МІНІСТЕРСТВО ОСВІТИ І НАЙКИ УКРАЇНИ

Харківський національний університет радіоелектроніки

Факультет _____Центр післядипломної освіти _____

Кафедра _____Програмної інженерії_____

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

рівень вищої освіти - другий (магістерський)

Дослідження методів та засобів розробки веб-застосувань для програмної реалізації веб-сервісу щодо проведення міських квестів

Виконав:

Студент _2_ курсу, групи _____ІПЗздм-19-1_____

_____Зубрєва Д.Ю._____

Спеціальність 121-Інженерія програмного забезпечення

Тип програми _____освітньо-наукова_____

Керівник ___Вечур__О.В._____

Допускається до захисту

Зав. Кафедри, проф. _____ З.В. Дудар

2021 р.

Харківський національний університет радіоелектроніки

Факультет Центр післядипломної освіти
(повна назва)

Кафедра Програмної інженерії
(повна назва)

Рівень вищої освіти другий (магістерський)

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

Тип програми Освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Інженерія програмного забезпечення
(повна назва)

ЗАТВЕРДЖУЮ:

Зав.кафедри _____
(підпис)

« ___ »_____ 2021р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студента *Зубрєвої Діани Юріївни*
(прізвище, ім'я, по батькові)

1. Тема роботи *Дослідження методів та засобів розробки веб-застосувань для програмної реалізації веб-сервісу щодо проведення міських квестів*
затверджена наказом університету від *26.03.2021* № *34 Стз*
2. Термін подання роботи до екзаменаційної комісії *14 травня* 2021р.
3. Вихідні дані до роботи *засоби створення клієнтських додатків, засоби для роботи з мапами , методи нормалізації критеріїв та пріоритетів, об'єктно-орієнтована мова програмування C#, ASP.NET Core, фреймворк Angular, React, Vue.js, середовище розробки Visual Studio Professional 2019*
4. Перелік питань, що потрібно опрацювати в роботі *мета роботи, аналіз проблемної галузі і постановка задачі, аналіз методів та засобів для дослідження, розробка додатків, оцінювання методів та засобів..*
5. Перелік графічного матеріалу із зазначенням креслеників, схем, слайдів ілюстрацій *мета, опис обраних технологій, визначення критеріїв, діаграма варіантів використання, діаграма розгортання, діаграма кооперацій ,реалізація, інтерфейс, аналіз та порівняння, оцінка критеріїв, висновки.*

6. Консультанти розділів роботи

Найменування Розділу	Консультант (посада, прізвище, ім’я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	Дата
Спецчастина	Вечур О.В.		

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1.	Аналіз предметної галузі	17 лютого 2021р.	виконано
2.	Аналіз методів та засобів дослідження	10 березня 2021р.	виконано
3.	Проектування додатку	17 березня 2021р.	виконано
4.	Розробка додатку	31 березня 2021р.	виконано
5.	Оцінка методів та засобів дослідження	05 травня 2021 р.	виконано
6.	Підготовка пояснювальної записки	06 травня 2021р.	виконано
7.	Підготовка презентації та доповіді	06 травня 2021р.	виконано
8.	Перевірка на академічний плагіат	08 травня 2021р.	виконано
9.	Нормоконтроль	11 травня 2021р.	виконано
10.	Рецензування	11 травня 2021р.	виконано
11.	Попередній захист	14 травня 2021р.	виконано
12.	Занесення диплома в електронний архів	14 травня 2021р.	виконано
13.	Допуск до захисту у зав. Кафедри	14 травня 2021р.	виконано

Дата видачі завдання 25 січня 2021р.

Студент _____
(підпис)

Керівник роботи _____ _____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ / ABSTRACT

Кваліфікаційна робота магістра містить: 59 с., 19 рис., 5 табл., 18 джер.

КЛІЄНТ-СЕРВЕРНА ВЗАЄМОДІЯ, МАПА, КЛІЄНТСЬКІ ФРЕЙМВОРКИ, КВЕСТИ.

Об’єктом дослідження є технології розробки веб-застосувань з використання мап на прикладі веб-сервісу щодо проведення міських квестів.

Метою роботи є розглянення існуючих технологій розробки клієнтських додатків, а також розробка та реалізація веб-сервісу щодо проведення міських квестів.

Методи розробки базуються на фреймворку Angular, бібліотеках React та Vue.js, фреймворку Asp.Net Web Api [1], бібліотеках Leaflet.js та OpenLayers, базі даних MsSql , середовищі розробки Visual Studio та Visual Studio Code.

Результатом роботи є розробка та реалізація веб-сервісу з використанням трьох різних клієнтських технологій, аналіз їх ефективності та їх порівняння.

CLIENT-SERVER INTERACTION, MAPS, CLIENT FRAMEWORKS, QUESTS.

The object of research is the technology of developing web applications using maps on the example of a web service for urban quests.

The purpose of the work is to consider the existing technologies for client applications development, as well as the development and implementation of a web service for conducting urban quests. Development methods are based on Angular, React, Vue.js, Asp.Net Web Api [1], Leaflet.js and OpenLayers libraries, MsSql databases, Visual Studio and Visual Studio Code development environments.

The result is the development and implementation of a web service using three different client technologies, analysis of their effectiveness and their comparison.

Я, Зубрєва Діана Юріївна, студент гр. ІПЗздм-19-1, здобувач вищої освіти на другому (магістерському) рівні кафедри «Програмна інженерія», заявляю: моя кваліфікаційна робота на тему «Дослідження методів та засобів розробки веб-застосувань для програмної реалізації веб-сервісу щодо проведення міських квестів», що буде представлена в екзаменаційну комісію для публічного захисту, виконана самостійно, в ній не містяться елементи плагіату і вона може бути опублікована в електронному архіві відкритого доступу ElAr KhNURE. Всі запозичення з друкованих та електронних джерел мають відповідні посилання.

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

ЗМІСТ

Вступ 8
1 Аналіз предметної галузі та постановка задачі 9
1.1 Опис проблеми 9
1.2 Опис клієнтських технологій побудови додатків 12
1.2.1 Опис фреймворку Angular 12
1.2.2 Опис бібліотеки React 13
1.2.3 Опис фреймворку Vue.js 15
1.3 Опис технологій роботи з мапами 15
1.3.1 Опис бібліотеки Leaflet 15
1.3.2 Опис бібліотеки OpenLayers 17
1.4 Постановка задачі 18
2 Аналіз методів та критеріїв для дослідження 19
2.1 Визначення критеріїв оцінки клієнтських технологій побудови додатків 19
2.1.1 Маштабованість 19
2.1.2 Технічна підтримка 20
2.1.3 Продуктивність 20
2.1.3 Складність навчання та розробки 20
2.2 Визначення критеріїв оцінки фреймворків для роботи з мапами 21
2.2.1 Розмір супроводжуючого коду 21
2.2.2 Гнучкість та потужність 21
2.2.3 Документація 22
3 Розробка програмної системи 23
3.1 Формування вимог 23
3.2 Проектування програмної системи 24
3.3 Реалізація додатку 27

4 Оцінювання методів та засобів дослідження 32
4.1 Результати аналізу технологій побудови клієнтських додатків..... 32
4.2 Результати аналізу технологій для роботи з мапами 36
Висновки 39
Перелік джерел посилання 40
Додаток А Перелік джерел посилання кафедри програмної інженерії 42
Додаток Б Звіт результатів перевірки роботи на унікальність тексту 43
Додаток В Слайди презентації 44
Додаток Г Апробація результатів роботи 54

ВСТУП

Швидкість поширення інформації зростає кожен рік все більше і більше. Одним з найбільших інструментів поширення інформації є Інтернет. Він надає можливість мати доступ до мільйонів веб-сервісів різних спрямувань.

Але мало хто знає, що створення та підтримка таких сервісів є досить важкою працею.

З кожним роком з'являються нові тенденції та з цим зростає кількість різноманітних технологій розробки, які конкурують один з одним.

Метою дослідження є аналіз найпопулярніших технологій для розробки клієнтської частини додатків з використанням мап та розробка з їх використанням.

Для досягнення мети дослідження необхідно вирішити такі завдання:

– визначити найпопулярніші бібліотеки та фреймворки для порівняння;

– дослідити їх;

– визначити основні характеристики порівняння;

– розробити веб-сервіс щодо проведення міських квестів з використанням обраних технологій;

– визначити найефективніші технології для розробки системи.

Об'єкт дослідження – технології для створення клієнтської частини веб-систем та технології роботи з мапами.

Методи дослідження. Дослідження ґрунтується на аналізі певних технологій розробки веб-систем.

Практичне значення одержаних результатів полягає в можливості спрощення вибору певних технологій з урахуванням направленості системи, що розробляється.

Це дасть можливість визначити найбільш ефективних підхід до розробки клієнтського додатку, що працює з мапами.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Опис проблеми

Досить зрозуміло, що світ не стоїть на місці і дуже швидко розвивається. Та з цим ростуть потреби користувача. Користувач хоче якісного дизайну, зручного сервісу, швидкості та універсальності платформи, де він купує або використовує певну послугу чи річ.

Найчастіше для створення таких платформ та задовільнення потреб користувача, розробники використовують клієнт-серверну архітектуру (див.рис.1.1).

