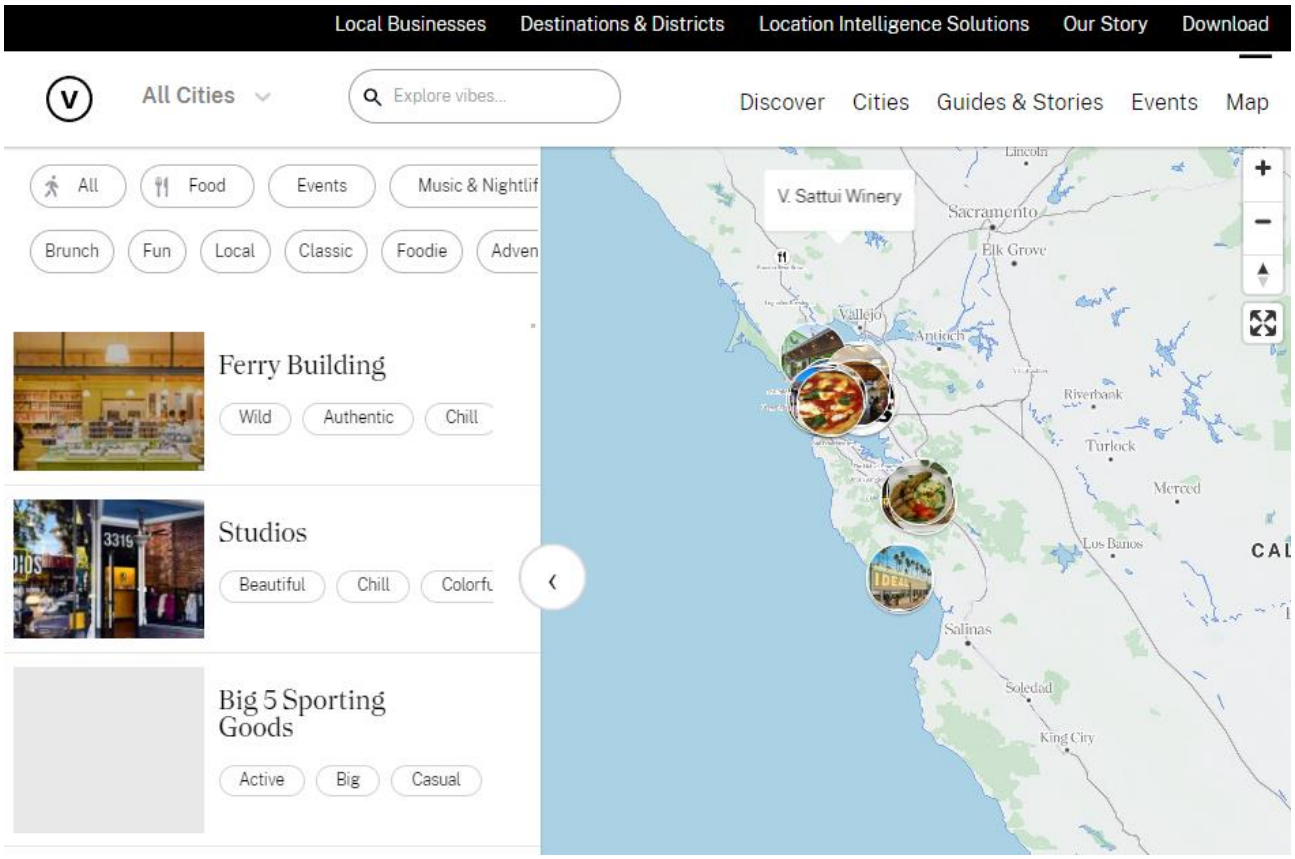

Рисунок 1.3 — Сторінка сайту vibemap.com

Що до tinder.com потрібно зазначити, що сайт має приємний дизайн та гарну швидкість, але має одну основну функцію – пошук зацікавлених у зустрічах або спілкуванні людей (рис. 1.4).

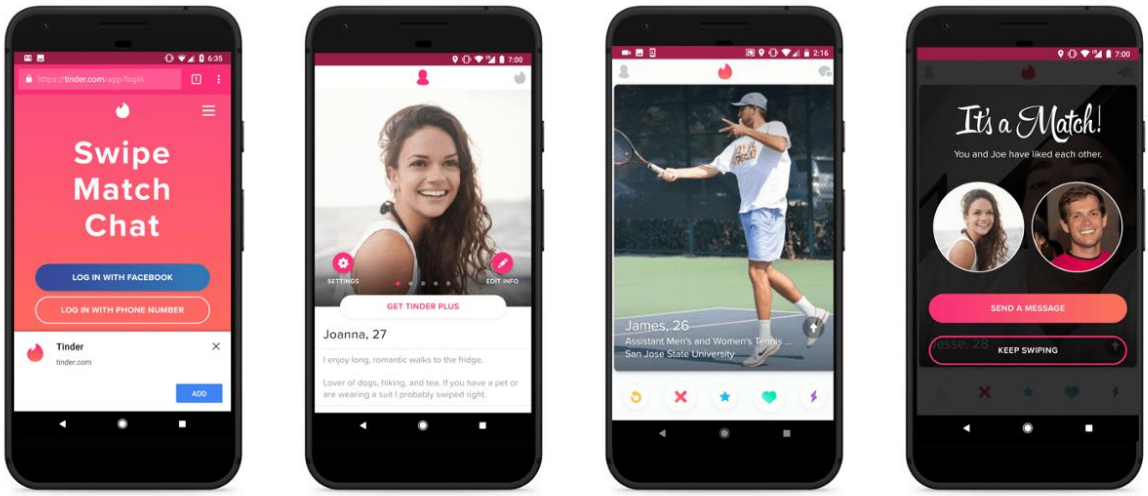

Рисунок 1.4 – Сторінка сайту tinder.com

1.3 Актуальність розробки

Багато людей сьогодні відчувають нестачу спілкування та відпочинку. І щоб не шукати своїх нових однодумців на вулиці, зупинці, в метро чи кафе, користуються можливістю знаходження на сайті. Сидячи вдома в кріслі за чашкою чаю або кави можна знайти собі гарного співрозмовника або місце для веселого проведення часу.

Усе більше і більше з’являються осіб, які хочуть знайти нові знайомства, заходи, щоб розвиватися у своєму напрямку або ж просто гарно провести час.

Зараз дедалі більше бізнес-процеси йдуть в онлайн. Рідко зустрічаються візитки без адреси сайту чи електронної пошти.

Більшість івентів, які проходять у різних місцях, не мають великих афіш або реклам. Мережа стає основним джерелом лідогенерації та інструментом розповсюдження інформації.

Оскільки цей рік вже по праву може вважатися наймізернішим на заходи та знайомства через перенесення майже всіх очікуваних подій через вірусну інфекцію та війну, тому більшість намагається їх проводити без оголошення та реклами, що ускладнює пошук подібних речей та людей.

Також у цей складний час для людства, яке включає дуже багато футболістів, гімнастів, борців, а одним словом – спортсменів, що потребують фестивалів, конкурсів та чемпіонатів, щоб і надалі розвиватися у своїй улюбленій сфері й мати нагоду спілкуватися з такими як вони. За допомогою веб-додатка буде зручно та ефективно знаходити будь-які знайомства, ідеї, відпочинок та багато чого іншого.

Для цього потрібно, по-перше, лиш мати телефон або комп’ютер з доступом до інтернету, по-друге, бути відкритим до нових пригод та зробити всього один клік і вже маєте змогу зустрітися з новими товаришами, відкрити для себе нові емоції та місця.

1.4 Постановка задачі

Створити ефективну інформаційну систему, яка б сприяла поліпшенню ефективності знаходження нових знайомств, заходів та візуальну й серверну частину (Front-end та Back-end) з необхідним функціоналом та адмініструванням.

Потрібно створити такі сайти:

– Лендінг – перше, що зустрічає користувач, який користується продуктом цього проєкту;

– Завантаження значків – додаток для запропонування нових ідей, тегів, зображень;

– Публічна сторінка – для перегляду мапи, сторінки юзерів та івентів, які відображаються з тегами;

– Адмін панель – веб-сайт для адміністрування заходами, постами, запропонованих іконок та їх тегів.

Розглянути можливі методології керування проєктом. Розробити макет для основних функціональних блоків кожного з додатків. Визначити основний стек технологій. Розділити виконавчі ролі у команді. Реалізувати аналіз виконаної роботи та надавання оцінки майбутнім завданням. Створити сторінки проєкту у соціальних мережах Facebook та Instagram.

2 ВИБІР МЕТОДУ РІШЕННЯ

2.1 Вибір програмної реалізації

При розробці веб-додатків, а саме передньої їх частини, було використано такий стек технологій:

- Angular 12 – 14;
- HTML5;
- CSS3(SCSS);
- TypeScript;
- Bootstrap.

У той час як серверну частину було розроблено на C# та платформі .NET.

HTML

HTML розшифровується як HyperText Markup Language . Це стандартна мова розмітки для створення веб-сторінок. Він дозволяє створювати та структурувати розділи, абзаци та посилання за допомогою елементів HTML (будівельних блоків веб-сторінки), таких як теги та атрибути.

HTML має багато варіантів використання, а саме:

- Веб-розробка. Розробники використовують HTML-код для проєктування того, як браузер відображає елементи веб-сторінки, такі як текст, гіперпосилання та медіа-файли;
- Інтернет-навігація. Користувачі можуть легко переходити та вставляти посилання між пов’язаними сторінками та веб-сайтами, оскільки HTML широко використовується для вставлення гіперпосилань;
- Веб-документація. HTML дає змогу впорядковувати та форматувати документи, подібно до Microsoft Word.

Також варто зазначити, що HTML не вважається мовою програмування, оскільки він не може створювати динамічні функції. Тепер він вважається офіційним веб-стандартом. Консорціум Всесвітньої павутини (W3C) підтримує та розробляє специфікації HTML, а також забезпечує регулярне оновлення.

Звичайний веб-сайт містить кілька різних HTML-сторінок. Наприклад, домашня сторінка, сторінка про програму та сторінка контактів матимуть окремі файли HTML. Документи HTML – це файли, які закінчуються розширенням .html або .htm. Веб-браузер читає файл HTML і відображає його вміст, щоб користувачі Інтернету могли його переглядати. Усі сторінки HTML мають ряд елементів HTML, що складаються з набору тегів і атрибутів. Елементи HTML є будівельними блоками веб-сторінки. Тег повідомляє веб-браузеру, де починається і закінчується елемент, тоді як атрибут описує характеристики елемента.

