МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет Кібербезпеки, комп'ютерної та програмної інженерії
Кафедра комп'ютеризованих систем управління

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри

_____ Литвиненко О.Є.

«___» _____2021 р.

ДИПЛОМНИЙ ПРОЄКТ

(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСКНИКА ОСВІТНЬО–КВАЛІФІКАЦІЙНОГО РІВНЯ "БАКАЛАВР"

Тема: Веб-додаток «Туристичний гід»

Виконавець: Безвіконний О.В.

Керівник: Масловський Б.Г.

Нормоконтролер: Тупота Є.В.

Київ 2021

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет Кібербезпеки, комп'ютерної та програмної інженерії
Кафедра комп'ютеризованих систем управління

Освітньо-кваліфікаційний рівень бакалавр
Спеціалізація 6.050102 "Системне програмування"
(шифр, найменування)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Литвиненко О.Є.
«_____» _____2021 р.

ЗАВДАННЯ
на виконання дипломної роботи (проекту)
Безвіконного Олексія Володимировича
(прізвище, ім'я, по батькові випускника в родовому відмінку)

1. Тема проєкту (роботи): Веб-додаток «Туристичний гід»
затверджена наказом ректора від "04" лютого 2021 року №135/ст.

2. Термін виконання проєкту (роботи): з 17.05.2021 до 20.06.2021

3. Вихідні дані до проєкту (роботи): інформація про розвиток сучасних веб-додатків в туристичній сфері; мови програмування *JavaScript* та *PHP*; вимоги до оформлення дипломних робіт.

4. Зміст пояснювальної записки (перелік питань, що підлягають розробці):
1) аналіз принципів побудови веб-додатків в туристичних системах;
2) веб-додаток «Туристичний гід»;
3) розробка веб-додатку.

5. Перелік обов'язкового графічного матеріалу:
1) діаграма прецедентів;
2) діаграма кооперації;
3) алгоритм реєстрації користувача;
4) інтерфейс головної веб-сторінки.

6. Календарний план

№ п/п	Етапи виконання дипломного проєкту	Термін виконання етапів	Примітка
1	Аналіз літератури та джерел за темою дипломного проєкту	17.05.2021-18.05.2021	
2	Написання першого розділу	19.05.2021-22.05.2021	
3	Вибір функцій майбутнього додатка	23.05.2021-24.05.2021	
4	Написання другого розділу	25.05.2021-27.05.2021	
5	Розробка веб-додатка	28.05.2021-01.06.2021	
6	Написання третього розділу	02.06.2021-04.06.2021	
7	Оформлення пояснювальної записки	05.06.2021-07.06.2021	
8	Підготовка графічного та демонстраційного матеріалу	08.06.2021-10.06.2021	

7. Дата видачі завдання «17» травня 2021 р.

Керівник дипломного проєкту _____ Масловський Б.Г.
(підпис)

Завдання прийняв до виконання _____ Безвіконний О.В.
(підпис випускника) (П.І.Б.)

РЕФЕРАТ

Пояснювальна записка до дипломного проєкту роботи «Веб-додаток «Туристичний гід»» викладена на 46 с., містить 23 рис., 4 таблиці, 13 літературних джерел.

Ключові слова: веб-додаток, турист, гід, екскурсія, туризм, маршрут

Об’єкт дослідження: загальні принципи побудови веб-систем які пропонують туристичні послуги – онлайн екскурсії та туристичні гіди.

Предмет дослідження: мультиплатформенний веб-додаток для організації екскурсії.

Мета дипломного проєкту: створення інформаційного веб-додатку для взаємодії людей з послугами туризму.

Методи дослідження: створення веб-додатку використовуючи мови програмування *JavaScript* та *PHP*.

Рекомендації щодо використання результатів: Результатами проєкту зможуть користуватись туристи та подорожуючі, жителі Києва, які зацікавлені в дослідженні міста, педагоги, що організовують просвітницьку та виховну роботу з дітьми, молоддю, туристичні фірми, підприємства туристичної інфраструктури, міська рада.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ СКОРОЧЕНЬ ТА ТЕРМІНІВ

CSS – Cascade Style Sheets

HTTP – Hyper Text Transfer Protocol

JS – JavaScript

MAMP - Macintosh, Apache, MySQL, PHP

PHP – Personal Home Page

SQL - Structured Query Language

VS – Visual Studio

СУБД – Система управління базами даних

ВСТУП

На сьогоднішній день впровадження інформаційних технологій в усі сфери діяльності людини, суспільства, держави є звичайною справою. Вони стали невід'ємною частиною більшості процесів. Їх використання призводить до поліпшення якості продукції, прискорення обробки інформації, обміну інформацією та доступу до різноманітних інформаційних джерел, зменшення трудових і економічних витрат.

Засоби зв’язку дозволяють об’єднувати комунікаційні системи у глобальну мережу. Завдяки чому людина отримує можливість обмінюватися інформацією в межах всієї планети, не залежно від кордонів і відстаней. Інтернет являє собою надзвичайно перспективний засіб комунікації, що пояснює його сучасний надзвичайний розвиток та популярність.

За останні декілька десятків років завдяки бурхливому розвитку Інтернету в програмуванні виділився окремий напрямок – веб-програмування. Веб-програмування – це галузь веб-розробки і різновид дизайну, в завдання якої входить проєктування користувальницьких веб-інтерфейсів або ж веб-додатків. Сьогодні веб-додатки використовуються в усіх сферах діяльності людини, що набагато спрощує процес комунікації. Туристична сфера не є винятком.