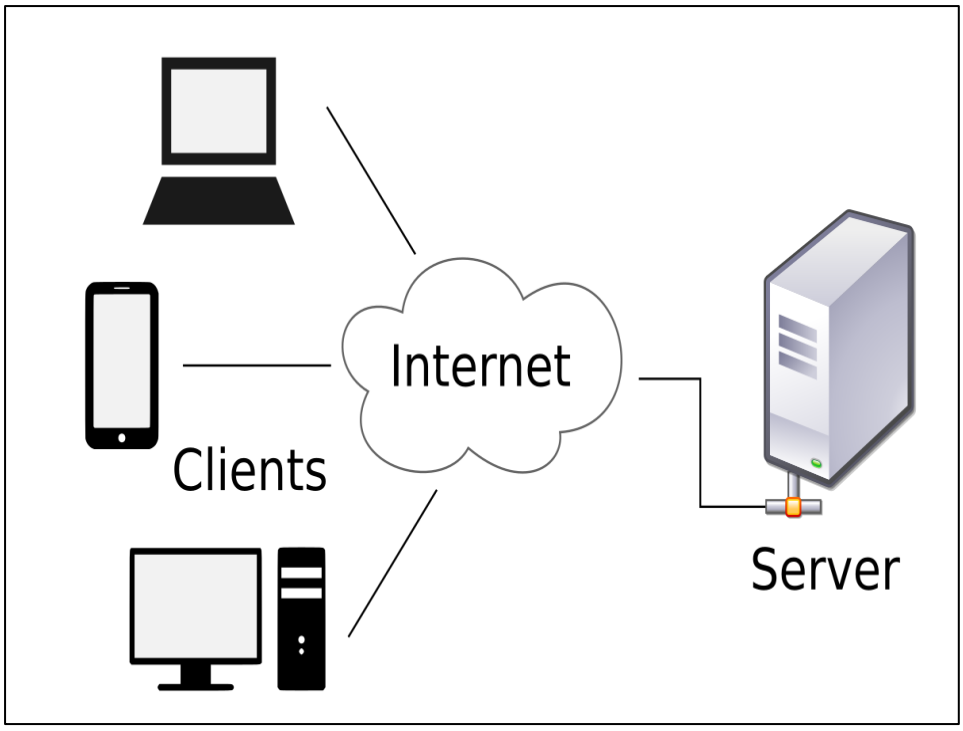

Рисунок 1.1 – Клієнт-серверна архітектура

Дана архітектура є шаблоном програмного забезпечення. Використовується для створення розподільних застосунків, зв'язок та обмін даних котрих проходить через мережу.

Основними компонентами даної архітектури є:

— сервери, що є джерелом інформація для програм, що звертаються до них;

— клієнти, котрі звертаються до серверів для отримання тої чи іншої інформації;

— мережа, що забезпечує зв'язок між серверами та клієнтами.

Також варто помітити, що сервери не повинні знати один про одного, та разом з цим не повинні знати про клієнтів. Клієнти у свої чергу не повинні знати про інших клієнтів та мають знати лише про сервери, до котрих звертаються.

Технології для розробки серверів є доволі постійними, тобто дуже часто вихід нового фреймворку є покращеною версією вже існуючої. Та найчастіше розробник обирає ту технологію, котра дозволяє працювати йому з рідною для нього мовою програмування.

Але технологій для розробки клієнтської частини додатків, наприклад JS фреймворків, з кожним роком стає все більше і більше.

JS фреймворки – це бібліотеки програмування JavaScript, в котрих є попередньо написаний код, що може бути використаний в стандартних функціях та задачах програмування.

Написання коду спокійно може протікати без їх використання, але вірно підібрана середа може значно облегшити роботу. Більш того, вони безкоштовні та з відкритим кодом.

У першу чергу це збільшить продуктивність. Це можна розглядати як свого роду обхідний шлях: потрібно буде писати менше коду вручну, тому що вже є попередньо написаний код та готові до використання функції та шаблони. Також можна створювати та розширювати попередньо створенні компоненти, а також використовувати патерни.

Фреймворки є більш адаптовані для дизайну веб-сайтів, й більшість розробників сайтів надають їм перевагу.

З урахуванням усієї кількості фреймворків, у розробників виникає питання, яку саме обрати для створення того чи іншого веб-застосування.

Найпопулярнішими JavaScript- фреймворків на даний момент є:

— React;

— Vue;

— Angular;

— Ember;

— Backbone.js.

Однак згідно з трендами Google Search, найбільшою популярністю користуються фреймворки React, Angular і Vue (див. рис. 1.2).

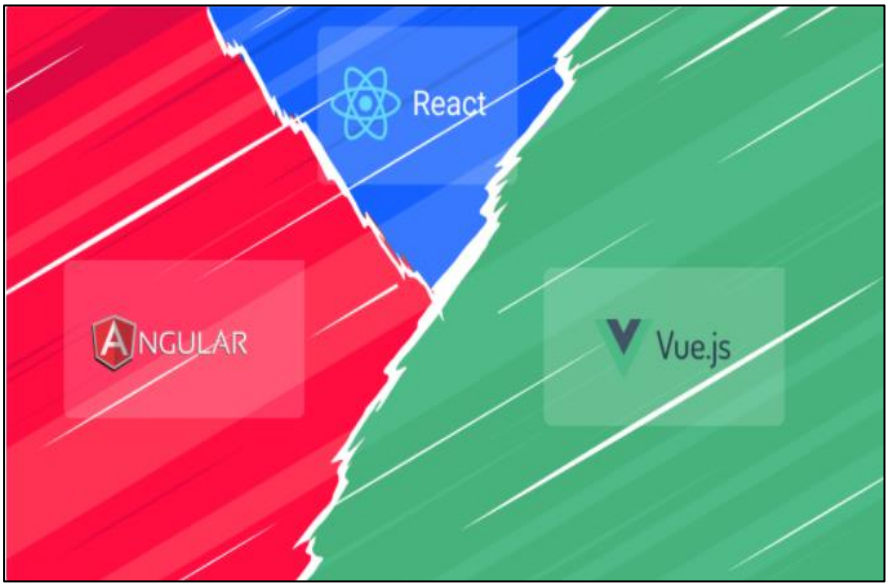

Рисунок 1.2 – Найпопулярніші JS-фреймворки

Вже не перший рік вони стають не лише найпопулярнішими запитами в пошукових сервісах, а й найбільш якісними та зручними для розробки фронтенду.

Відштовхнувшись від популярності і поширеності інструментів розробки для створення мап у клієнтських додатках можна виділити Leaflet.js та OpenLayers [10].

1.2 Опис web-технологій побудови додатків

1.2.1 Опис фреймворку Angular

Angular – це JavaScript фреймворк, що був випущений у світ у 2016 та створений на основі мови програмування TypeScript. Він розроблювався й підтримується фірмою Google та описується як JavaScript MVW(model-view-whatever)-фреймворк (див. рис. 1.3) [2].

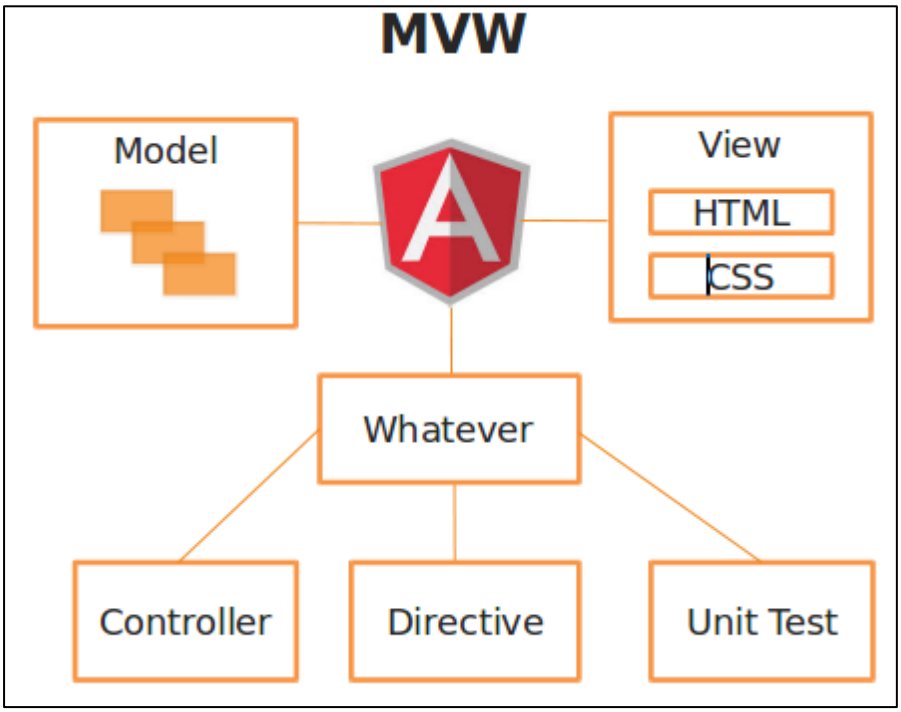

Рисунок 1.3 – Архітектура MVW

На даний момент його називають фрейворком, що найчастіше використовується для розробки односторінкових додатків [14] і він може похвалитися найбільшою спілкою розробників.