Три основні частини елемента:

– Відкриваючий тег – використовується для вказівки, де елемент починає діяти. Тег загорнутий у відкриваючі та закриваючі кутові дужки. Наприклад, використовуйте початковий тег , щоб створити абзац;

– Контент – це вихідні дані, які бачать інші користувачі;

– Закриваючий тег – те саме, що відкриває тег, але з похилою рискою перед назвою елемента. Наприклад, для завершення абзацу або параграфу.

CSS(SCSS)

Каскадні таблиці стилів , які люблять називати CSS, — це проста мова дизайну, призначена для спрощення процесу створення веб-сторінок, які виглядають презентабельно. CSS керує зовнішнім виглядом веб-сторінки.

Використовуючи CSS, можна керувати кольором тексту, стилем шрифтів, інтервалом між абзацами, розміром і розташуванням стовпців, фоновими зображеннями або кольорами, дизайном макета, варіаціями відображення для різних пристроїв і розмірів екрана, а також безліч інших ефектів.

CSS простий у вивченні та розумінні, але він забезпечує потужний контроль над представленням документа HTML. Найчастіше CSS поєднується з мовами розмітки HTML або XHTML.

Переваги CSS:

– CSS економить час − Ви можете написати CSS один раз, а потім повторно використовувати той самий аркуш на кількох HTML-сторінках. Ви можете визначити стиль для кожного елемента HTML і застосувати його до будь-якої кількості веб-сторінок;

– Сторінки завантажуються швидше − якщо ви використовуєте CSS, вам не потрібно щоразу писати атрибути тегів HTML. Просто напишіть одне правило CSS для тегу та застосуйте його до всіх входжень цього тегу. Тож менше коду означає швидший час завантаження;

– Просте обслуговування – Щоб внести глобальні зміни, просто змініть стиль, і всі елементи на всіх веб-сторінках оновляться автоматично.

– Стилі кращі за HTML – CSS має набагато ширший набір атрибутів, ніж HTML, тому ви можете надати своїй HTML-сторінці набагато кращий вигляд порівняно з атрибутами HTML;

– Сумісність із кількома пристроями − таблиці стилів дозволяють оптимізувати вміст для кількох типів пристроїв. Використовуючи той самий HTML-документ, можна представити різні версії веб-сайту для кишенькових пристроїв, таких як КПК і мобільні телефони, або для друку;

– Глобальні веб-стандарти − Зараз атрибути HTML застаріли, і рекомендується використовувати CSS. Тому було б гарною ідеєю почати використовувати CSS на всіх сторінках HTML, щоб зробити їх сумісними з майбутніми браузерами.

SCSS : Syntactically Awesome Style Sheet — це надмножина CSS. SCSS — це вдосконалена версія CSS. SCSS був розроблений Hampton Catlin і розроблений Chris Eppstein і Natalie Weizenbaum. Завдяки розширеним функціям його часто називають Sassy CSS. Файл SCSS має розширення .scss.

Різниця між SCSS та CSS:

– SCSS містить усі функції CSS і містить більше функцій, яких немає в CSS, що робить його хорошим вибором для розробників;

– SCSS має багато розширених функцій;

– SCSS пропонує змінні, ви можете скоротити свій код за допомогою змінних. Це велика перевага перед звичайним CSS;

– Знання SCSS допоможе налаштувати Bootstrap;

– Додає функцію @import, яка дозволяє імпортувати налаштовані файли SCSS.

TypeScript

TypeScript є надмножиною JavaScript, що означає, що він містить усі функції JavaScrip. Таким чином, будь-яка програма, написана на дійсному JavaScript, також працюватиме належним чином у TypeScript.

TypeScript пропонує нам більше контролю над нашим кодом за допомогою анотацій типів, інтерфейсів і класів.

TypeScript був створений Microsoft і випущений у 2012 році після двох років розробки. Він був створений для забезпечення додаткової статичної перевірки типу, яка була б особливо корисною під час розробки великомасштабних програм. Цікаво, що однією з причин, чому Microsoft розробила TypeScript, було те, що їхні внутрішні команди мали проблеми з масштабуванням JavaScript для власних проєктів Microsoft, зокрема команда, яка працювала над Bing Maps.

JavaScript типізується динамічно. Таким чином, програми, написані на JavaScript, не знають типу даних змінної, доки цій змінній не буде присвоєно значення під час виконання. Змінну можна перепризначити або примусово призначити значення іншого типу без проблем або попереджень. Це може призвести до помилок, які часто не помічаються, особливо у великих програмах.

TypeScript, з іншого боку, використовує статичний тип. Змінним можна надати тип, коли вони оголошені. TypeScript перевірятиме типи під час компіляції та видаватиме помилку, якщо змінній буде надано значення іншого типу. Однак помилка не перешкоджає виконанню коду. Код усе одно буде скомпільовано у звичайний JavaScript і працюватиме нормально. Таким чином, TypeScript схожий на «перевірку правопису» для вашого коду. Це дасть знати,

коли щось виглядає не так, але це не змінить роботу вашого коду. Статичний тип є необов’язковим у TypeScript. Змінним можна присвоїти тип any, що дозволить мати будь-які значення. Якщо тип не вказано, тип буде встановлено any за замовчуванням.

Лише після появи ECMAScript2015, або ES6, JavaScript представив класи в мові. Однак важливо зазначити, що ці класи не змінюють успадкування мови на основі прототипу. TypeScript дозволяв використовувати класи до того, як вони були офіційно реалізовані в ES6. TypeScript дозволяє розробнику розширювати існуючі класи для створення нових за допомогою успадкування.

До переваг TypeScript варто віднести наступне:

– Необов’язковий статичний тип (ключ тут необов’язковий);

– Типовий висновок, який дає деякі переваги типів, фактично не використовуючи їх;

– Доступ до функцій ES6 і ES7 до того, як вони стануть підтримуватися основними браузерами;

– Можливість компілювати до версії JavaScript, яка працює в усіх браузерах;

– Чудова підтримка інструментів за допомогою IntelliSense.

Angular

Angular — це платформа розробки, на якій побудовано TypeScript. Як платформа Angular включає:

– Компонентна основа для створення масштабованих веб-додатківж

– Набір добре інтегрованих бібліотек, які охоплюють широкий спектр функцій, включаючи маршрутизацію, керування формами, зв’язок клієнт-сервер тощо;

– Набір інструментів розробника, які допоможуть вам розробляти, будувати, тестувати й оновлювати код;

З Angular можна використовувати переваги платформи, яка може масштабуватися від проєктів одного розробника до програм корпоративного

рівня. Angular розроблено, щоб зробити оновлення максимально простим, тому використовуйте переваги останніх розробок, докладаючи мінімум зусиль. Найкраще те, що екосистема Angular складається з різноманітної групи з понад 1,7 мільйона розробників, авторів бібліотек і творців контенту.

JavaScript є найбільш часто використовуваною мовою програмування веб-додатків на стороні клієнта. Його вбудовано в документи HTML, щоб уможливити взаємодію з онлайн-сторінками різними способами. Це проста, легка для вивчення мова, яка добре підходить для створення сучасних програм завдяки її широкій підтримці. Тепер є безліч фреймворків і бібліотек, які допомагають досягати цілей. Angular, як приклад, є фреймворком, який вирішує багато, якщо не всі проблеми, з якими стикаються розробники під час використання JavaScript самостійно. Розробники Angular можуть створювати веб-програми, SPA та PWA.

Bootstrap

Bootstrap робить адаптивний веб-дизайн реальністю. Це дає змогу веб-сторінці чи додатку визначати розмір і орієнтацію екрана відвідувача та автоматично адаптувати відображення відповідно до цього. Підхід, орієнтований на мобільні пристрої, передбачає , що смартфони , планшети та мобільні програми для конкретних завдань є основними інструментами співробітників для виконання роботи.

Bootstrap відповідає вимогам цих технологій у дизайні та включає компоненти інтерфейсу користувача, макети, інструменти JavaScript і структуру реалізації. Програмне забезпечення доступне попередньо скомпільоване або у вигляді вихідного коду . Марк Отто та Джейкоб Торнтон розробили Bootstrap у Twitter, щоб покращити узгодженість інструментів, що використовуються на сайті, і скоротити обслуговування. Програмне забезпечення раніше було відомо як Twitter Blueprint, а іноді його ще називають Twitter Bootstrap.

C# та .NET

У розробці програмного забезпечення C# і .NET є двома найбільш використовуваними технологіями. Обидва мають різні функції та призначення. В IT-індустрії людина повинна мати навички та знання про мови та фреймворки програмування. І під час вивчення та пошуку в Google цих предметів найчастіше загальними результатами є C# та .NET. Розробники .NET зазвичай використовують C# як мову програмування.

У загальному вигляді C# — це мова програмування, створена Microsoft для Загальної мовної інфраструктури (CLI). C# є як компонентною, так і об'єктно-орієнтованою мовою. Отже, це ідеальна мова програмування для використання та створення програмних компонентів. Крім того, C# походить від мов програмування C, що полегшує вивчення C, C++, Java та JavaScript програмістів. C# також має функції, які допомагають у розробці довговічних програм і програмного забезпечення, наприклад:

- Простий у вивченні та використанні
- Підтримує просте впровадження багатопоточної архітектури
- Забезпечує підтримку умовної компіляції
- Має стандартні бібліотеки Індексатори для організації даних
- Має положення та підтримку для автоматичного збору сміття Інтегрована Windows
- Його обробка винятків забезпечує структурований підхід до виявлення та відновлення помилок

Як було зазначено раніше, C# є мовою програмування. З іншої сторони, .NET — це платформа для розробників з відкритим кодом. Microsoft створила .NET, щоб забезпечити платформу, на якій можлива розробка різних програм. Крім того, платформа дозволяє використовувати кілька бібліотек, мов і редакторів. Крім того, це дозволяє розробляти програми або програмне забезпечення для робочого столу, Інтернету, ігор та IoT (Internet of Things).

2.2 Визначення середовища розробки

Для того, щоб створити подібний проєкт з різними налаштуваннями, великим об’ємом коду, достатньої кількості директорій, зображень та додаткових файлів можна використовувати різні середовища для розробки інтернет-додатків, але найзручнішими у цій сфері будуть такі середовища як WebStorm, Brackets та Visual Studio Code (рис. 2.1 – 2.3)

IDE, або інтегроване середовище розробки, дозволяє програмістам консолідувати різні аспекти написання комп’ютерної програми. IDE підвищують продуктивність програміста, об’єднуючи звичайні дії з написання програмного забезпечення в одній програмі: редагування вихідного коду, створення виконуваних файлів і налагодження. Написання коду є важливою частиною програмування.