Туризм виконує одну з основних функцій у світовій економіці, формуючи значну частину світового валового продукту. В останні доепідеміологічні роки туризм був одним з найприбутковіших видів бізнесу в світі. Ця галузь економіки розвивалася високими темпами. Щорічне зростання інвестицій в індустрію туризму становило близько 30%. У цій галузі задіяно приблизно 6% всього світового капіталу. Успіх туристичного бізнесу завжди залежав від кількості клієнтів. Проте, у зв’язку з епідеміологічною ситуацією в світі, сфера туризму зазнала великих економічних втрат.

За даними Всесвітньої туристської організації при ООН світовий туризм зможе повернутися до докоронавірусного рівня не раніше, ніж через два роки. Криза через пандемію стала найсильнішою в сфері туризму за всю історію. За даними організації,

втрати сфери туризму в 2020 році склали 1,3 трлн доларів. Кількість міжнародних поїздок знизилася на 74%. Число туристів, які подорожують в інші країни, знизилося в минулому році на 1 млрд в порівнянні з показником 2019 року.

Туристичні послуги, а саме послуги проведення екскурсій з великим скупченням людей наразі досить складний, а іноді навіть неможливий процес. Такі підстави згенерували потребу у розробленні сучасних інформаційних технологій, що спрямовані на підвищення рівня надання туристичних послуг. Використання інформаційних технологій у галузі туризму набирають попит в екскурсійних бюро, а також у разі надання послуг окремим туристам та туристичним групам. Тому послуги онлайн екскурсій набирають велику популярність серед туристів та мають безліч переваг порівняно зі звичайними на сьогоднішній день.

Актуальність теми дипломного проєкту полягає у популярності використанні веб-додатків у туристів**.**

Мета і завдання виконання дипломної роботи. Метою роботи є отримання готового інформаційного веб-додатка для взаємодії людей з послугами туризму, а саме – онлайн екскурсій, за допомогою персональних комп'ютерів, мобільних телефонів, планшетів та інших пристроїв, які мають доступ до всесвітньої глобальної мережі.

Об’єктом дослідження є пропозиції та веб-системи організацій, які пропонують туристичні послуги – онлайн екскурсії та туристичні гіди .

Предметом дослідження є веб-додаток для послуг онлайн туризму

Методи дослідження. Вивчення теорії та використання її на практиці для проектування і розробки веб-додатка.

Наукова новизна отриманих результатів. В результаті розроблено веб-додаток за допомогою якого можна надати жителям столиці та подорожуючим якісну, перевірену інформацію про туристичну інфраструктуру, історичні, культурні та природні пам’ятки Києва в одному зручному для використання веб-додатку.

Практичне значення отриманих результатів. Результатами проєкту зможуть користуватись туристи та подорожуючі, жителі Києва, які зацікавлені в дослідженні

свого міста, педагоги, що організовують просвітницьку та виховну роботу з дітьми, молоддю, туристичні фірми, підприємства туристичної інфраструктури, міська рада.

РОЗДІЛ 1
АНАЛІЗ ПРИНЦИПІВ ПОБУДОВИ ВЕБ-ДОДАТКІВ В ТУРИСТИЧНИХ СИСТЕМАХ

1.1. Використання веб-додатків, веб-сайтів та веб-сторінок

Одним з ключових моментів в розвитку всесвітньої павутини грає веб-розробка - процес створення веб-сайту або веб-додатка. Термін включає розробку додатків електронної комерції, веб-дизайн, програмування на стороні клієнта і серверу, а також конфігурування веб-серверу. Основними етапами веб-розробки є:

- проєктування сайту або веб-додатка;
- створення макетів сторінок;
- наповнення;
- обслуговування працюючого сайту або його програмної основи.

Залежно від поточної задачі деякі з етапів можуть бути відсутні, або бути тісно пов'язані один з іншим.

Веб-сторінку можна визначити як окрему сторінку веб-сайту. Доступ до веб-сторінки можна отримати за допомогою однієї *URL*-адреси. Прості веб-сторінки в основному складаються з простої архітектури на основі мови гіпертекстової розмітки *HTML* і служать лише платформою для відображення інформації. Веб-сторінки можуть містити текст, зображення або музику. Відображення сторінки можна змінити додаванням до неї таблиці стилів *CSS* або сценаріїв на мові *JavaScript*. Перегляд веб-сторінки не вимагає навігації, на відміну від веб-сайту.

Веб-сайт – це колекція веб-сторінок, з’єднаних між собою, які доступні через мережу, таку як Інтернет і мають єдине доменне ім’я. Він може містити текст, графіку, аудіо, відео, гіперпосилання на інші сторінки тощо. Веб-сайт може бути специфічним для конкретної галузі, конкретним продуктом або конкретними послугами. Вони призначені для ознайомлення відвідувачів з інформацією про їхню галузь, продукти або послуги. Наприклад, звичайний веб-сайт може містити детальну інформацію про серію товарів, але у нього немає можливості замовнику замовляти товар та здійснювати платежі через веб-сайт. Для цих та інших цілей, пов’язаних з інтерактивом або взаємодією користувача (клієнта) за сервером розробляються веб-додатки. Приклад структури веб-сайту зображений на рис. 1.1.