Angular надається з великою кількість функцій, які дозволяють розробити все від веб до мобільних додатків.

Вони мають покращене впровадження залежностей, а їх архітектура створена

на основі компонентів.

Взагалі впровадження залежностей, що також відомо як DI (dependency injection) використовується у фреймворці майже всюди, що дає можливість змінювати поведінку існуючого коду досить гнучко.

Тобто можна вдаватися в певні процеси роботи або ж можна повністю змінювати деякі частини продукту своїми.

Крім того гнучкість дозволяє реалізовувати багато добре відомих патернів, таких як Фабрика, Фасад, Одинак та інші.

Впровадження залежностей використовується починаючи від взаємодії компонентів, і закінчуючи створенням сервісів для відображення модальних вікон або нотифікацій користувачеві.

Даний фреймворк використовують такі системи як Google, Wix, weather.com, Forbes та ряд інших.

1.2.2 Опис бібліотеки React

React – це бібліотека для розробки інтерфейсів користувачів. Вперше вона була представленна у марті 2013 року [3].

Дана бібліотека розроблювалася та підтримується компанією Facebook.

Незважаючи на те, що React є більше бібліотекою ніж фреймворком, вона стоїть за призначеним для користувачів інтерфейсом таких відомих систем як Facebook та Instagram.

Даний факт свідчить про те, що фреймворк є досить ефективним у динамічних додатках з великим трафіком.

Взагалі React вважається найбільш швидкозростаючою JS бібліотекою. На сьогоднішній день налічується близько 1000 авторів даної бібліотеки на Github

(системі контролю версій).

Бібліотеку можна з легкістю інтегрувати у будь-яку архітектуру. У моделі MVC вона використовується, як View.

Зростання продуктивності забезпечується завдяки використанню віртуального DOM-дерева.

Також бібліотека надає можливість створити та використати повторно компоненти. Вони можуть бути використані у інших додатках (див. рис. 1.4).

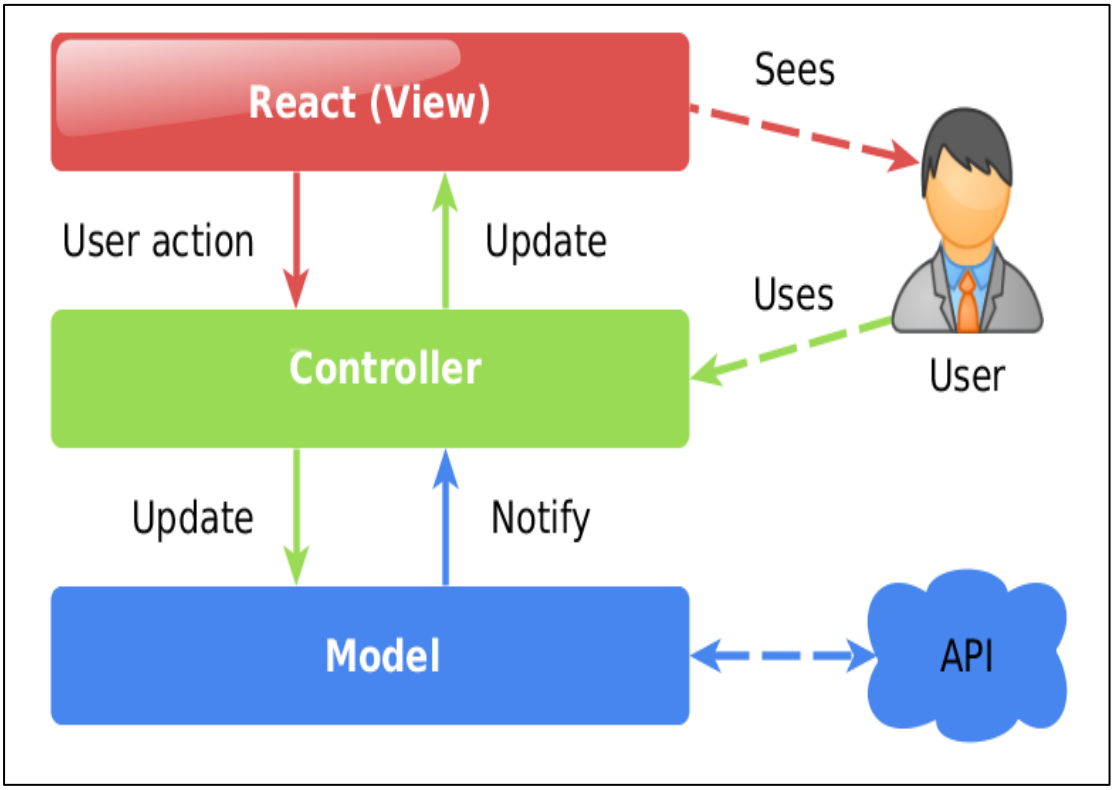

Рисунок 1.4 – React у MVC архітектурі

Розробка додатку з використанням даної бібліотеки є досить простою та легкою. Та гарно підходить для розробки додатків, що потребуються складних програмних рішень з великим навантаженням.

Дану бібліотеку використовують такі системи як Airbnb. Uber, Netflix, Twitter та ряд інших.

1.2.3 Опис фреймворку Vue.js

Фреймворк Vue.js 2.0 представлений у 2016 році. Кажуть, що для його створення узяли найкраще з Angular фреймворку та бібліотеки React, та поклали це у зручну для використання упаковку. Бібліотека довела, що може бути швидкою і компактною.

Vue.js має двосторонню прив’язку даних, візуалізація проходить на стороні серверу.

Vue.js вважається кращим вибором для швидкої розробки кросплатформних аплікацій. Найефективнішим даний фреймворк буде при створення додатку з високою продуктивністю.

Даний фреймворк використовують такі системи як Alibaba. Baidu, GitLab та ряд інших.

1.3 Опис технологій роботи з мапами

1.3.1 Опис бібліотеки Leaflet.js

Leaflet є однією з найкращих бібліотек JavaScript для створення mobile-friendly інтерактивних мап [5].

Дана бібліотека була започаткована у 2011 році.

Вона досить маленька, но при цьому підтримує дуже велику кількість функцій та плагінів, простий API. Також бібліотека гарно працює на мобільних та десктопних платформах. Але варто помітити, що вони повинні підтримувати стандарти HTML5 та CSS3 [15].

Leaflet підтримується більшістю світових компаній: GitHub, Flickr, Facebook,

Etsy та іншими.

Також проект має відкритий код, тому члени спільноти можуть стати й учасником репозіторія проекту та зробити його краще.

Вона не потребує від програмісту досвіду роботи с мапами та спрощує задачу підключення мапи до клієнтських додатків

Leaflet дозволяє використовувати різні публічні сервіси тайлів, що використовуються у якості джерел мап (див. рис. 1.5). Тайл – це порізані зображення мапи.

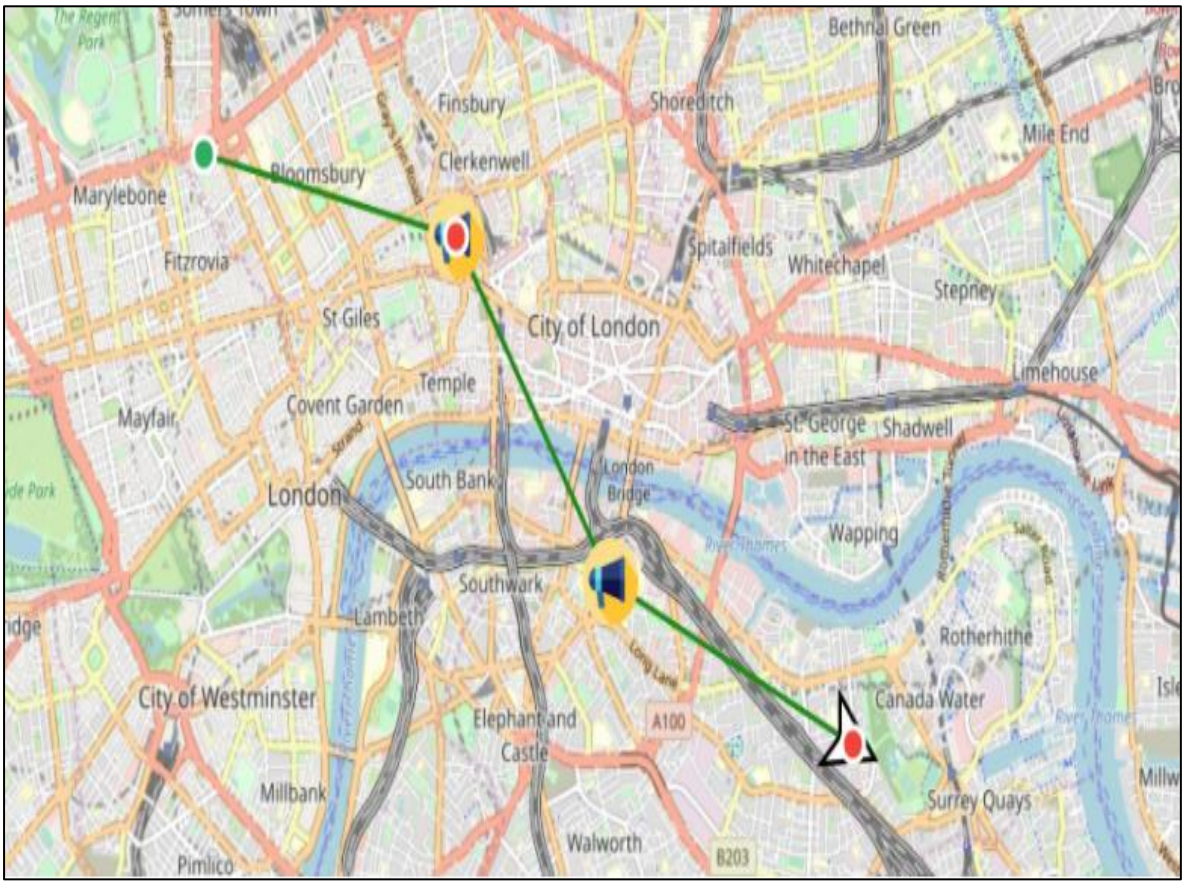

Рисунок 1.5 – Приклад роботи Leaflet бібліотеки

За проектування та розробку фреймворка відповідав Агафонкін Володимир. Під час виходу першої версії додатку він працював у компанії CloudMade. З 2013 року він працює в MapBox.