Починаючи з порожнього файлу, пишемо кілька рядків коду, і народжується програма. IDE полегшують цей процес за допомогою таких функцій, як підсвічування синтаксису та автозаповнення. Жоден програміст не уникає написання помилок і програм з помилками.

Коли програма не працює належним чином, такі середовища надають інструменти для налагодження, які дозволяють програмістам досліджувати різні змінні та перевіряти свій код навмисно. IDE також надають підказки під час кодування, щоб запобігти помилкам перед компіляцією.

Більшість редакторів – це велика великий коєфіцієнт роботи, швидко запускаються та спритно працюють, також, сумісні з багатьма сучасними мовами програмування. IDE – у нашому випадку найкраще буде підходити, так як швидкість роботи та пам’ять комп’ютера як раз підходить під IDE-вимогу.

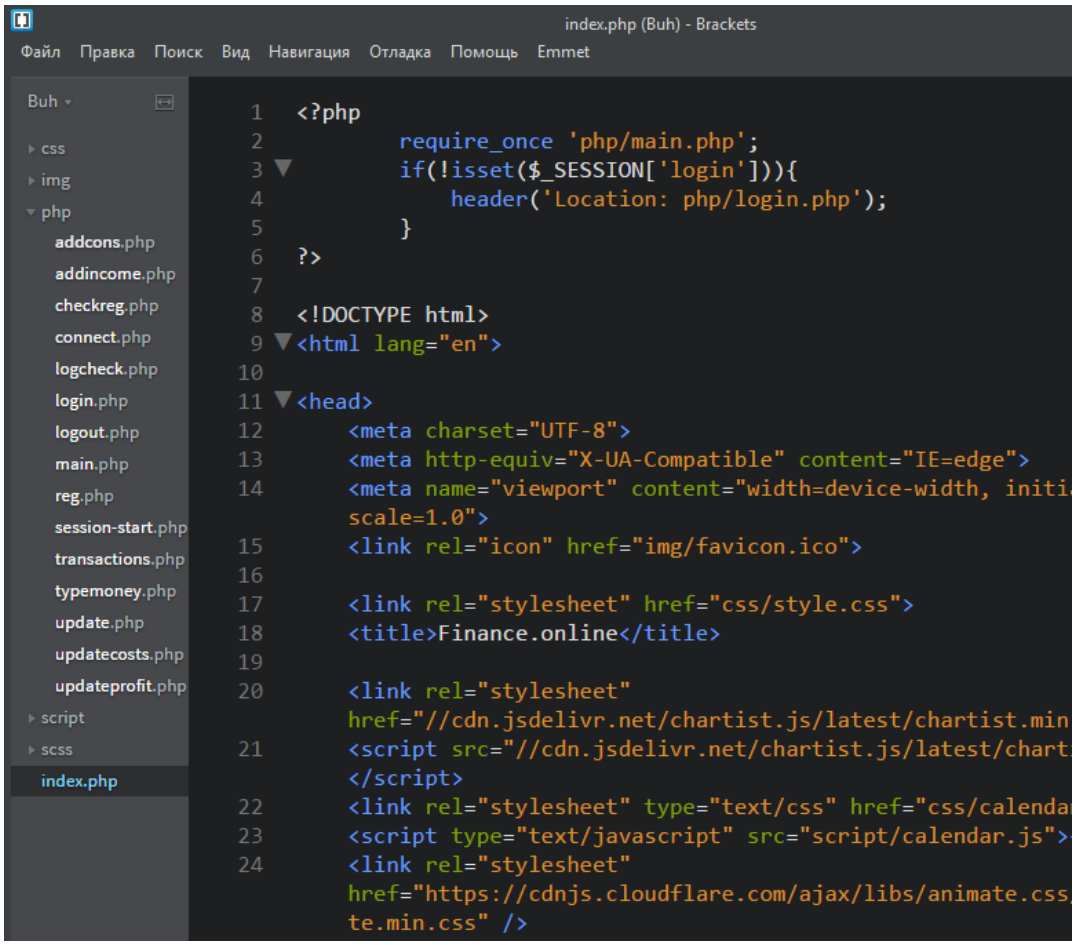

Рисунок 2.1 – Редактор коду Brackets

У кожного середовища розробки є свої плюси та мінуси, наприклад WebStrom має зручний інтерфейс та авторозпізнання коду типу JavaScript, Brackets має основні технології для реалізації роботи з HTML, CSS та JavaScript, але найзручнішим у цій справі буде саме VS Code, так як наділений гнучкими та інноваційними можливостями, наприклад:

- відкриття свого серверу без зайвих рухів;
- неймовірна база плагінів, котрі полегшують роботу у рази;
- низька ресурсозатратність та швидкодія;
- встановлення модулів за допомогою Node.js;
- вбудувана консоль, що має усі права як і Windows cmd;
- відладчик з роботою над Git.

Отже, найвдалішим вибором серед середовищ для розробки є Visual Studio Code, який, до речі, розроблений Microsoft для Windows та Linux.

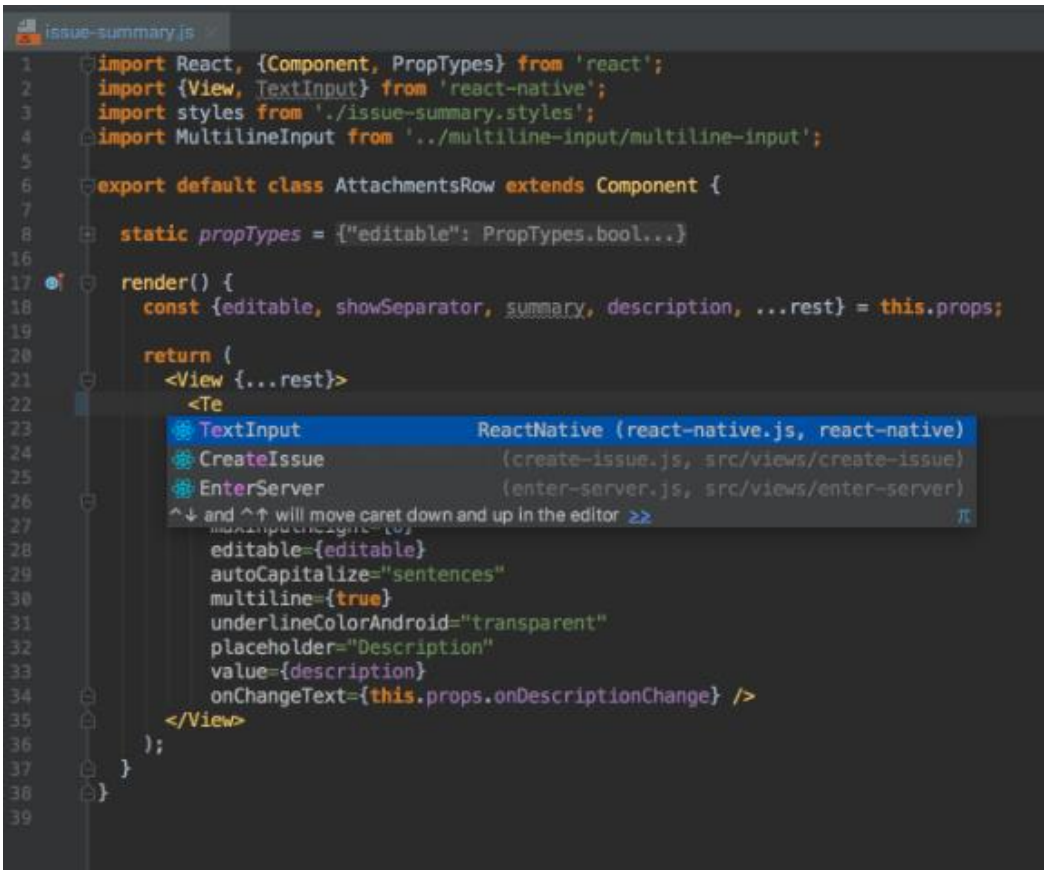

Рисунок 2.2 – Редактор коду WebStorm

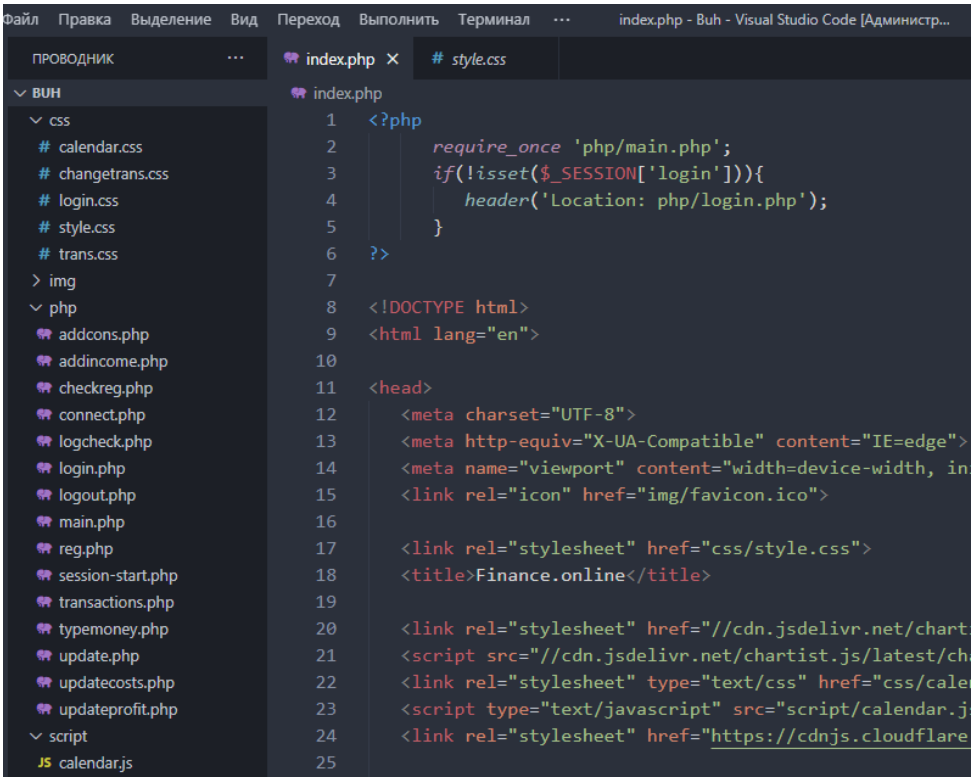

Рисунок 2.3 – Редактор коду VS Code

3 ІНФОРМАЦІЙНЕ ТА ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМИ

3.1 Інформаційна модель

Згідно з вимогами до веб-ресурсу можна побудувати наступну інформаційну модель (рис. 3.1 – 3.3).