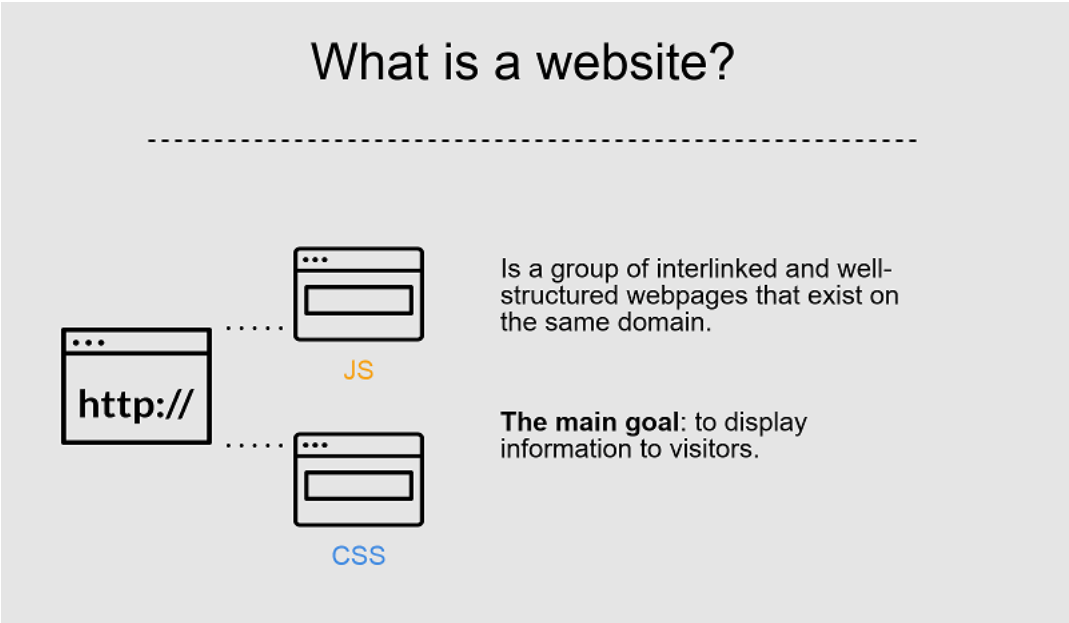

Рис. 1.1. Структура веб-сайту

Веб-додаток - це інтерактивна комп'ютерна програма, розроблена для мережі інтернет, що дозволяє користувачам вводити, отримувати і маніпулювати даними за допомогою взаємодії. Додаток може бути вбудований у веб-сторінку, або сама веб-сторінка може бути додатком. Логіка додатка зосереджена на сервері, а браузер найчастіше відповідає лише за відображення інформації, завантаженої з сервера, і за передачу на сервер даних користувача. Веб-додаток отримує запит від клієнта і виконує обчислення, після цього формує веб-сторінку і відправляє її клієнту через

мережу з використанням протоколу *HTTP*. Саме веб-додаток може виступати як клієнт інших служб, наприклад, бази даних або іншого веб-додатку, розташованого на іншому сервері. Яскравим прикладом веб-додатку є система управління вмістом статі Вікіпедії: безліч її учасників можуть приймати участь в створенні мережної енциклопедії, використовуючи для цього браузери своїх операційних систем (будь то *Windows, GNU/Linux, IOS* або будь-яка інша операційна система) і не завантажуючи додаткових виконуваних модулів для роботи з базою даних статі. Приклад структури веб-додатку зображений на рис. 1.2.