1.3.2 Опис фреймворку OpenLayers

OpenLayers – це бібліотека з відкритим кодом, що написана на мові програмування JavaScript. Основною її функцією є створення карт на основі програмного інтерфейсу [6].

Бібліотека є досить легкою для підключення у веб-додатках. Вона надає можливість працювати з різними картографічними матеріалами, котрі можуть мати різний формат.

Завдяки даній бібліотеці розробник може створити власну карту (див. рис. 1.6), що буде мати шари, які надані різніми серверами, наприклад MapServer, ArclMS, GeoServer.

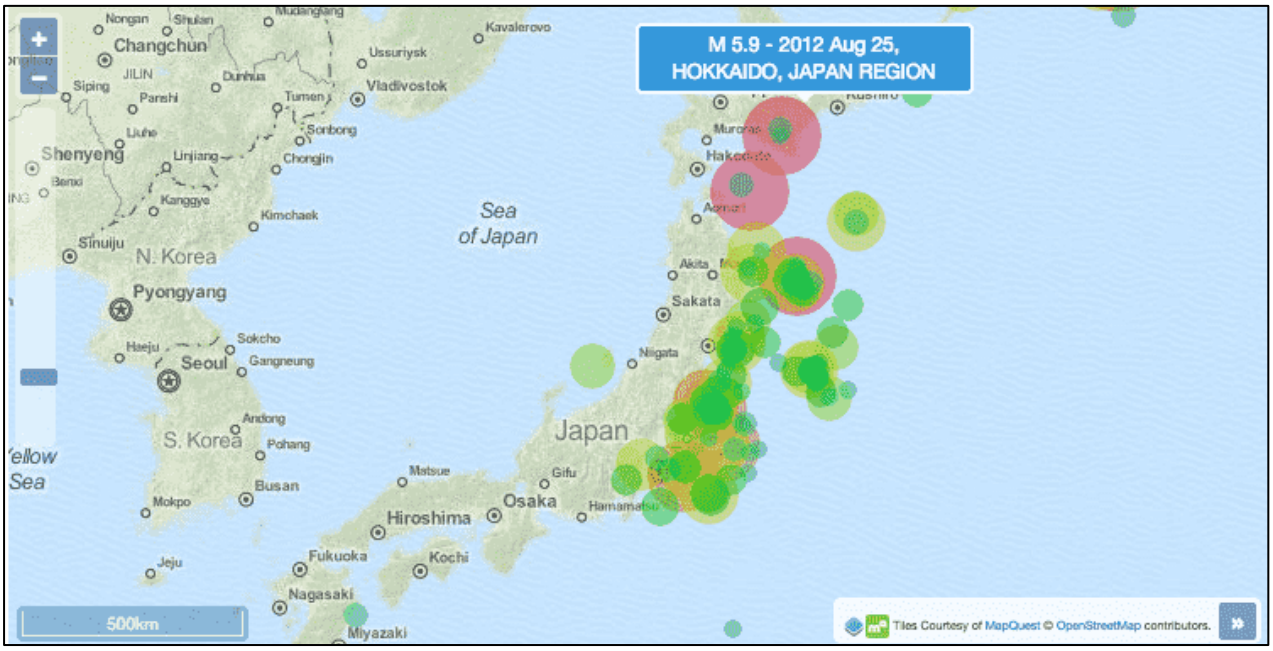

Рисунок 1.6 – Приклад роботи OpenLayers бібліотеки

OpenLayers був розроблений MetaCarta в проміжку між конференцією O'Reily Where 2.0 29-30 червня 2005 року та до Where 2.0 13-14 червня 2006 року. З листопаду 2007 року OpenLayers є проектом Open Source Geospatial Foundation.

1.4 Постановка задачі

Незважаючи на кількість js-фреймворків, різноманітних бібліотек для створення мап та аналізів, стосовно того, який з них все ж таки є переважним для створення того чи іншого продукту, ще неможливо зрозуміти найбільш ефективніші технології для створення веб-систем для роботи з мапами.

Саме тому, в рамках даної атестаційної роботи, будуть представлені результати дослідження різних характеристик одних з найпопулярніших фреймворків [7].

Метою даної роботи є дослідження методів та засобів розробки веб-застосувань з використанням мап.

Для досягнення мети атестаційної роботи необхідно буде розробити веб-сервіс щодо проведення міських квестів з використанням наведених технологій.

Мірою ефективності будуть слугувати обрані критерії для встановлення найефективніших технологій.

2 АНАЛІЗ МЕТОДІВ ТА ЗАСОБІВ ДЛЯ ДОСЛІДЖЕННЯ

2.1 Визначення критеріїв оцінки клієнтських технологій побудови додатків

Під час аналізу фреймворків для створення програмних систем потрібно обрати критерії для порівняння, які значно описують певні риси кожного та будуть слугувати факторами для обрання певних фреймворків [8].

Також важливо додати, що усі фреймворки будуть дослідженні у системі з клієнт-серверною архітектурою.

Дане рішення було прийняте, оскільки з даною архітектурою кожен з фреймворків буде відокремлений від серверу [13], що дозволить провести максимально точний аналіз.

Основними критеріями є:

– масштабованість;
– технічна підтримка;
– швидкодія;
– складність навчання та розробки.

2.1.1 Масштабованість

Масштабованість є одним з найважливіших критеріїв для порівняння.

Даний критерій відповідає за те, наскільки веб-застосування підтримує розширення функціональності.

За цим стоїть складність додавання нового функціоналу до вже існуючого та звязність між ними.

2.1.2 Технічна підтримка

Під час вибору будь-якої певного фреймворку не можна ігнорувати технічну підтримку програмістів.

Навіть прості функції можуть визвати складнощі у починаючого та іноді навіть маючого великий досвід програміста.

Чим складніше проект, тим частіше розробник звертається як до технічної документації обраної середи, так як і за порадою до спілки програмістів.

2.1.3 Швидкодія

Обираючи фреймворк, необхідно також думати о швидкодії.

Швидкодія у клієнтських застосуваннях – це перш за все швидкість відклику сторінки на дії користувача.

Вона надає користувачу інтерактивність та відчуття того, що застосування миттево реагує на дії користувача й перебудовується у відповідності до його потреб.

2.1.4 Складність навчання та розробки

Будь ми програмістом, який тільки починає свій шлях чи вже досвідним програмістом, який прийшов на новий проект, складність навчання js-фреймворку зіграє дуже важливу роль.

За нею ховається такі важливі пункти як:

— необхідність початкових знань певних мов програмувань;
— необхідність початкових знань певних технологій;
— складність налаштування фреймворку;
— кількість навчальних матеріалів;
— розмір та різноманітність можливостей фреймворку.

2.2 Визначення критеріїв оцінки фреймворків для роботи з мапами

2.2.1 Розмір супроводжуючого коду

Розмір супроводжуючого коду є мірою об’єму підключення певного фреймворку для роботи з мапами.

І на основі цієї міри можна вимірювати складність реалізації усіх необхідних функцій для роботи з картами.

Даний критерій є досить важливий до порівняння, оскільки він гарно показує наскільки складна є реалізація необхідних функцій для роботи з мапами та складність впровадження фреймворку.

2.2.2 Гнучкість та потужність

Гнучкість програмної системи є досить важливою характеристикою. Вона може досить сильно спростити роботу програміста при розробці програмних систем. Наявність багатої кількості функцій дозволить прискорити швидкість розробки, тому що не потрібно буде власноруч писати додатковий код. Та

найчастіше розробники фреймворку набагато гарніше розуміють як краще повинна бути реалізована певна функція, оскільки вони знайомі с усіма деталями та нюансами системи.

Та з цим також не потрібно забувати про потужність. Вона дуже сильно впливає на подальшу роботу системи з реальними клієнтами, оскільки затримка при роботі може стати досить важливою причиною для відмови від використання розробленого додатку.

2.2.3 Документація

У разі виникнення питань під час розробки, програмісти звертаються до документації.

Вона є дуже важливим інструментом, який надає інформацію про правильність написання та використання певних функцій системи. Наявність документації може значно спростити та прискорити роботу програміста.