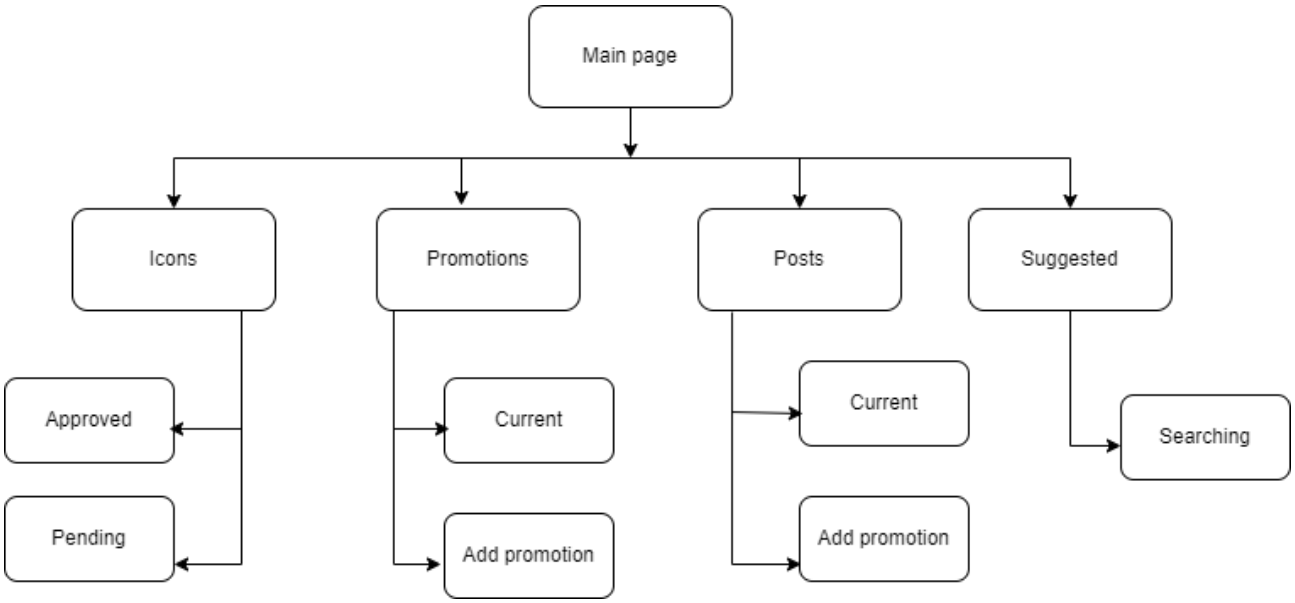

Рисунок 3.1 – Інформаційна модель “Icon admin console”

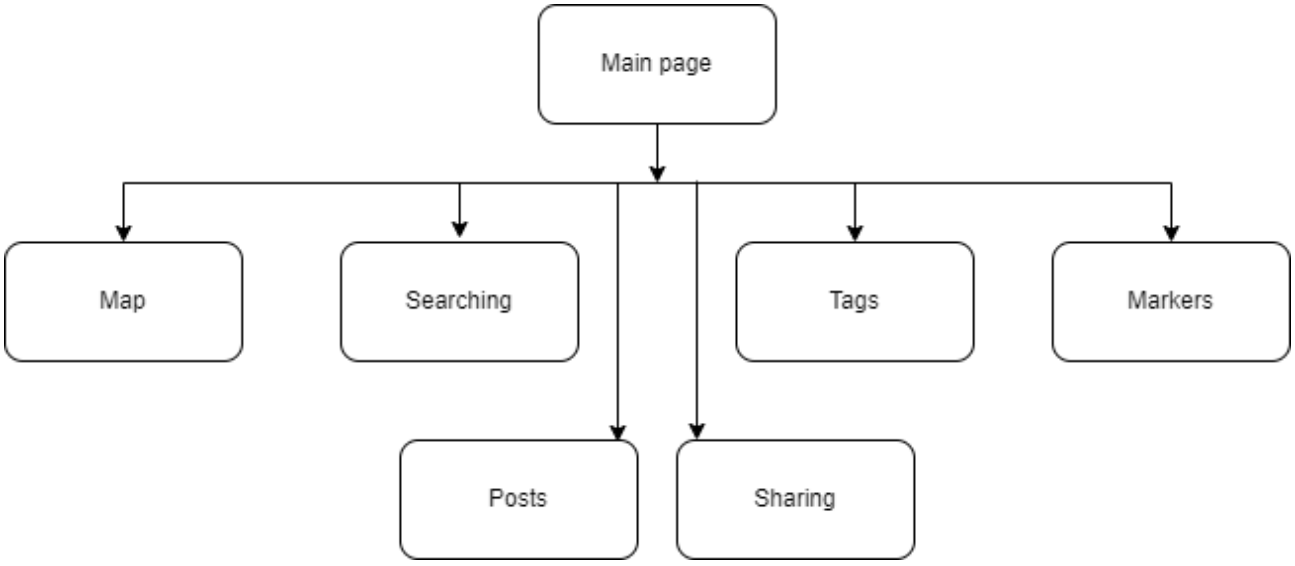

Рисунок 3.2 – Інформаційна модель “Public web client”

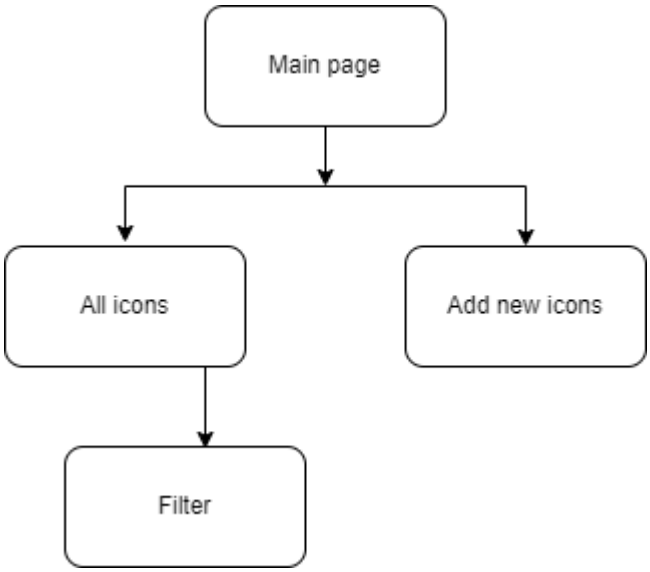

Рисунок 3.3 – Інформаційна модель “Icon upload page”

Веб-ресурси розрахований на два типи користувачів, а саме для адміністратора (можливість керування БД) та для звичайного користувача (без прав адміністрування).

3.2 Програмна реалізація

В першу чергу був створений проєкт на Angular за допомогою таких команд:

```
npm install -g @angular/cli
ng new icon-admin-console
```

В свою чергу вони запровадили Angular CLI та створили так званий initial project.

Спочатку були створені додаткові директорії та файли, які допоможуть більш структурно підійти до розробки сайту (рис. 3.4).

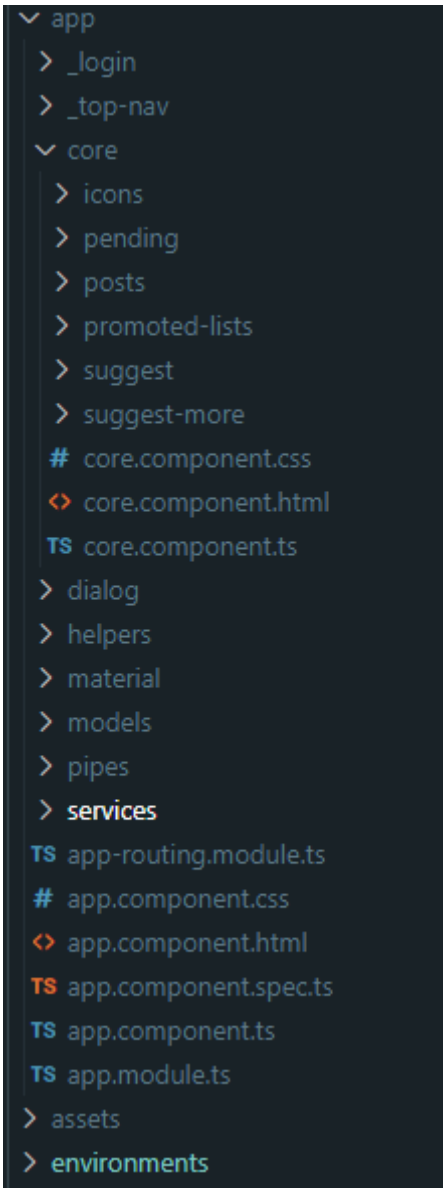

Рисунок 3.4 — Додаткові директорії

Також на головній сторінці сайту, а також і на інших вкладках, повинні бути присутні асинхронні запроси. Тому було прийняте рішення, щоб додати до директорій сервіси де буде знаходитися дані та основні запити до сервера (рис. 3.5).

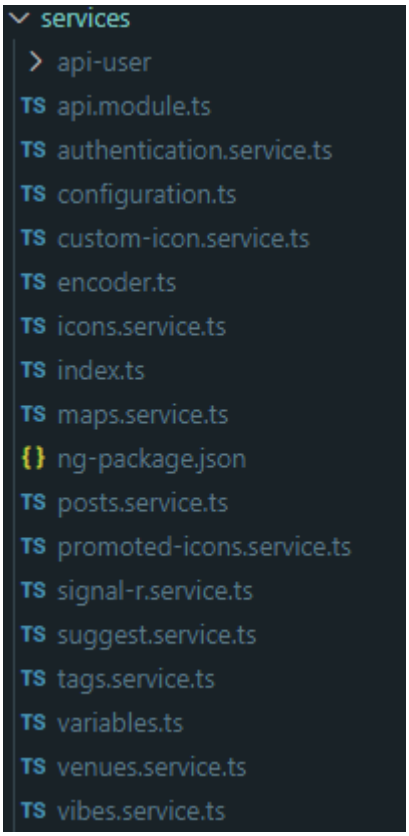

Рисунок 3.5 — Додаткові сервіси

Також були додані зв’язки з компонентами та модулями до основного модулю:

```
@NgModule({
  imports: [
    ApiModule,
    BrowserModule,
    HttpClientModule,
    GoogleMapsModule,
    HttpClientJsonpModule,
    NgbModule,
    BrowserModule,
    ReactiveFormsModule,
    AppRoutingModule,
    BrowserAnimationsModule,
    FormsModule,
    MaterialModule,
    A11yModule,
    PickerModule,
    AgmCoreModule.forRoot({
      apiKey: '################',
      libraries: ['places'],
      language: 'en',
      region: 'EN',
    }),
    MatGoogleMapsAutocompleteModule,
  ],
  declarations: [
```

```
    AppComponent,
    TopNavComponent,
    LoginComponent,
    CoreComponent,
    IconDetailsComponent,
    PromotedComponent,
    SpotLocationComponent,
    AddIconsComponent,
    SuggestComponent,
    AutocompleteComponent,
    PromotionDetailsComponent,
    PromotedListsComponent,
    AddPromotedComponent,
    EditPromoteComponent,
    DateAgoPipe,
    AllIconsComponent,
    VerificationComponent,
    PendingComponent,
    IconsComponent,
    PostsComponent,
    SuggestMoreComponent,
    CreatePostComponent,
    NumberThPipe,
    DateSoonPipe,
    SelectPostsComponent
  ]})
```