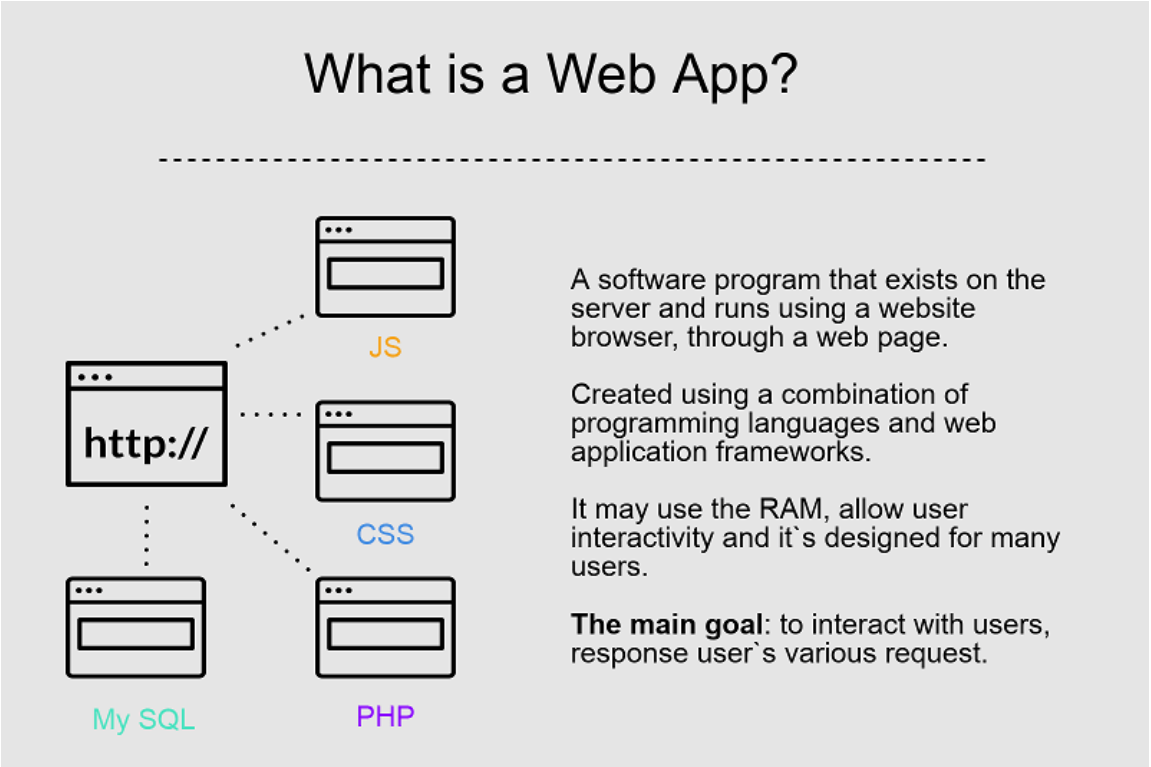

Рис. 1.2 Структура веб-додатку

1.2. Особливості веб-сайтів та веб-додатків

1.2.1. Авторизація

Веб-додатки пропонують процес авторизації користувача. Цей процес включає в себе введення призначених для користувача даних для отримання доступу до веб-сайту або системі. У деяких випадках це використовується для того, щоб дати більше

можливостей, які недоступні неавторизованим користувачам. Таким чином веб-додатки дозволяють своїм відвідувачам обмінюватися миттєвими повідомленнями (чат-платформи, соцмережі, блоги), створюють контент на основі призначених для користувача переваг, забезпечують необмежений доступ до них, використовують міні-вбудовані програми для розваг.

Процес авторизації важливий для систем, яким потрібно будь-яка особиста інформація про клієнта. Більш того, на цьому етапі, особливу увагу приділяють безпеці. Важливо мінімізувати можливість доступу до особистих даних користувачів стороннім особам.

1.2.2. Інтерактивність

Сайти не пропонують можливості взаємодії з програмою. Його користувачі не мають доступу до розміщення своєї інформації крім заповнення форми для отримання підписки. Найбільш яскравими прикладами типових сайтів є новинні, кулінарні, прогнози погоди.

В той час як веб-додатки створені для взаємодії з користувачами. Веб додатки теж можуть бути інформативними, але також вони можуть і обробляти інформацію, отриману від користувачів.

Як приклад веб-додатку, розглянемо системи онлайн банкінгу «Приват24» (рис.1.3 та рис. 1.4). Для того, щоб користуватись основними функціями веб-додатку (наприклад переказ коштів, поповнення мобільного, оплата онлайн послуг та інші), користувачеві потрібно авторизуватись, тобто ввести свій номер телефону та пароль. Або ж створити акаунт.

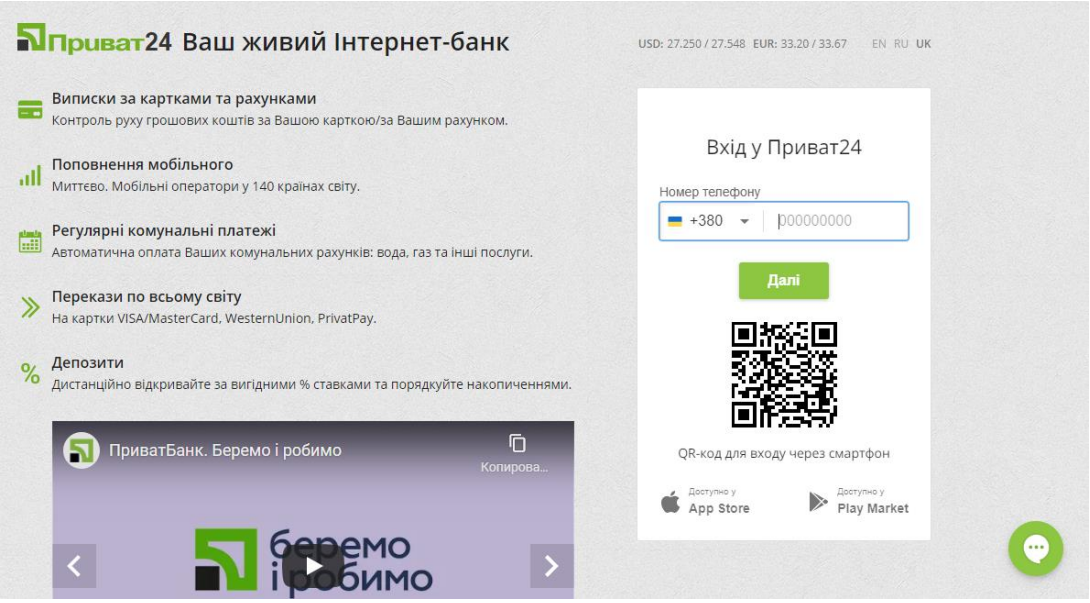

Рис. 1.3. Авторизація у веб-додатку «Приват24»