Але крім того, що документація повинна бути, вона й повинна бути досить зрозумілою та простою для використання. Бо у разі незрозумілості або ж складності пошуку матеріалу, вона може не спростити, а навпаки, ускладнити роботу програмісту.

3 РОЗРОЗБКА ПРОГРАМНОЇ СИСТЕМИ

З метою повноцінної оцінки обраних фреймворків створимо веб-додатки з використанням кожного з них. Кожен з додатків буде представляти собою веб-сервіс щодо проведення міських квестів.

3.1 Формування вимог

Веб-сервіс, що розробляється має такі типи користувачів, як авторизований та неавторизований. Також авторизований користувач може бути звичайним користувачем та користувачем з правами адміністратора [17]. Виходячи з цього можемо виділити наступні функціональні вимоги до кожної з груп користувачів.

Незареєстрованому користувачу повинні бути надані наступні можливості:

— авторизація у випадку існування профілю;
— реєстрація у системі;
— перегляд усіх існуючих квестів;
— перегляд деталей будь-якого існуючого квесту.

Звичайному авторизованому користувачу повинні бути надані наступні можливості:

— перегляд усіх існуючих квестів;
— перегляд детальної інформації про квест;
— замовлення квесту;
— редагування власного профілю;
— перегляд усіх замовлених квестів;
— проходження усіх замовлених квестів.

Авторизованому користувачу з правами адміністратора повинні бути надані можливості звичайного авторизованого користувача та наступні:

— створення нових квестів;

— редагування існуючих квестів;

— підтвердження замовлених квестів;

— видалення та редагування користувачів.

Додаток повинен працювати з серверною частиною, котра повинна бути написана на мові програмування C# з використанням фреймворку ASP NET Core.

Авторизація повинна буду реалізована за допомогою JWT-токенів[16].

Робота з даними серверної частини повинна бути проведена завдяки фреймворку Entity Framework. Дані повинні зберігатися у базі даних MsSQL.

3.2 Проектування ПЗ

Виходячи з вимог до програмного забезпечення побудуємо UML діаграми (див. рис. 3.1) [12].

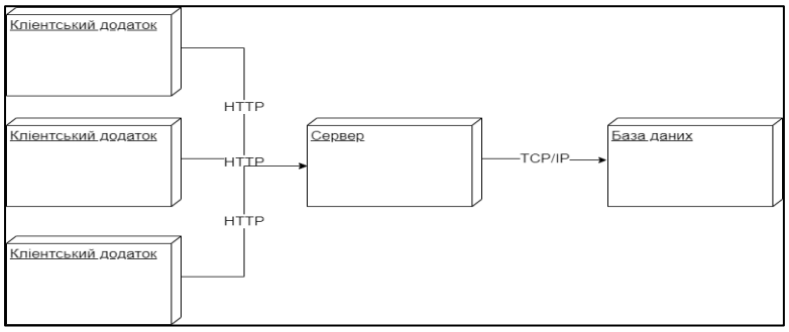

Рисунок 3.1 – Діаграма розгортання

Першою була побудована діаграма розгортання, що зображена на рисунку 3.1.

На ній відображують усі обчислювані вузли, які використовуються під час роботи.

Вона призначена для того, щоб візуалізувати елементи та компоненти програм, які існують тільки на етапі її використання.

Наступною побудуємо діаграму варіантів використання. Дана діаграма зображена на рисунку 3.2.

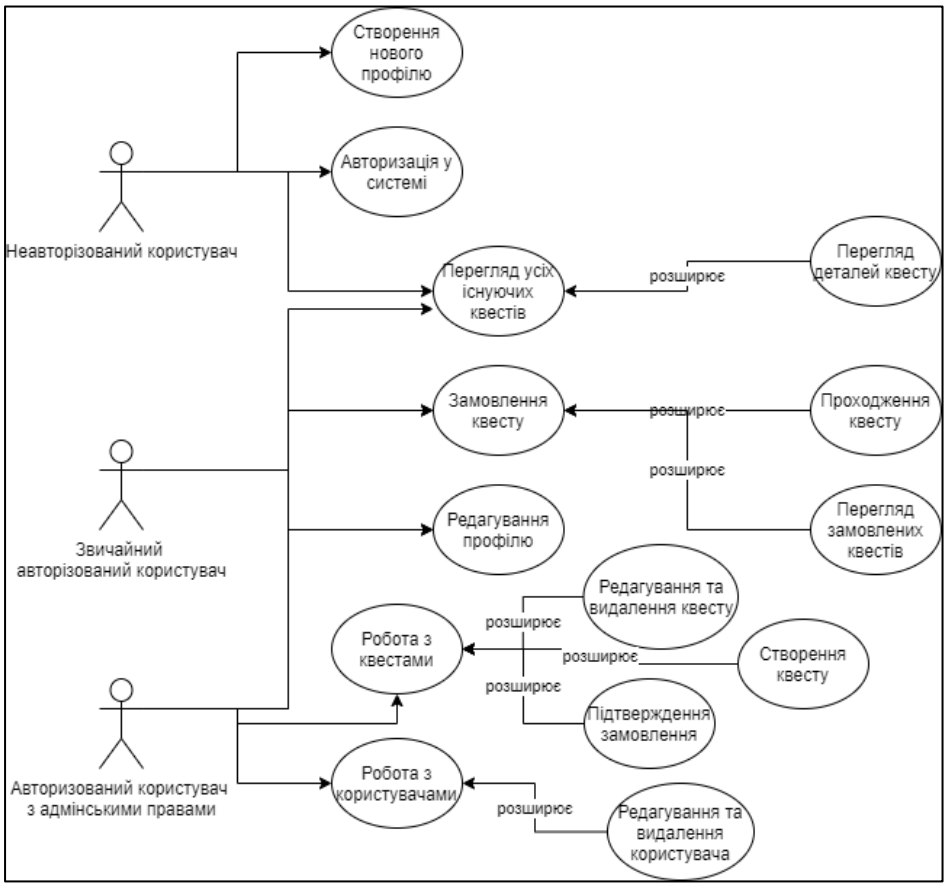

Рисунок 3.2 – Діаграма варіантів використання

Вона містить 3 актори:

– неавторизований користувач;

– звичайний авторизований користувач;

– авторизований користувач з правами адміністратора.

Діаграма зображує усі можливості, котрі вона повинна надавати користувачам різних типів.

Та останньою була змодельована діаграма кооперацій. Вона призначена для того, щоб описувати аспекти моделюючої системи у часі та показує яким чином об’єкти взаємодіють один з одним. Дана діаграма зображена на рисунку 3.3.

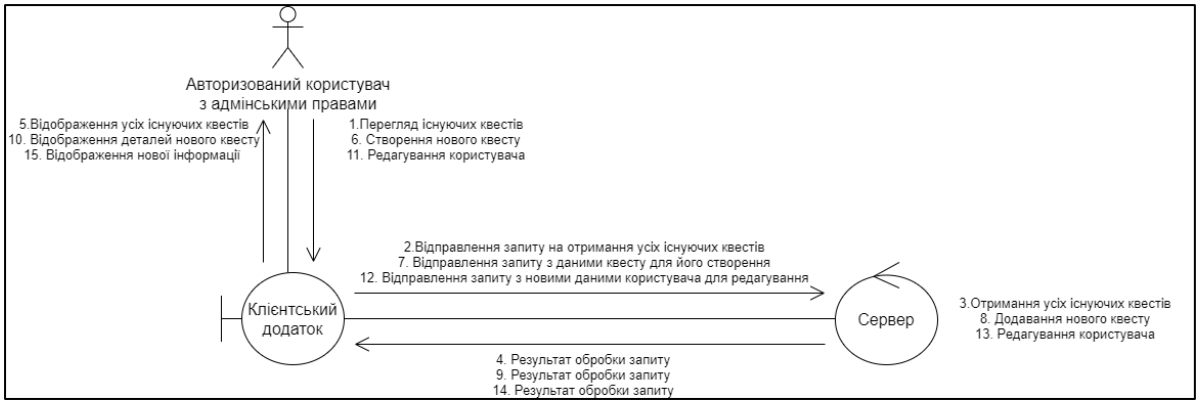

Рисунок 3.3 – діаграма кооперацій

Основними об’єктами діаграми є авторизований користувач з правами адміністратора, клієнтський додаток та сервер. Користувач напряму працює з клієнтським додатком. Після цього клієнтський додаток отримує запит на певну дію від користувача та відправляє запит до серверу на обробку. Сервер обробляє запит та відправляє відповідь. Після чого клієнтський додаток обробляє її та відображує результат користувачу.

3.3 Реалізація додатку

На основі визначених вимог розробимо веб-додатки з використанням фреймворків Angular, React та Vue.js [9].

До кожного з додатків підключимо бібліотеки для роботи за мапами Leaflet та OpenLayers.

Домашня сторінка є основною сторінкою додатка. Дана сторінка зображена на рисунку 3.4 для додатка, написаного використанням фреймворку Angular.