Для того, щоб робити запити на сервер потрібні url-api, тому було прийняте рішення створити окремі зміні та додати їх до проєкту:

```
providers: [
    { provide: BASE_PATH_IDENTITY, useValue: environment.identityUrl },
    { provide: BASE_PATH_API, useValue: environment.apiUrl },
    { provide: BASE_PATH_ICONS, useValue: environment.customIconsUrl },
    { provide: BASE_PATH_IMAGE_HOST, useValue: environment.imageHostUrl },
    { provide: BASE_PATH_GOOGLE_MAP, useValue: environment.googleMapUrl },
  ],
```

Перед відправкою запиту потрібно додавати до нього поле Authentication разом з токеном. З цим завдання може впоратися interceptor, який також був об’явлений у якості провайдера.

```
  { provide: HTTP_INTERCEPTORS, useClass: ErrorInterceptor, multi: true }
```

Для того, щоб отримати іконки на головній сторінці потрібно створити запит на їх отримання:

```
return this.http.request<IconFilterResultDtoIListResponse>(
      'get',
      `${this.basePath}/icons/filter`,
      {
        params: queryParameters,
        withCredentials: this.configuration.withCredentials,
        headers,
        observe,
        reportProgress,
      }
```

```
    );
```

Цей запит вертається у вигляді Observable з дженеріком any на який потрібно підписатися у головному компоненті core.component.ts:

```
    this.iconService.iconsFilterGet( null, this.orderType, null, null, null,
this.skipIcons, this.takeIcons ).pipe(take(1)).subscribe((icons) => {
    if (icons.success && this.selectedChipMenu.all) {
    if (this.selectedChipMenu.all) this.icons = icons.value;
    this.isLoading = false; } });
```

Щоб скористатися цим сервісом потрібно його інжектувати у конструкторі того компонента, де він буде виконуватися:

```
    constructor(private iconService: CustomIconService,) {}
```

Також для відображення цих даних потрібно написати розмітку властиву фреймворку та за допомогою структурної директиви ngFor виводимо дані у html документ і можемо побачити код та його виповнений результат на рисунку 3.6:

```
    <div *ngFor="let icon of icons; let i = index" [ngClass]="{'loading_end':
isLoading}">
    <a href="javascript:void(0);" (click)="openIconDetailsDialog(icon)"
class="w-100">
    <div class="iconAll">
    <div *ngIf="icon && icon.iconType ==='Emoji'" class="iconEmoji">
<div>{{getSymbols(icon.code)}}</div>
    </div> <div *ngIf="icon && icon.iconType === 'Picture'">
    <img class="icon mt-0" src="{{getCustomIcons(icon.id, i)}}" alt=""> </div>
</div> </a> </div>
```

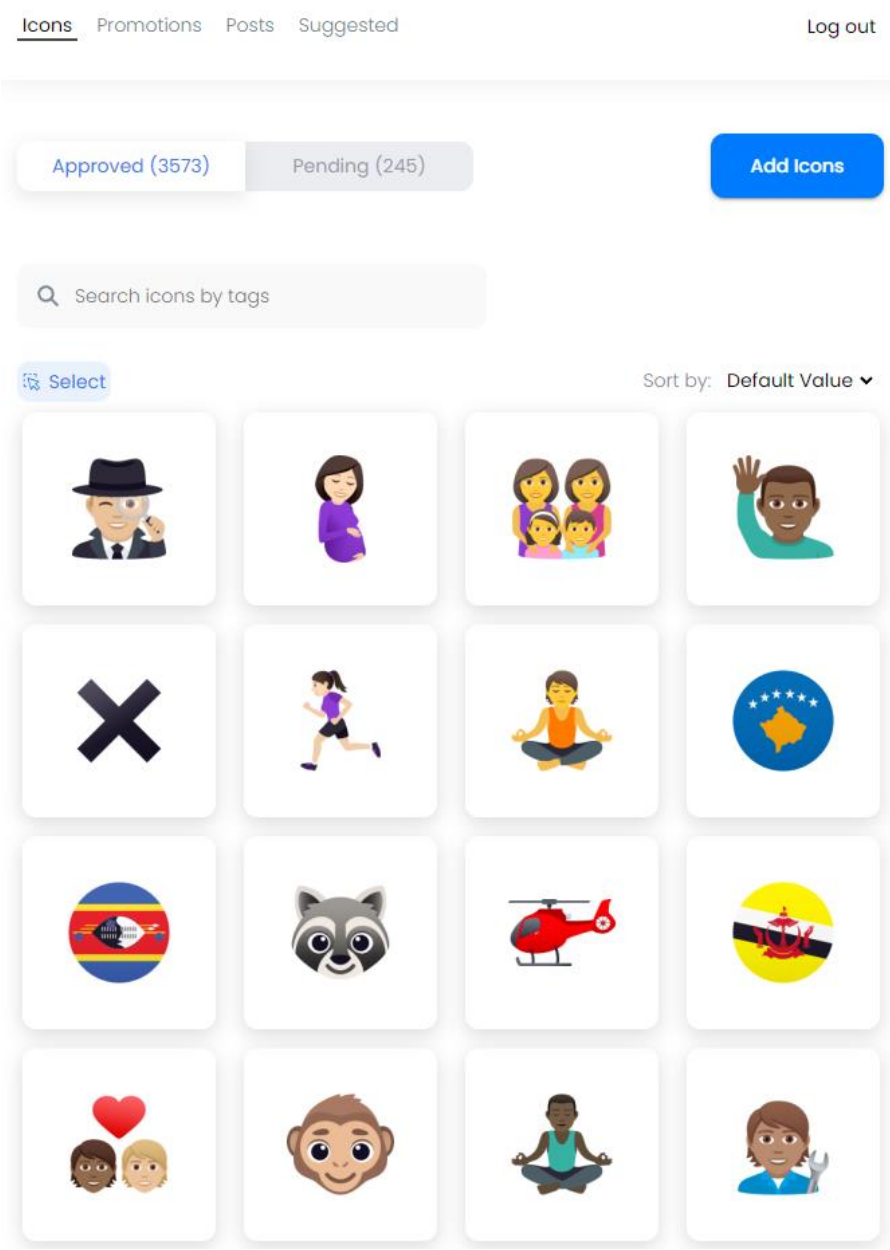

Рисунок 3.6 — Головна сторінка

При авторизації користувач повинен занести свої дані у user name та password (рис. 3.7).

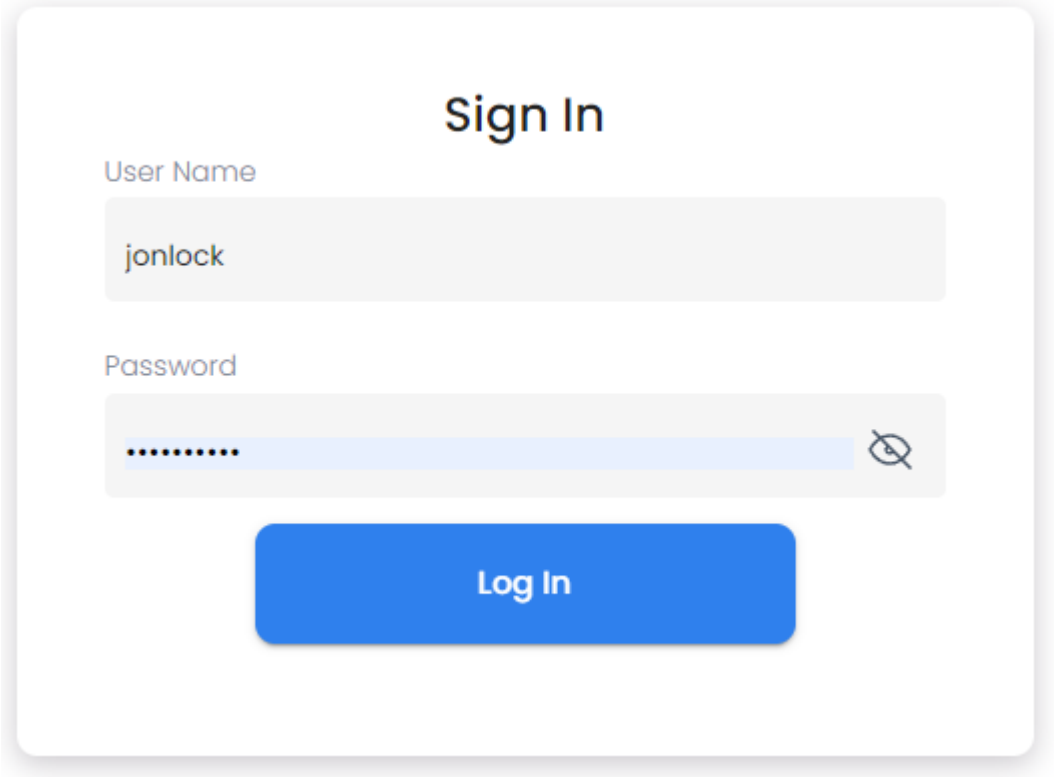

Рисунок 3.7 — Авторизація

У даному коді можна побачити як проходить валідація юзера за допомогою реактивних форм та запита на сервер, який був відправлений через сервіс.

```
      signIn(): void {
      if (!this.authForm.get('password').hasError('required')) {
this.authForm.get('password').setErrors(null); }
      if (this.authForm.invalid){ return; }
      this.authLoading = true;
      this.authenticationService.login(this.authForm.value) .pipe(first())
      .subscribe( data => { this.router.navigate([this.returnUrl]); }, error =>
{
      if (error) { this.authForm.get('username').setErrors({ error: 'Wrong login
or password' });
      this.authForm.get('password').setErrors({ error: 'Wrong login or password'
}); } this.error = error; this.authLoading = false; }); }
```

На інших сторінках окрім першої, тобто icons, потрібно використовувати токін, який був записаний у локальне сховище після підтвердження користувача та його паролю. Авторизаційний пароль добавляється за допомогою jwtinterceptor:

```
      intercept( req: HttpRequest<any>, next: HttpHandler ):
Observable<HttpEvent<any>> {
      if (localStorage.getItem('token') == null) return next.handle(req);
      let authReq = req;
      let token =
this.authenticationService.isAuthenticatedUserValue.access_token; // if
(!authToken) return next.handle(req);
```

```
      if (token != null) { authReq = this.addTokenHeader(req); }

      private addTokenHeader(request: HttpRequest<any>) {
      const authToken =
this.authenticationService.isAuthenticatedUserValue.access_token;
      if (authToken) { return (request = request.clone({ setHeaders: {
Authorization: `Bearer ${authToken}`, }, })); }
      else { return request; } }
```

Для public web client були використані подібні функції та зв’язки, але без прив’язки до токену (рис. 3.8).