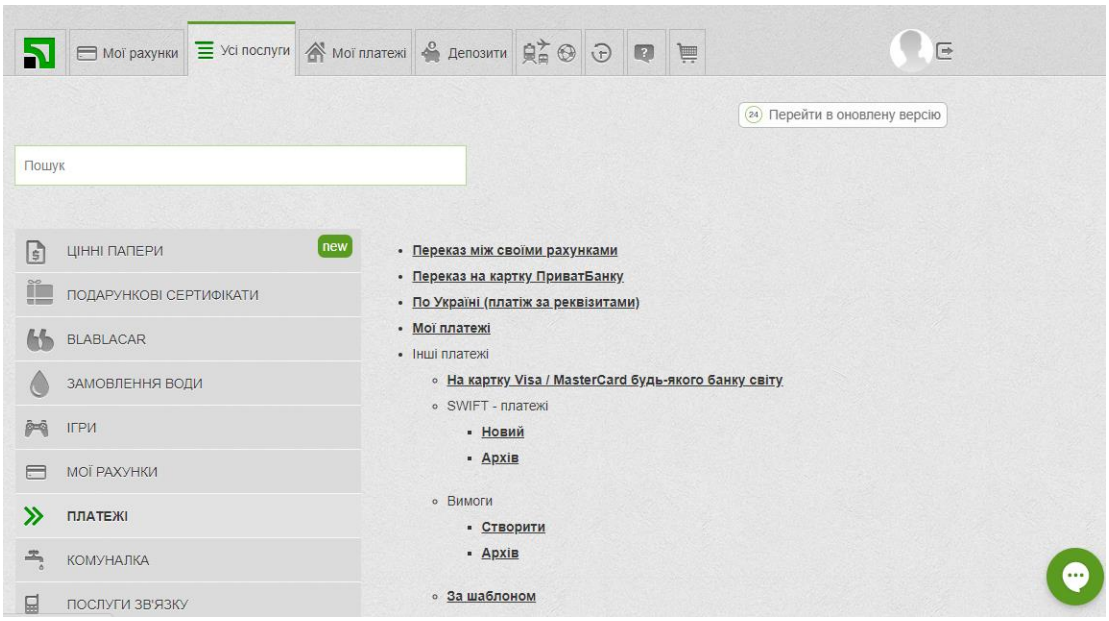

Рис. 1.4. Послуги веб-додатку «Приват24»

1.2.3. Переваги розробки веб-додатків

Незалежність від операційної системи клієнта. Додаток створюється один раз для довільно вибраної платформи і на ній розгортається. Замість того, щоб писати різні версії для *MS Windows, Mac OS X, GNU/Linux* та інших операційних систем, додаток створюється один раз і відображається на будь-якій платформі.

Відсутність програмного забезпечення для клієнтів**.** Не потрібно нічого завантажувати і встановлювати. Це спрощує процес користування. Вирішивши запустити свій проект веб додатку (на відміну від десктопного), ви відразу ж зробите його доступним для масової аудиторії.

Доступність**.** Програма доступна з будь-якої точки планети, де є доступ до Інтернету.

Адаптивність**.** Можливість користуватися додатком, використовуючи будь-який пристрій (настільний комп'ютер або ноутбук, смартфон або планшет, тощо) з будь-яким розрішенням екрану. Для цього при розробці потрібно використовувати адаптивну верстку. Адаптивна верстка – підхід, що припускає зміну дизайну залежно від поведінки користувача, розміру екрана, платформи і орієнтації девайса. Іншими словами, сторінка повинна автоматично підлаштовуватися під дозвіл, змінювати розмір картинок і так далі. Це дозволить усунути потребу в розробці дизайну для кожного нового пристрою, що з'являється у продажу.

Просте та зручне оновлення. Веб додаток набагато простіше оновлювати, ніж їх настільні аналоги. Щоб оновити веб-додаток, необхідно внести зміни тільки в одному місці: на сервері, на якому обробляється інформація. Оновлення ПО для комп'ютера є набагато більш складним завданням: необхідно розробити, протестувати і, нарешті, поширити оновлення серед усіх кінцевих користувачів. Також суттєво оптимізується процес обслуговування, модернізації інтерфейсу. Всі оновлення активуються автоматично при завантаженні сторінки.

1.2.4. Недоліки розробки веб-додатків

Індивідуальні налаштування браузерів клієнта**.** Різна реалізація *HTML, CSS, DOM* і інших специфікацій в браузерах може викликати проблеми при розробці веб-додатків і подальшої підтримки. Крім того, можливість користувача настроювати багато параметрів браузера (наприклад, розмір шрифту, кольору, відключення підтримки сценаріїв) може перешкоджати коректній роботі додатку. Проте цю проблему можна вирішити використовуючи так звану кросбраузерну верстку.

Кросбраузерна верстка - це верстка сайту з його однаковою видимістю в різних браузерах. При кросбраузерній верстці потрібно враховувати всі особливості кожного з браузерів, так як користувач заходить з того браузера до якого він звик і ми його не можемо змусити перейти на інший софт, тож повинні підлаштовуватися під нього.

Ослаблений захист**.** Через відкритий доступ до керування веб-додатком, кількість хакерських атак на такі програми дуже велика. Системи, подібні вебу, *RSS* і *AJAX*, схожі тим, що особливих перешкод для їх повторного використання не існує. Велика частина корисного програмного забезпечення має відкриті тексти, а якщо і немає, то існує мало способів захистити свою інтелектуальну власність. Стандартна функція оглядача інтернет "переглянути вихідний код" дозволяє будь-якій людині скопіювати будь-яку веб-сторінку.

Необхідність оптимізації**.** В успішних веб-проектах на сайт приходять десятки користувачів на секунду. Тому варто замислитися над оптимізацією швидкості виконання програми. Від моменту натискання на посилання до появи результату часто проходить більше 1 секунди (а буває й більше). Тому людина встигає помітити затримку і навіть занудьгувати. Задача програміста - максимально скоротити час реакції додатка.

Недоступність деяких функцій. На відміну від звичайних (нативних) додатків, веб-додаток не може ефективно використовувати всі функції смартфона. Камери, *GPS*, телефонний набір і інші функції, інтегровані в пристрій, не завжди добре розроблені для адаптивних веб-сайтів. Також існує велика кількість додатків, які не можуть бути реалізовані в браузері.