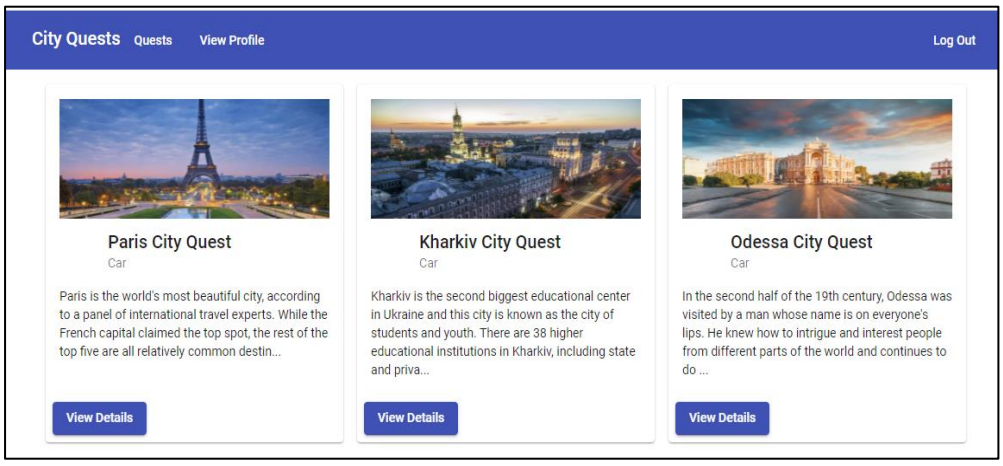

Рисунок 3.4 – Домашня сторінка додатка

Сторінка додатка, написаного з використанням бібліотеки React зображена на рисунку 3.5.

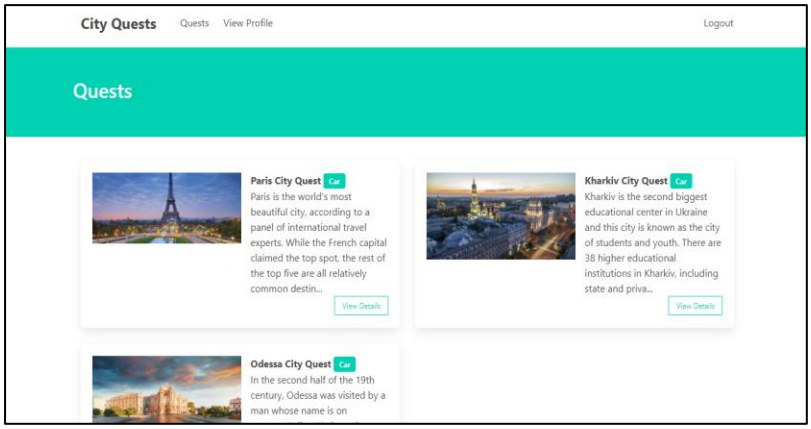

Рисунок 3.5 – Домашня сторінка додатку

На даній сторінці є два важливих для роботи користувача компоненти. Перший – це панель навігації, котра буде надалі доступна на кожній з сторінок

додатку.

Дана панель надає можливість користувачу перейти до домашньої сторінки, у випадку, коли він знаходиться на будь-якій іншій. На рисунку 3.6 зображена сторінка з використанням фреймворку Vue.js.

Та найголовніше, панель навігації надає можливість перейти на сторінку авторизації у випадку, коли користувач є неавторизованим користувачем.

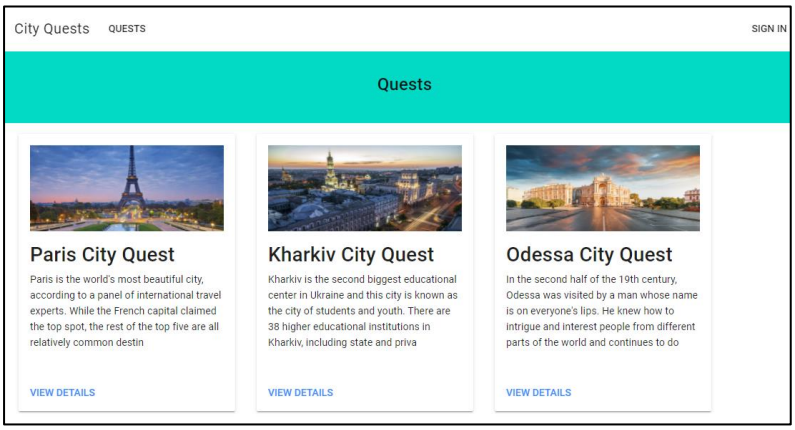

Рисунок 3.6 – Домашня сторінка додатку,

Для цього потрібно натиснути кнопку «Sign In», що знаходиться на панелі навігації з правої сторони.

Сторінка авторизації у додатку, що написаний на Angular, зображена на рисунку 3.7.

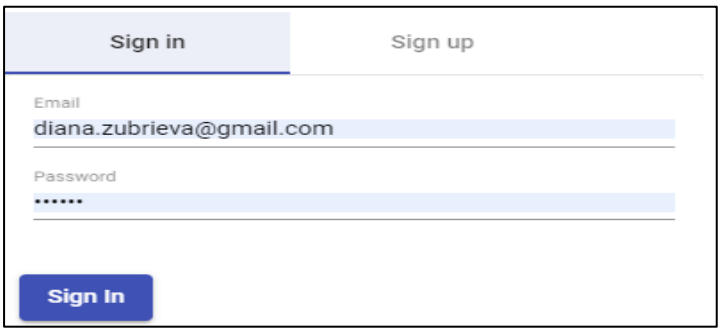

Рисунок 3.7 – Сторінка авторизації у додатку

Сторінка авторизації у додатку, що написаний на React, зображена на рисунку 3.8.

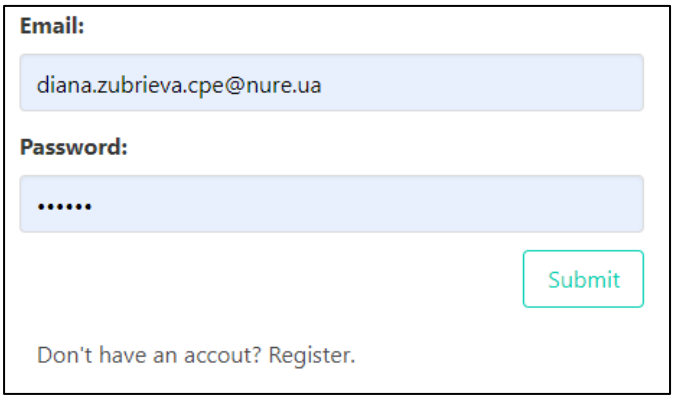

Рисунок 3.8 – Сторінка авторизації у додатку

Користувач повинен ввести адресу своєї електронної пошти та пароль, після чого, у разі існування профілю, він буде успішно авторизований у системі.

Сторінка авторизації додатку, що написаний з використанням бібліотеки Vue.js, зображена на рисунку 3.9.

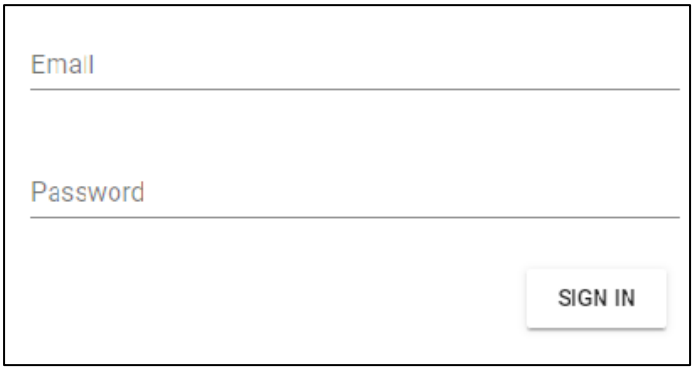

Рисунок 3.9 – Сторінка авторизації у додатку

Другим компонетом є компонент перегляду усіх існуючих квестів у системі. Даний компонент містить у собі невеликі компоненти, які відповідають за відображення часткової інформації про квест.

У випадку, коли користувач бажає переглянути детальну інформацію, йому потрібно перейти до сторінки детального перегляду. Для цього потрібно натиснути кнопку «View Details». Сторінка детального перегляду квесту у додатку, написаному з використанням фреймворку Angular зображена на рисунку 3.10. На даній сторінці зображена мапа з усіма локаціями та повний опис самого квесту.

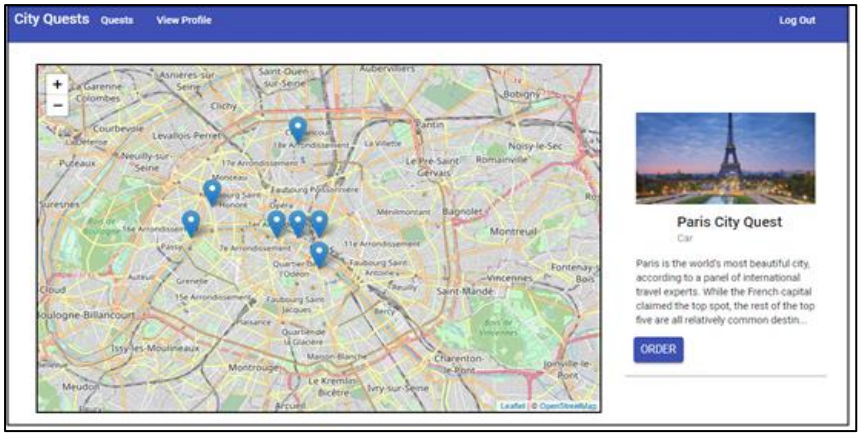

Рисунок 3.10 – Сторінка детального перегляду квесту

Сторінка перегляду квесту з використанням бібліотеки React зображена на рисунку 3.11.