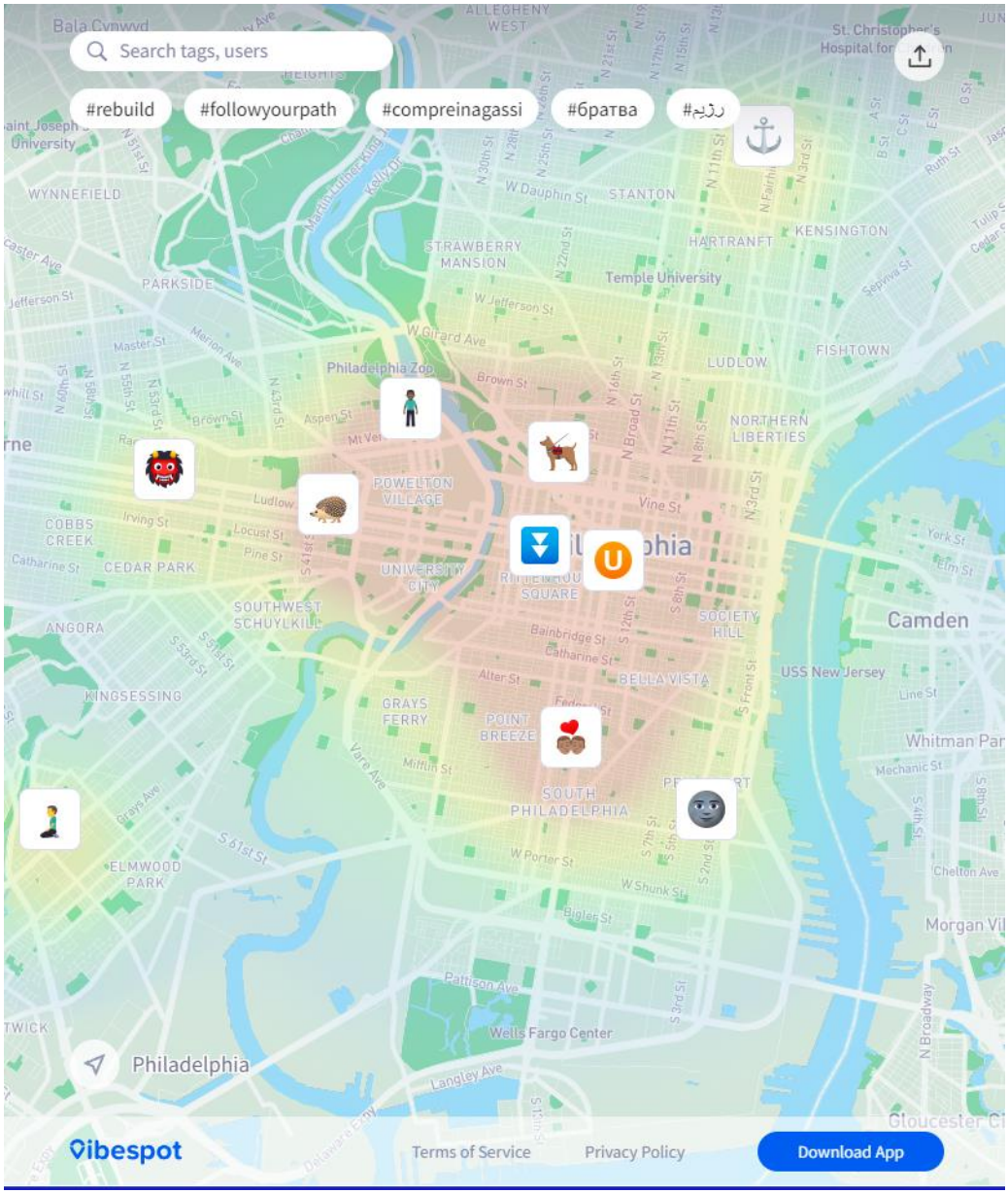

Рисунок 3.8 – «Public web client» сторінка

Для відображення мапи було використано технологію Mapbox – це платформа даних про місцезнаходження, яка підтримує карти та служби

визначення місцезнаходження, які використовуються в багатьох популярних програмах. Визначаємо карту, маркери та івенти в html компоненті:

```
      <mgl-map [style]="'mapbox://styles/vibespot/{{link}}?optimize=true'"
[zoom]="[12]" [center]="center" (mapLoad)="map = $event; getVibes()"
(mapDragEnd)="getVibes()" (mapDragStart)="!isActiveSearchTags ? null :
openTagsSearch(); isActiveList = false;" (zoomEnd)="getVibes()"
(mapLoad)="onMapLoad()" (click)="tapOnMap($event)">
      <mgl-marker *ngFor="let marker of markers; let i = index" class="mgl-mrk"
[lngLat]="[marker.lng, marker.lat]" anchor="bottom">
      </mgl-marker>
      </mgl-map>
```

При натисканні на адресу потрібно відображати пости, пов'язані з цим місцем та іконою, тобто при кліку потрібно передати у функцію longitude, latitude та venueId (рис. 3.9 – 3.11).