Підключення до мережі**.** Щоб користуватися веб-додатком та його функціями, пристрій користувача обов’язково повинен бути підключений до інтернету.

1.3. Існуючі аналоги туристичних веб-додатків

На сьогоднішній день існує велика кількість веб-додатків, що можуть проводити онлайн екскурсії. Розглянемо деякі з них та порівняємо.

1) Музеї України просто неба.

Віртуальний тур музеями України під відкритим небом - проект, який пропонує всім бажаючим абсолютно безкоштовно побувати в таких місцях:

- Ужгородський Національний музей народної архітектури та побуту;
- Пирогово - Національний музей народної архітектури та побуту України;
- «Мамаєва слобода»;
- Музей народної архітектури та побуту Середньої Наддніпрянщини;
- «Резиденція Богдана Хмельницького»;
- «Шевченківський гай»;
- "Запорізька Січ".

Інтерфейс веб-додатку зображений на рис. 1.5.

Рис. 1.5. Інтерфейс додатку «Музеї України просто неба»

Веб-додаток має такі позитивні сторони:

- унікальний та водночас простий і зрозумілий інтерфейс;
- аудіосупровід;
- 3*D* огляд музеїв;
- детальна та цікава інформація, яку можна прослухати;
- безкоштовні екскурсії;
- підтримка сайту англійської, української та російської мов.

Попри велику кількість позитивних сторін також є й недолік - можливо відвідати лише 7 музеїв України.

2) *Google Tour Creator.*

Google Tour Creator – веб додаток від компанії *Google* за допомогою якого можна здійснити 3*D* екскурсію по найпопулярнішим місцям планети, таким як: Нью-Йорк, 7 чудес світу, Велика китайська стіна, Париж, Еверест, тощо. А також подивитися як відбуваються такі процеси як вагітність, розглянути анатомію людини, помандрувати сонячною системою, роздивитись роботу мікропроцесорів. Також дуже важливо, що в *Google Tour Creator* є можливість створювати свої онлайн екскурсії, попередньо завантаживши необхідні фото, текстові матеріали та аудіофайли. Інтерфейс додатку зображений на рис. 1.6.

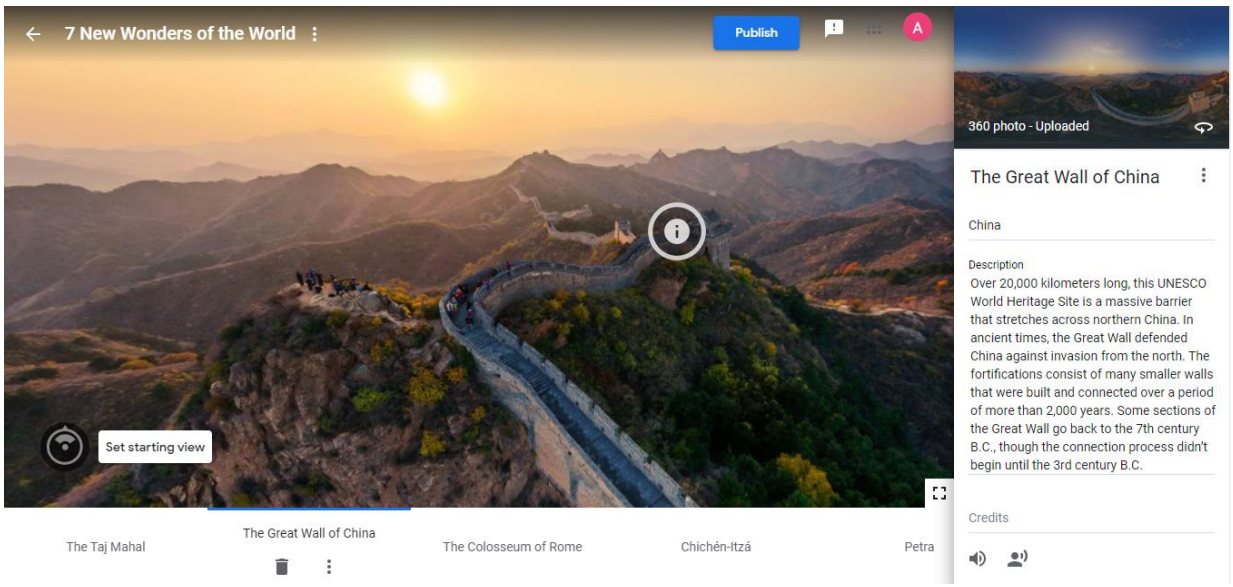

Рис. 1.6. Інтерфейс додатку *Google Tour Creator*

Переваги:

- інтуїтивно-зрозумілий та сучасний інтерфейс;
- можливість створення власної екскурсії;
- 3D огляд;
- наявність як аудіофайлів з інформацією, так і текстової інформації;
- безкоштовні екскурсії.

Недоліки:

- одномовність: інтерфейс та екскурсії лише англійською мовою;
- іноді довго завантажується контент та висне сторінка;

3) *Infoportal.*

Infoportal – веб-додаток, що містить багато інформації про Київ, а також має можливість проводити онлайн екскурсії. До його можливостей входять: перегляд погоди в Києві, новини в Києві, транспорт Києва, афіші, тощо. Інтерфейс додатка представлений на рис. 1.7.