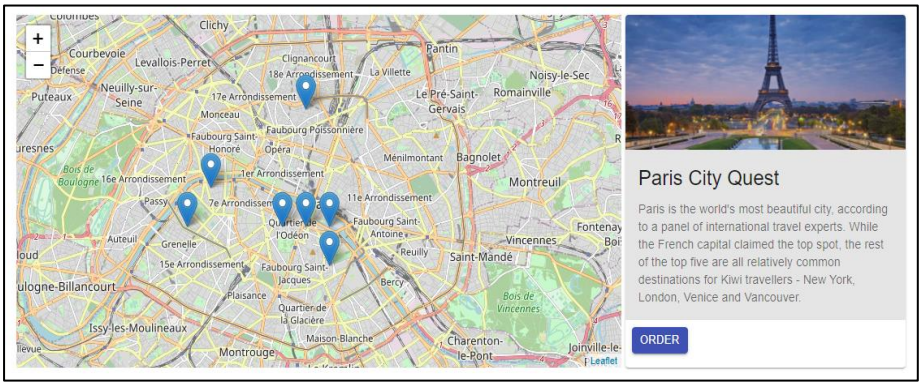

Рисунок 3.11 – Сторінка детального перегляду квесту

Сторінка з детального перегляду квесту з використанням бібліотеки Vue.js зображена на рисунку 3.12.

Користувач може перейти до сторінки замовлення квесту натиснувши кнопку «Order».

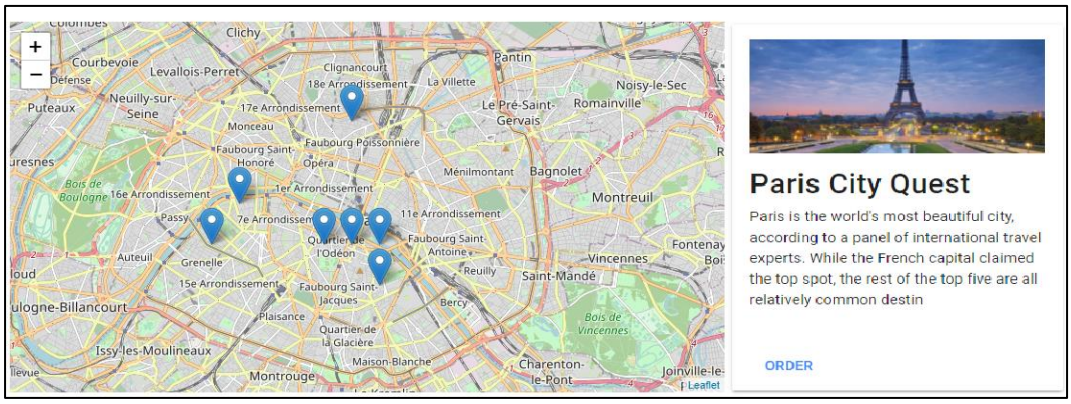

Рисунок 3.12 – Сторінка детального перегляду квесту

Після заповнення форми замовлення, квест буде у непідтвердженому стані. І тільки після того, як користувач з правами адміністратора підтвердить замовлення, квест буде доступний користувачеві.

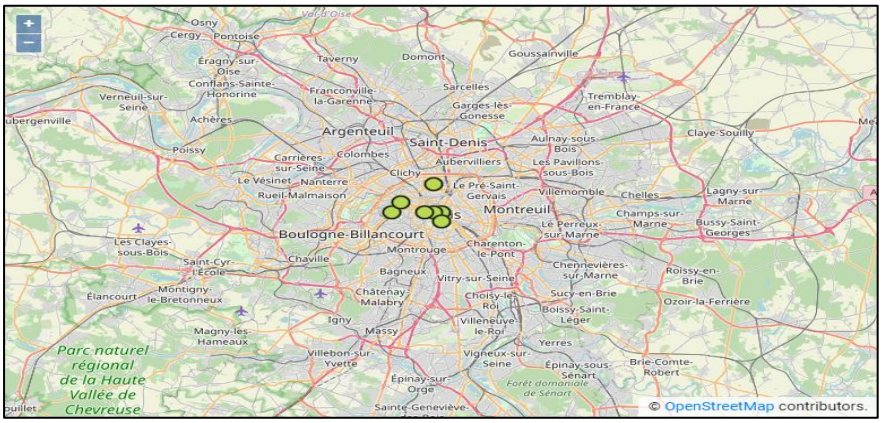

Рисунок 3.13 – Мапа з локаціями з використання бібліотеки OpenLayers

Варто зазначити, що на рисунках 3.10, 3.11 та 3.12 була використана бібліотека Leaflet.js, котра використовується для роботи з мапами. На рисунку 3.13 зображена мапа з локаціями квесту. У кожному з додатків вона має однаковий вигляд.

4 ОЦІНЮВАННЯ МЕТОДІВ ТА ЗАСОБІВ ДОСЛІДЖЕННЯ

Оцінювання фреймворків буде проводитися на основі методу лінійної згортки [14].

Основна ідея даного методу полягає в тому, що результуючий критерій

$$F = F(f_1^*(x), f_2^*(x), \dots, f_m^*(x))$$

зображують у вигляді лінійної комбінації одиничних показників

$$F = p_1 f_1^* + p_2 f_2^* + \cdots + p_m f_m^*$$

тобто у вигляді суми значень конкретних критеріїв, що помножені на числові коефіцієнти $p_1, p_2, \dots, p_m$.

4.1 Аналіз технологій побудови клієнтських додатків

Під час аналізу обраних технологій за критерієм масштабованості можна виділити Angular та React.

Angular фокусується на масштабованості завдяки своїй модульній системі розробки. Завдяки цьому більшість стилей, функцій, компонентів та модулей може бути використано декілька разів у різних частинах додатків.

Тобто кількість дублювання коду значно зменшується та звязність коду не є досить великою.

Та з цим додавання нової функціональності буде досить легким, оскільки потрібно буде створити окремий модуль, або ж тільки новий компонент у вже існуючий.

React у свою чергу використовує компонентний метод для отримання результата.

Це означає, що код є досить складним для повторного використання у інших частинах додатку.

Vue.js у порівнянні зі своїми конкурентами приблизно на тому ж рівні, що й React, оскільки має синтаксис на основі шаблонів.

Далі проаналізуємо обрані бібліотеки за критерієм технічної підтримки.

Технічна підтримка React організована досить непросто.

Наявність досить великої спілки розробників є великим плюсом, однак важно значни повноцінну документацію, котра описувала б ті чи інші процеси, функції та властивості бібліотеки.

Vue.js має гіршу підтримку, ніж React.

Кількість розробників у спільці є не досить великим, та й кількість матеріалів, що допомогла б у вивчанні не є значною.

Angular у свою чергу надає користувачу не тількі гарну документацію з описом функцій та властивостей, але й велику кількість мануалів з прикладами роботи. Та й спілка розробників є досить великою.

Порівняння за продуктивністю обраних бібліотек буде проводитись на основі наступних критеріїв:

— робота з DOM;

— значення основних метрік системи;

— використання пам’яті.

Оскільки створенні додатки є досить малими для порівняння такої продуктивності, використаємо один з найпопулярніших сервісів зі збору статистики Rawgit [11].

Оберемо бібліотеки у сервісі та подивимость на порівняльні таблиці.

Таблиця 4.1 – Результати вимірювання дій з DOM

Назва дії для заміру	Назва		
	Vue.js	Angular	React
Створення рядка	108.7(мс)	133.6(мс)	143.8(мс)
Заміна усіх рядків	93.5(мс)	105.5(мс)	111.1(мс)
Часткове оновлення	164.2(мс)	150.7(мс)	210.7(мс)
Вибір рядку	158.5(мс)	78.1(мс)	112.8(мс)
Обмін рядками	49(мс)	343.9(мс)	343.8(мс)
Видалення рядків	21.3(мс)	22.8(мс)	23(мс)
Створення багатої кількості рядків	928.7(мс)	1154.9(мс)	1506.3(мс)

Аналізуючи результати замірів, наведених у таблиці 4.1, можна сказати, що Vue.js у більшості випадків він показує гарний результат, окрім заміру з обранням рядку. Тому виходячи з результатів замірів можна сказати, що Vue.js досить швидкіше за своїх конкрутентів. Після нього вже йде Angular, та найгіршим є React.

Таблиця 4.2 – Результати вимірювання різних базових метрік

Назва	Vue.js	Angular	React
Відсутність дій (песимістичний TTI)	2114,5	2439,3	2579,7
Час завантаження сценарію	16	147,8	16
Загальна кількість кілобайт(ресурси)	197,8	294,7	274

Оцінюючи результати замірів, наведених у таблиці 4.2, можна сказати, що Vue.js є кращим, ніж свої конкуренти по кожному з замірів.

Аля в цьому випадку Angular та React міняються містами. Angular є найгіршим серед трьох обраних бібліотек.