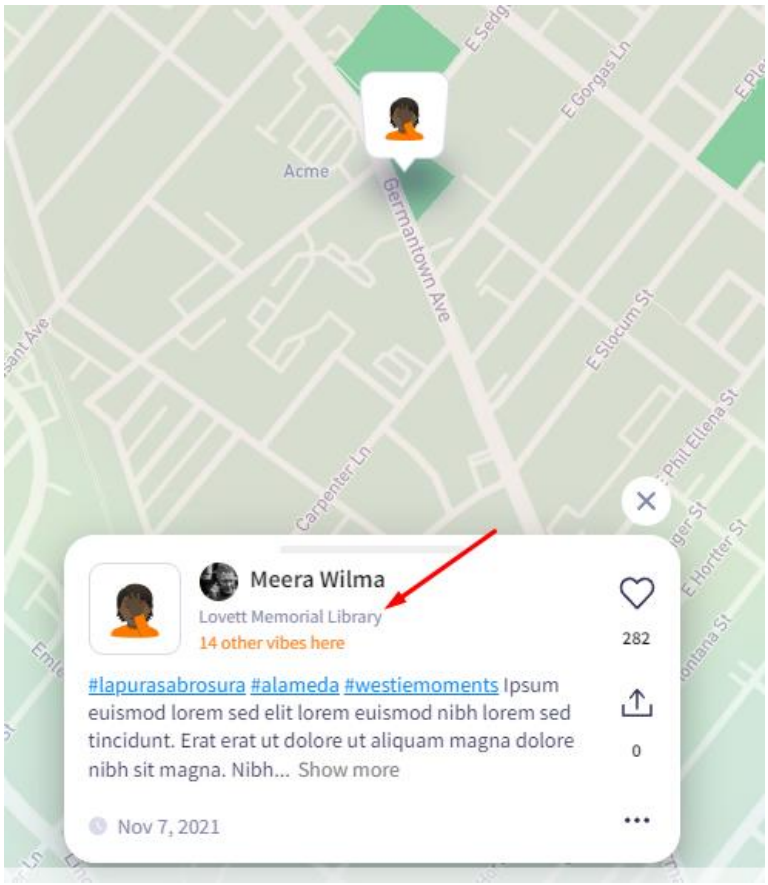

Рисунок 3.9 – Відображення поста

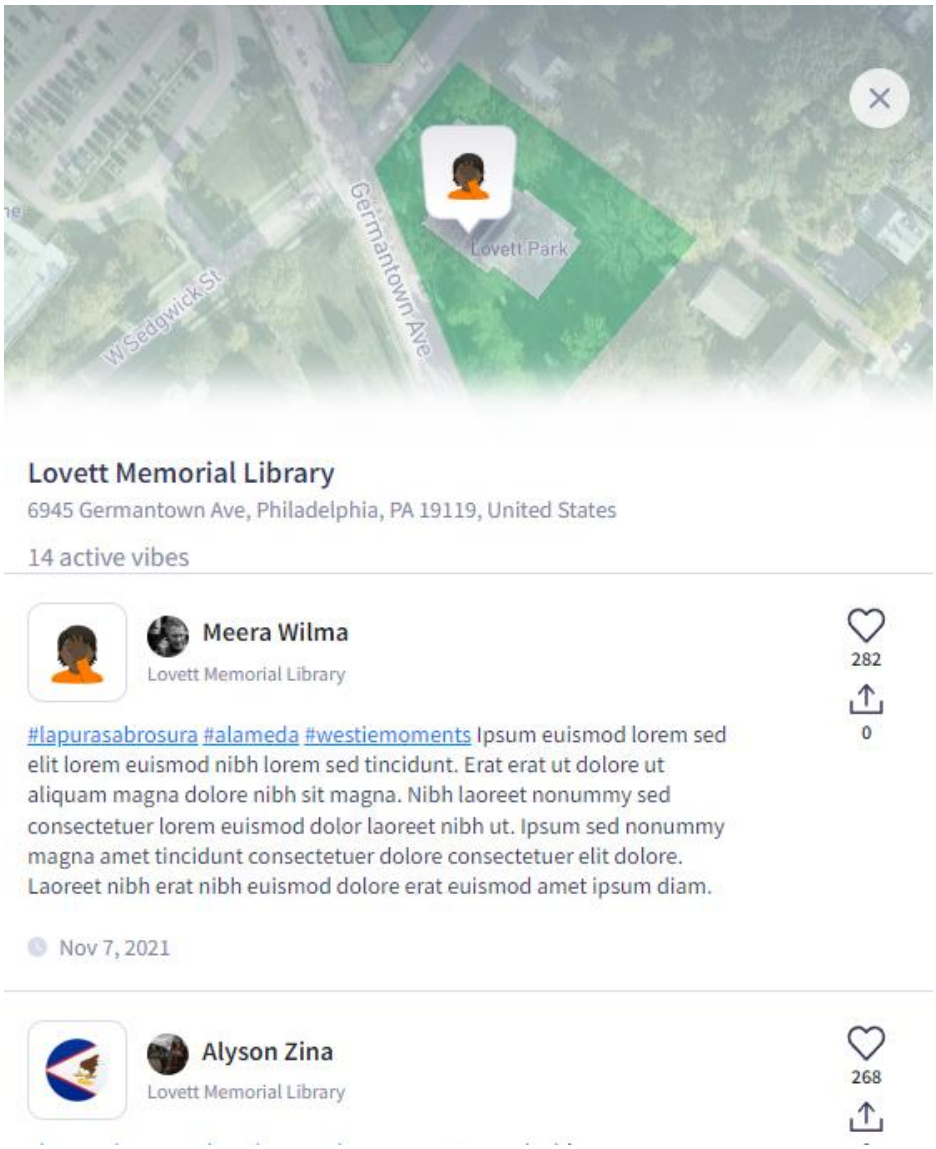

Рисунок 3.10 – Відображення постів

Щоб додати зум при виборі тегу та сортування постів, орієнтуючись на радіус та на поточне положення карти, потрібно додати цей код:

```
      getEventsWithATag(vibes: any) {
      // Geographic coordinates of the LineString
      let coordinates;
      coordinates = vibes.map((v) => [ v.venue.geo.longitude,
v.venue.geo.latitude, ]); //
      // Create a 'LngLatBounds' with both corners at the first coordinate.
      const bounds = new mapboxgl.LngLatBounds(coordinates[0], coordinates[0]);
// Extend the 'LngLatBounds' to include every coordinate in the bounds result.
      for (const coord of coordinates) { bounds.extend(coord); }
      this.map.fitBounds(bounds, { padding: { top: 100, left: 100, bottom: 100,
right: 100 }, });
      let success = false; interval(200) .pipe(takeWhile(() => !success))
.subscribe(() => { if (!this.mapsService.isLoadingVibes) { success = true;
this.getVibes(); } }); }
```

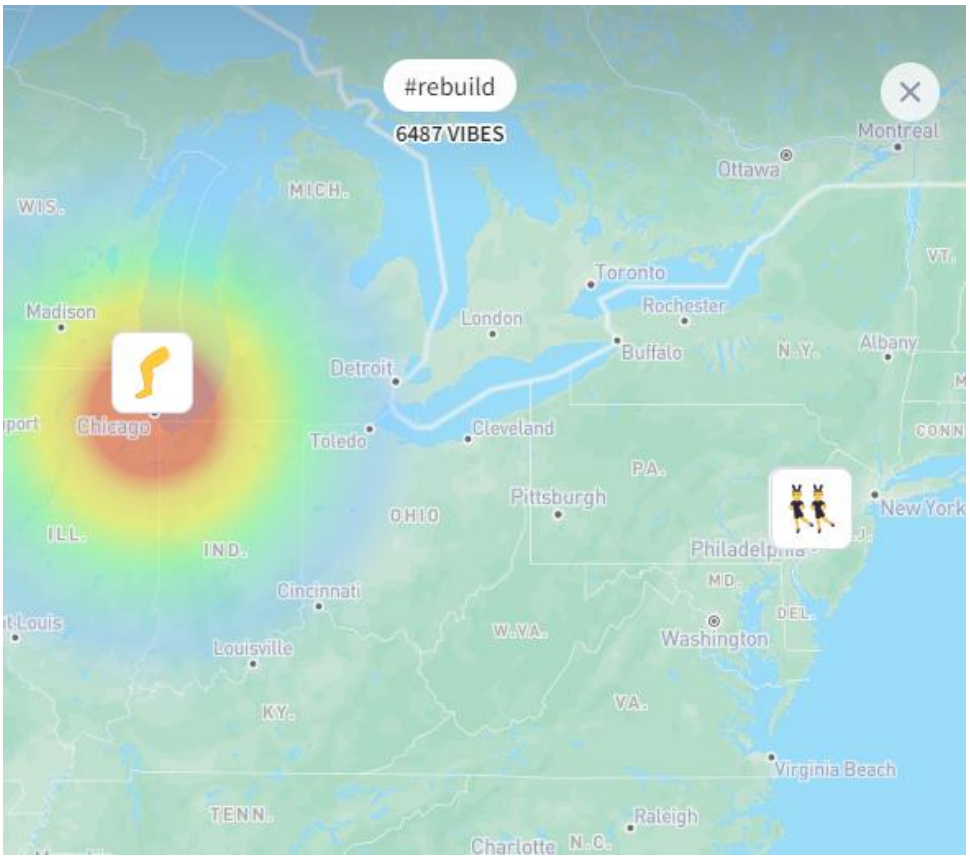

Рисунок 3.11 — Приклад роботи при обиранні тегу

У кожному з сайті було додано адаптив для будь-якого екрану, тобто це від 2560 до 376px (рис. 3.12 – 3.14). Було прийняте рішення розроблювати не під конкретні розміри, а під всі. Реалізовано на препроцесорі SCSS з змінними та медіа запитами:

```
$phoneSizeXr: 470px;
@media (max-width: $phoneSizeXr) {}
```

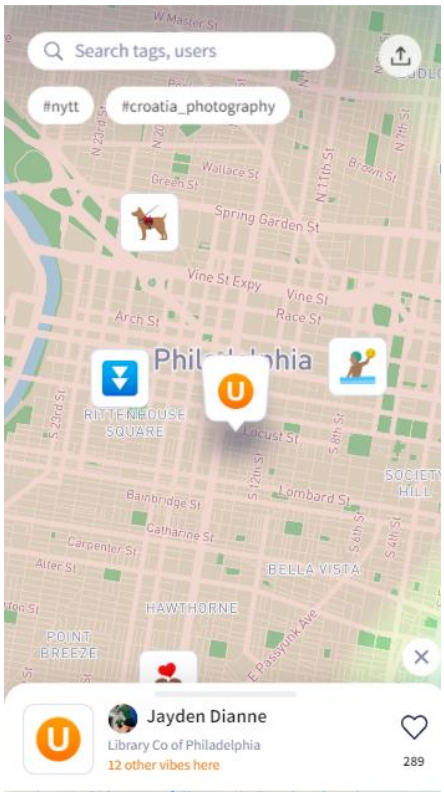

Рисунок 3.12 – Приклад мобільного зразка

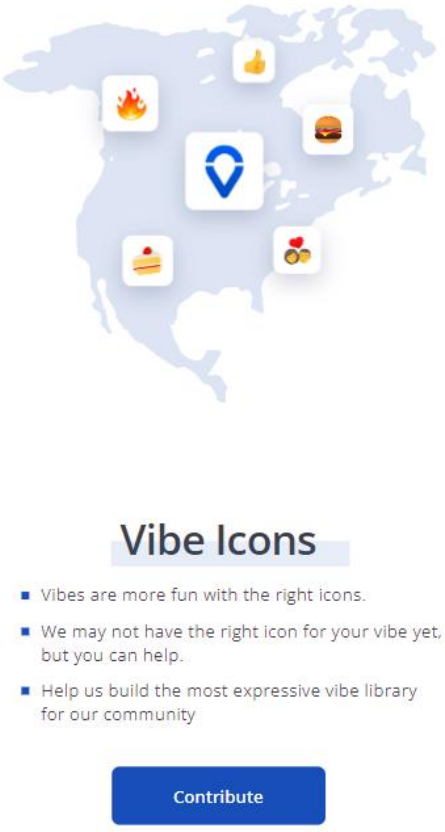

Рисунок 3.13 – Приклад мобільного зразка “Landing page”

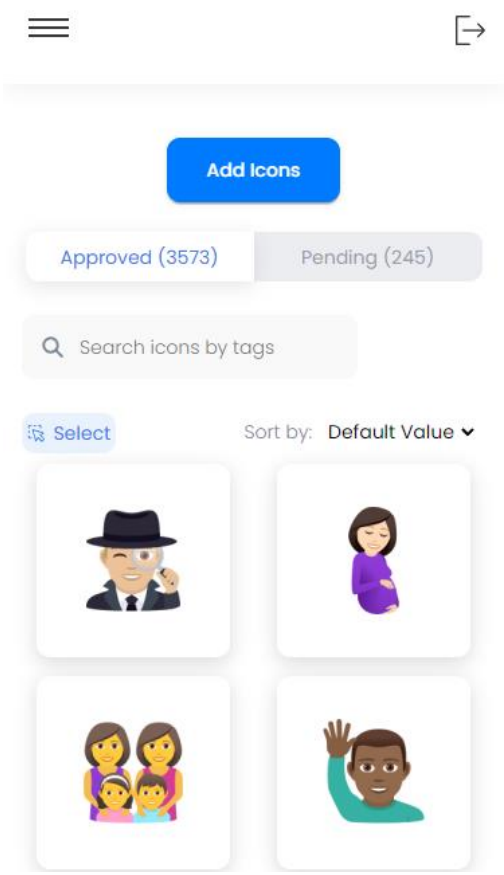

Рисунок 3.14 – Приклад мобільного зразка “Admin console”

У шапці було додано бургер меню з анімацією, щоб на маленьких екранах було лаконічно і зручно продивлятися матеріал та користуватися ним (рис 3.15).