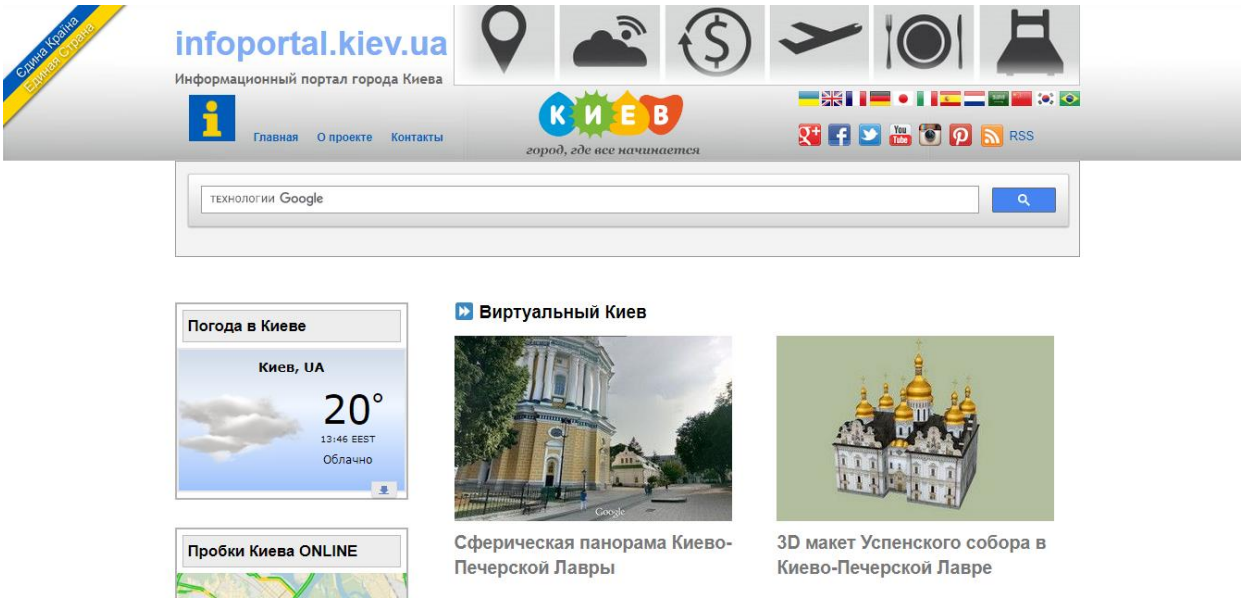

Рис. 1.7. Інтерфейс додатку *Infoportal*

Переваги:

- наявність повної інформації про Київ на одному сайті;
- багатомовність;
- велика кількість безкоштовних екскурсій.

Недоліки:

- застарілий дизайн;
- незручний інтерфейс сайту;
- наявність реклами;
- наявність великої кількості зайвої інформації;
- відсутність можливості прослухати екскурсію в аудіо форматі.

Порівняємо три веб-додатки на наявність чи відсутність деяких критеріїв та приведемо їх в таблиці 1.1.

Таблиця 1.1

Порівняння туристичних веб-додатків

Критерій	**Музеї України просто неба**	***Google Tour Creator***	***Infoportal***
Зручність інтерфейсу	+	+	-
Сучасність та простота використання	+	+	-
Багатомовність	+	-	+
Великій вибір екскурсій	-	+	+
Безкоштовні екскурсії	+	+	+
Аудіофайли	+	+	-
Велика кількість текстової інформації про об’єкти	-	-	+

1.4. Висновки до розділу

На основі проведеного аналізу можна сказати, що на сьогоднішній день розробка веб-сайтів або веб-додатків є однією з основних вимог для створення свого бізнесу, надання послуг, інформування клієнтів, тощо. Майже всі компанії в будь-якій сфері діяльності мають власні веб-сайти/веб-додатки, за допомогою яких їх клієнти можуть з легкістю дізнатися потрібну їм інформацію. В залежності від потреб компанії, вони створюють звичайні веб-сторінки, веб-сайти або веб-додатки. Веб-

додатки мають значну перевагу порівняно з веб-сайтами, оскільки користувач може взаємодіяти з інформацією, яка йому представляється. Такі системи не потребують багато часу на розробку, порівняно зі звичайними додатками, і підтримуються на всіх ОС. Розробка веб-додатків зменшує комерційні витрати, оскільки вони потребують менше технічного обслуговування та мають низькі вимоги до системи кінцевого користувача.

Отже, за результатами проведеного аналізу і для розкриття теми дипломного проєкту треба виконати наступне:

1) описати загальну структуру додатку за допомогою діаграм;
2) розробити алгоритм реєстрації користувача;
3) розробити інтерфейс користувача;
4) розробити інструкції користувача;
5) провести тестування розробленого додатку.

РОЗДІЛ 2
ВЕБ-ДОДАТОК «ТУРИСТИЧНИЙ ГІД»

2.1. Обґрунтування вибору інструментів та технологій розробки

2.1.1. *HTML*

HTML - це код, який використовується для структурування і відображення веб-сторінки та її контенту. *HTML* не є мовою програмування, це мова розмітки, і використовується, щоб повідомляти вашому браузеру, як відображати веб-сторінки, які ви відвідуєте. *HTML* складається з ряду елементів, які ви використовуєте, щоб вкладати або обертати різні частини контенту, щоб змусити контент відображатися або діяти певним чином. Приклад такого елементу зображено рис. 2.1.