Та останніми розглянемо заміримо використання пам’яті.

Таблиця 4.3 – Результати вимірювання використання пам’яті

Назва	Vue.js	Angular	React
Завантаження сторінки	1.2	1.8	1.3
Додавання 1000 рядків	4.4	4.2	3.5
Заміна 1000 рядків	4.1	4.9	5.3
Створення 1000 рядків	2.6	3.4	3

За даними результатами, що наведені у таблиці 4.3, можна сказати, що Vue.js знов є кращим за своїх конкуретів, а Angular є найгіршим.

Тобто виходячи с результатів усіх вимірів можна сказати, що Vue.js є найкращим за продуктивністю, друге місце посідає React, та останне Angular.

Далі порівняємо обрані бібліотеки за критерієм складності для навчання та розробки.

Angular є найскладнішою для навчання та розробки серед своїх конкурентів.

Вважається, що дана бібліотека більш підходить для професійних розробників, оскільки вона потребує вивчення TypeScript та MVC.

Під час вивчення та створення додатку на React можуть бути потрібні додаткові функції та для цього стороні бібліотеки.

Даний факт може як полегшити, так і ускладнити навчання, оскільки вивчення нових бібліотек не завжди є досить легкою справою.

Vue.js у порівнянні зі своїми конкурентами досить легкий у навчанні, оскільки він надає досить широкі можливості початкового налагодження проекту.

Аналіз усіх критеріїв оцінки було проведено. Тому можемо переходити до етапу отримання результату.

Оскільки аналіз було проведено без виставлення будь-яких оцінок, приведемо результати до безмірного значення.

Результати наведені у таблиці 4.4.

Таблиця 4.4 – Значення критеріїв

Назва	Vue.js	Angular	React
Масштабованість	0,25	0,5	0,25
Важкість навчання	0,4	0,3	0,3
Продуктивність	0,5	0,2	0,3
Технічна підтримка	0,2	0,5	0,3

Варто зазначити, що критерії мають різні вагові коефіцієнти.

Продуктивність та масштабованість значно важливішими за важкість навчання та технічну підтримку.

Тому встановимо вагові коефіцієнти для обраних критеріїв.

Вагові коефіцієнти задані таким чином, що масштабованість та продуктивність мають ваговий коефіцієнт 0.35, технічна підтримка має ваговий коефіцієнт 0.2 та важкість навчання має ваговий коефіцієнт 0.1.

Виконаємо згортку, оскільки маємо нормалізовані значення критеріїв та їх вагові коефіцієнти.

Результат згортки с ваговими коефіцієнтами має наступний вигляд:

Vue.js: 0.25*0.35 + 0.4 * 0.1 + 0.5*0.35 +0,2*0.2 =0.3425

Angular: 0.5 * 0.35 + 0.3 * 0.1 + 0.2 * 0.35 + 0.5 * 0.2 = 0.375

React: 0.25 * 0.35 + 0.3 * 0.1 + 0.3 * 0.35 + 0.3 * 0.2 = 0.2825.

За результатом згортки найкращим з фреймворків є Angular.

4.2 Аналіз технологій роботи з мапами

Розмір супроводжуючого коду у бібліотеки Leaflet.js є досить малим.

Для підлючення OpenLayers необхідна більша кількість коду, однак він є більш структурованим.

Порівнюючи дані бібліотеки за гнучкістю та продуктивністю можна сказати, що кожна з них показує гарні результати.

Мапи відображаються досить швидко як без додаткових елементів так і з ними. Обидві бібліотеки мають досить велику кількість функцій, котрі можуть бути потрібними розробнику.

Однак варто зазначити, що під час підключення і створення мап, більш простою у використанні була Leaflet.js. Крім того Leaflet.js може бути більш гнучкою завдяки додатковим плагінам.

За критерієм документації дані бібліотеки також схожі. Обидві мають гарні манули, приклади та документацію для початку роботи. Однак OpenLayers має досить багато застарілих прикладів, що може трохи заплутати розробника.

На основі даного аналізу проведемо оцінку критеріїв. Результати наведені у таблиці 4.5.

Таблиця 4.5 – Значення критеріїв

Назва	Leaflet.js	OpenLayers
Розмір супроводжуючого коду	0,6	0,4
Продуктивність та гнучкість	0,6	0,4
Технічна підтримка	0,5	0,5

Варто зазначити, що критерії мають різні вагові коефіцієнти. Розмір супроводжуючого коду є найменш важливішим серед критеріїв.

Тому встановимо вагові коефіцієнти для обраних критеріїв.

Вагові коефіцієнти задані таким чином, що:

– розмір супроводжуючого коду має ваговий коефіцієнт 0.2;
– продуктивність та гнучкість має ваговий коефіцієнт 0.5;
– та технічна підтримка має ваговий коефіцієнт 0.3.

Виконаємо згортку, оскільки маємо нормалізовані значення критеріїв та їх вагові коефіцієнти.

Результат згортки с ваговими коефіцієнтами має наступний вигляд:

Leaflet.js: 0.2*0.6 + 0.5 * 0.6 + 0.3*0.5 =0.57.

OpenLayers: 0.2 * 0.4 + 0.5 * 0.4 + 0.3 * 0.5 = 0.43.

За результатом згортки найкращою з бібліотек для роботи з мапами є Leaflet.

ВИСНОВКИ

У результати роботи було проведено аналіз предметної області. Також були проведені дослідження технологій розробки клієнтських частин програмних систем.

На основі отриманих результатів аналізу було обрано найефективніші технології для розробки веб-сервісу щодо проведення міських квестів.

В результаті аналізу було встановлено критерії ефективності фреймворків, за допомогою котрих можна створити веб-сервіси для роботи з мапами.

За результатами обраних критеріїв було проведено порівняльну характеристику та встановлені оцінки для наданих фреймворків.

Результатами реалізації стали три веб-сервіса щодо проведення міських квестів.

В результаті аналізу і розробки поставлену задачу було виконано в заданому обсязі.

Також потрібно зауважити, що дану тему потрібно проаналізувати та дослідити також на інших розмірах застосувань.

Необхідність досліджень даної області ґрунтується на постійному розвитку Інтернет технологій, що у свою чергу приводить до створення нових бібліотек та фреймворків та складності обрання програмістом необхідних технологій для розробки застосування.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Сандерсон С. ASP.NET MVC Framework з прикладами на C# для професіоналів [Текст] / С. Сандерсон. – М.: ООО «И.Д. Вильямс», 2010. – 560 с.

2. Документація Angular [Електронний ресурс]. Режим доступу до ресурсу: www/URL: https://angular.io/.

3. Документація React [Електронний ресурс]. Режим доступу до ресурсу: www/URL: https://ru.reactjs.org/.

4. Документація Vue [Електронний ресурс]. Режим доступу до ресурсу: www/URL: https://vuejs.org/.

5. Документація Leaflet [Електронний ресурс]. Режим доступу до ресурсу: www/URL: https://leafletjs.com/.

6. Документація OpenLayes [Електронний ресурс]. Режим доступу до ресурсу: www/URL: https://openlayers.org/.

7. The Ultimate Comparison: Angular vs React vs Vue [Електронний ресурс] / портал sloboda studio: Колективний блог. –- Режим доступу до ресурсу: www/URL: https://sloboda-studio.com/blog/the-ultimate-comparison-angular-vs-react-vs-vue/.

8. The Ultimate Comparison: Angular vs React vs Vue [Електронний ресурс] / портал sloboda studio: – Режим доступу до ресурсу: www/URL: https://sloboda-studio.com/blog/the-ultimate-comparison-angular-vs-react-vs-vue/.

9. Angular vs React vs Vue [Електронний ресурс] / портал athemes: – Режим доступу до ресурсу: www/URL: https://athemes.com/guides/angular-vs-react-vs-vue/.

10. Leaflet vs OpenLayers. What to choose? [Електронний ресурс] / портал geoapify – Режим доступу до ресурсу: www/URL: https://www.geoapify.com/leaflet-vs-openlayers.

11. Ragfit [Електронний ресурс] / портал Ragfit – Режим доступу до ресурсу: www/URL: https://rawgit.com/.

12. Мартін Фаулер. UML. Основи. Короткий посібник з стандартному мови об'єктного моделювання: Пер. с англ. — М.: Наука, 2003. – 208 с.

13. Мартін Фаулер. Архитектура корпоративных программных приложений: Пер. с англ. – Російська редакція, 2004. – 544 с.

14. Michael Mikowski, Josh Powell. Single Page Web Applications. – Manning Publications Co., 2013. – 432 с.

15. Jon Duckett. HTML and CSS: Design and Build Websites. – Paperback, 2011. – 512 с.

16. M. Jones, J. Bradley, N. Sakimura, JSON Web Token (JWT). RFC 7519. 2015. С. 29.

17. Kachko O., N. Bilous , Semerkov V.Research on methods for secure web applications development Information Technologies in Innovation Business (ITIB), 7-9 October, 2015, Kharkiv, Ukraine Proceedings of ITIB, p.26-27, ISBN 978-966-659-214-2.

18. Кузенков П., Дударь, З.В., Вечур, А.В. Методи порівняння успішності академічних груп. Східно-Європейський журнал передових технологій - 2010. - № 4/2. - С. 52-55.