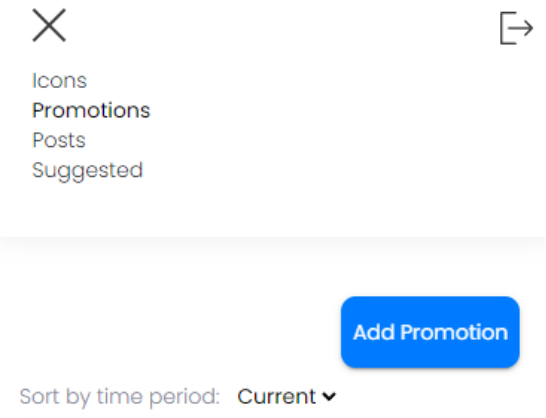

Рисунок 3.15 – Відкрите бургер меню

Після лоадеру на лендінгу можна побачити основний контент за задній фон, який має паралакс та текст з ефектом скрамблінгу й зображено на рисунку 3.16:

```
parallaxFunc() {
const parallax = document.querySelector('header');
if (parallax) {
const forParallax = 3;
const speed = 0.15;
let positionX = 0, positionY = 0;
let coordXprocent = 0, coordYprocent = 0;
const setMouseParallax = () => {
const distX = coordXprocent - positionX;
const distY = coordYprocent - positionY;
positionX = positionX + distX * speed;
positionY = positionY + distY * speed;
parallax.style.cssText = `background-position: ${ positionX / forParallax
}px ${positionY / forParallax}px`;
requestAnimationFrame(setMouseParallax); };
setMouseParallax();
parallax.addEventListener('mousemove', (e) => { const parallaxWidth =
parallax.offsetWidth;
const parallaxHeight = parallax.offsetHeight;
const coordX = e.pageX - parallaxWidth / 2;
const coordY = e.pageY - parallaxHeight / 2;
coordXprocent = (coordX / parallaxWidth) * 100; coordYprocent = (coordY /
parallaxHeight) * 100; }); } }
```

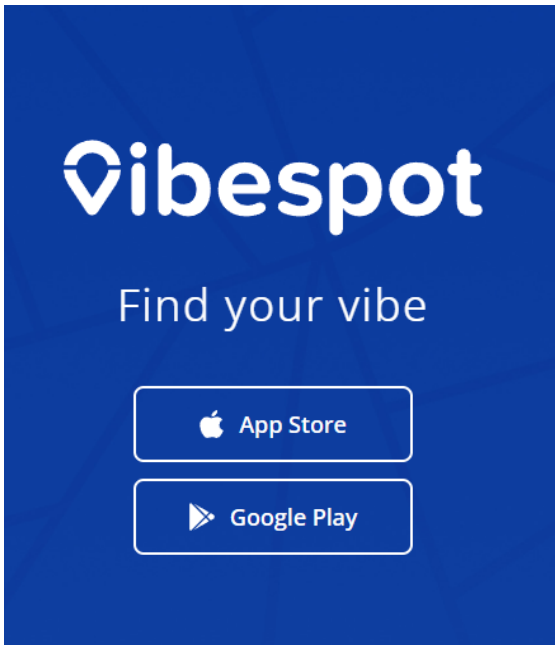

Рисунок 3.16 – Паралакс ефект при ведені мишкою

Код для друкування літер (рис. 3.17):

```
changeLetters(): void {
      const size = this.title.length;
      this.titleTags = 'F';
```

```
        let index = 0;
        let intervalOfChangingLetters = setInterval(() => {
        this.titleTags = this.title.substr(0, index + 1) + '|';
        if (index != this.title.length - 1) index++; else {
        this.titleTags=this.title.substr(0, index + 1);
        clearInterval(intervalOfChangingLetters); } }, 150);
}
```

Рисунок 3.17 – Паралакс ефект при ведені мишкою

У даному коді, який наведено нижче, описано функцію для підключення к БД, взяття користувача по currentUserI та зчитування даних з БД.

```
namespace Vibespot.Application.Services {
public class ContentQueryService : IContentQueryService {
private readonly IUserContentStore _db;
private readonly IUserContentQueryProvider _userContentQueryProvider;
public ContentQueryService(IUserContentStore db, IUserContentQueryProvider
userContentQueryProvider) {
_db = db;
_userContentQueryProvider = userContentQueryProvider; }
public async Task<Response<UserRef>> GetUser(Guid itemId, Guid?
currentUserId) { var item = await _db.GetUser(itemId);
if (item == null || item.State == UserState.Deleted || item.State ==
UserState.Inactive && itemId != currentUserId)
return "User was not found";
var reactions = currentUserId.HasValue ? await
_db.GetReactionsByUser(itemId, currentUserId.Value) : new
List<Reaction>();
item.MyLike = reactions.FirstOrDefault(i => i.EntityId == item.Id &&
i.Action == UserAction.Like);
item.MyComplaint = reactions.FirstOrDefault(i => i.EntityId == item.Id &&
i.Action == UserAction.Complain);
return item; }}}
```

Навантаження на сервер потрібно було зменшити, бо об’єм даних дуже великий. Тільки щоб отримати усі дані про юзерів, постів та інше потребується велика кількість ресурсів і щоб цього уникнути було використано кеш. В даному випадку кеш будувався як ключ, тобто cacheKey. На приклад, можна взяти запит

на отримання запропонованих іконок. Через інтерполяцію створюємо ключ зі заздалегідь обумовленими полями та додаємо до запиту параметри (рис. 3.18).

```
        [HttpGet("promoted")]
          [ResponseCache(CacheProfileName = "suggest/promoted")]
          [Produces("application/json")]
          [ProducesResponseType(typeof(Response<IList<IconStatResult>>), 200
)]
          public async Task<IActionResult> SuggestPromotedIcons([FromQuery]
SuggestPromoedIconsRequest model)
          {
        var cacheKey = $"{model.Location?.City}:{model.Location?.Country}:"
+$"{model.Location?.GeoLocation?.Latitude}:{model.Location?.GeoLocation?.L
ongitude}:" +$"{model.Location?.State}:{model.LocationComparisonLevel}:{mo
del.IncludeTagsData}:{model.Period}:{model.Skip}:{model.Take}";
        var result = await _distributedCache.GetOrCreateFileStream(cacheKey,
        async () => await _service.GetPromotedIcons(model.Location, model.Lo
cationComparisonLevel,
                    model.Period, model.IncludeTagsData, model.Skip,
                    model.Take));

            if (!result.Success)
                return BadRequest(result);

            return Ok(result);
        }
```

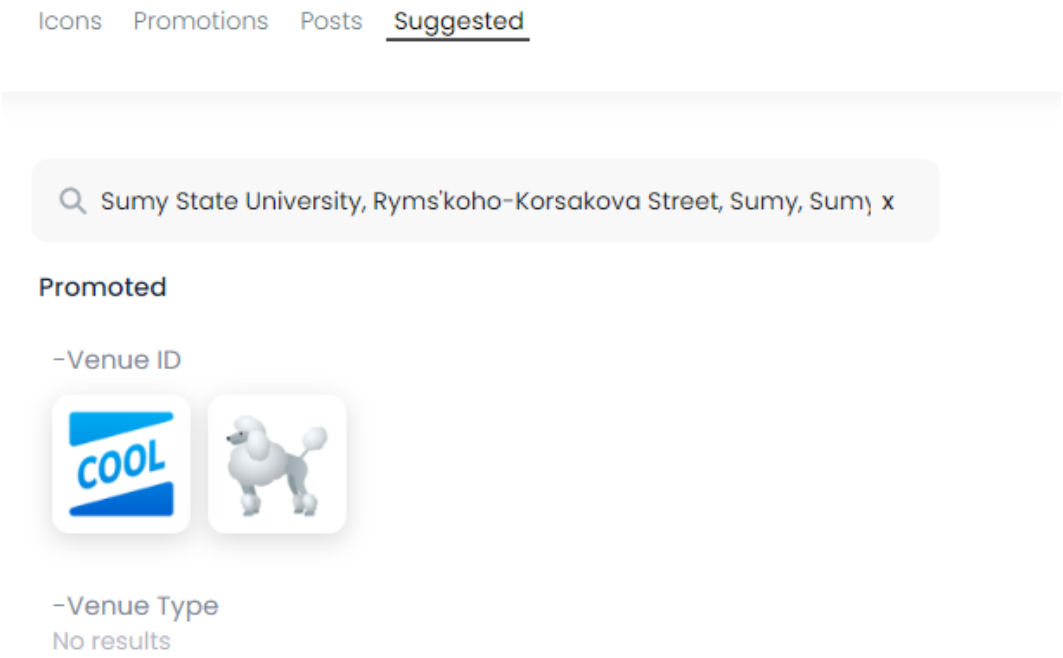

Рисунок 3.18 – Отримання результатів по suggest/promoted

ВИСНОВКИ

У ході виконання роботи було проведено аналіз предметної області застосування аналогічних сайтів та вирішувалась актуальність задачі зі створення даних веб-орієнтованих інформаційних систем для знаходження нових знайомств та місць. Проведено аналіз аналогів створюваного сайту, визначені наявні проблеми та сформульована мета. У роботі виконано моделювання ІС за допомогою структурно-функціональному аналізу, розроблено додатки.

Після реалізації проєкту проведено тестування, у результаті якого не виявлено суттєвих відхилень в роботі інформаційної системи.

У результаті дипломного проєкту було розроблено веб-ресурси, які дозволяють користувачам знаходити нові знайомства, отримувати інформацію про івенти та адмініструвати свої пости та іконки. Розроблений проєкт має зручний функціонал, що задовольняє всі вимоги, поставлені на етапі постановки завдання.

Інформаційні веб-ресурси вже розроблені та розміщені на хостингу. Це означає, що користувачі мають можливість доступатися до них через Інтернет і користуватися їхнім функціоналом. Розміщення проєкту на хостингу забезпечує стабільну роботу системи і дозволяє широкому колу користувачів використовувати його послуги.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Розробка сучасних веб-сайтів на Angular та консультування // Angular. URL: https://angular.io/.
2. Галицький Б. Фреймоврки та їх значення у проекті / Богдан Галицький. – Київ, 2019. – 180 с.
3. Денісон Д. Створення веб-сайтів за допомогою PHP, MySQL, JavaScript, CSS и HTML5 / Р. Ніксон. – [2-ге вид.]. – Київ, 2017. – 500 с.
4. Річне навчання веб програмування, створення ігор та розробки // It proger. URL: https://itproger.com/.
5. Кількісні ресурси для розробників від розробників. // Developer Mozilla. URL: https://developer.mozilla.org/.
6. Федорчук А. Посібник для прискореного навчання. Як створюються веб-сайти / Андрій Федорчук. – Польща, 2018. – 150 с.
7. Хантів К. Конверсія сайту. Перетворюємо відвідувачів на покупців / Кен Хантів., 2018. – 250 с.
8. Хоган Б. Книга веб-програміста. Секрети професійної розробки веб-сайтів / Б. Хоган, К. Уоррен, М. Уєбер. – Румунія, 2018. – 288 с.
9. Bourne M. The success and failure of performance measurement initiatives: Perceptions of participating managers / Bourne M., Neely A., Platts K., Mills, J // Int. J. Oper. Prod. Manag. 2002 – pp. 22, 1288–1310
10. Digital Analytics Association // San Francisco, 2018 – pp. 18-24
11. Conversion Rate Optimization Report // London: Econsultancy, 2011 – pp.18-22
12. Chandra B.S. / Chandra B.S., Ray, A.B // The Impact of Business Intelligence on Digital Marketing, International Journal of Engineering, IT and Scientific Research (IJEISR), 2017. Vol. 1, No. 1, pp.1-7. Doi:10.5281/zenodo.2643439
13. Smartinsights. D. Chaffey — Режим доступу: URL: https://www.smartinsights.com/digital-marketing-strategy/race-a-practical-framework-to-improve-your-digital-marketing/