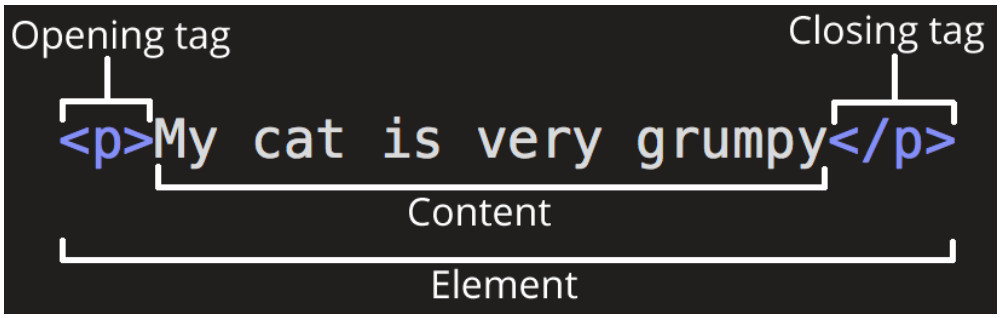

Рис. 2.1. Структура *HTML* елемента

Основними перевагами використання *HTML* є:

1) підтримка всіма браузерами;
2) легко інтегрується з іншими мовами;
3) миттєве відображення результату коду.

2.1.2. *CSS*

CSS - мова опису зовнішнього вигляду *HTML*-документа. Це одна з базових технологій в сучасному інтернеті. Практично жоден сайт не обходиться без *CSS*, тому *HTML* і *CSS* діють в єдиній зв'язці.

Каскадні таблиці стилів працюють з *HTML*, але це зовсім інша мова. *HTML* структурує документ і впорядковує інформацію, а *CSS* взаємодіє з браузером, щоб надати документу оформлення.

Переваги використання *CSS:*

1) незалежність від платформи;
2) прискорення завантаження сторінок і зменшення обсягів інформації;
3) можливість розділити зміст сторінки від її оформлення;

2.1.3. *JavaScript*

JavaScript - це динамічна мова програмування, яка застосовується до *HTML* документу, і може забезпечити динамічну інтерактивність на веб-сайтах. З точки зору веб-розробки, без знань цієї технології неможливо займатися створенням сучасних інтерактивних сайтів. Мова *JS* - це те, що «оживляє» розмітку сторінок (*HTML*) і призначений для користувача функціонал сайтів. За допомогою цієї мови реалізується можливість реакції сторінки або окремих її елементів на дії відвідувача. Сьогодні *JavaScript* є базовою мовою програмування для браузерів. Він повністю сумісний з операційними системами *Windows, Linux, Mac OS*, а також всіма популярними мобільними платформами. Всі призначені для користувача дії у вікні браузера створюють події, а програмування на JS дозволяє обробляти їх певним чином.

Стандартний алгоритм роботи виглядає наступним чином:

1) користувач виконав певну дію;
2) браузер відслідкував поді.;
3) активується *JS*-код;

4) на сторінці відбувається задана зміна.

Завдання програміста полягає в тому, щоб створити обробники для всіх подій, на які повинен реагувати сайт при взаємодії з користувачем.

2.1.4. *PHP*

PHP є однією з мов сценаріїв, таких як *JavaScript*. Різниця між цими мовами полягає в тому, що *PHP* в основному використовується для комунікації на стороні сервера, в той час як JavaScript може використовуватися як для фронтенд, так і бекенд розробки.

PHP дозволяє змінювати веб-сторінку на сервері безпосередньо перед тим, як вона буде відправлена браузеру. *PHP* вміє виконувати код - так звані сценарії. Сценарій - це програма, яка знаходиться на стороні сервера і запускається у відповідь на запит від браузера. В ході виконання *PHP* може змінити або динамічно створити будь-який *HTML*-код, який і є результатом виконання сценарію. Потім сервер відправляє цей код браузеру, що в свою чергу відображає контент.

Мова *PHP* з самого початку створювалась для веб-розробки і вже більше 20 років успішно використовується для розробки веб-сайтів. Основними перевагами розробки на *PHP* є:

1) запускається на більшості платформ;
2) запускається більшості веб-серверах;
3) інтуїтивно-зрозумілий та зручний синтаксис.

2.1.5. *MAMP*

Для роботи з веб-додатком потрібен локальний сервер. *MAMP* – це збірка веб-сервера, що містить *Apache*, *MySQL*, інтерпретатор скриптів *PHP*, графічний інтерфейс для роботи з базою даних *phpMyAdmin* і інші доповнення, призначені для веб-розробки. *MAMP* має зручний та зрозумілий інтерфейс, підтримує версії *PHP*

починаючи з 5.2.17. *MAMP* підтримує більшість серверів, баз даних та мов програмування. Завантажити *MAMP* можна з офіційного сайту безкоштовно.

2.1.6. *Visual Studio Code*

Для роботи з файлами потрібен текстовий редактор. *Visual Studio Code* - безкоштовний і дуже популярний редактор коду від *Microsoft*. З одного боку він підходить новачкам, тому що його інтерфейс інтуїтивно простий і зрозумілий. З іншого боку в *VS Code* вбудовані багато можливостей, які цікаві досвідченим розробникам.

При запуску редактора в перший раз відкривається стартова сторінка (рис. 2.3), на якій описані всі можливості редактора. Редактор підтримує безліч мов програмування і легко налаштовується під користувача. Можна встановити різні поєднання клавіш і колірні схеми. Всі посилання інтерактивні і відкривають потрібну частину інтерфейсу або сторінку в браузері.

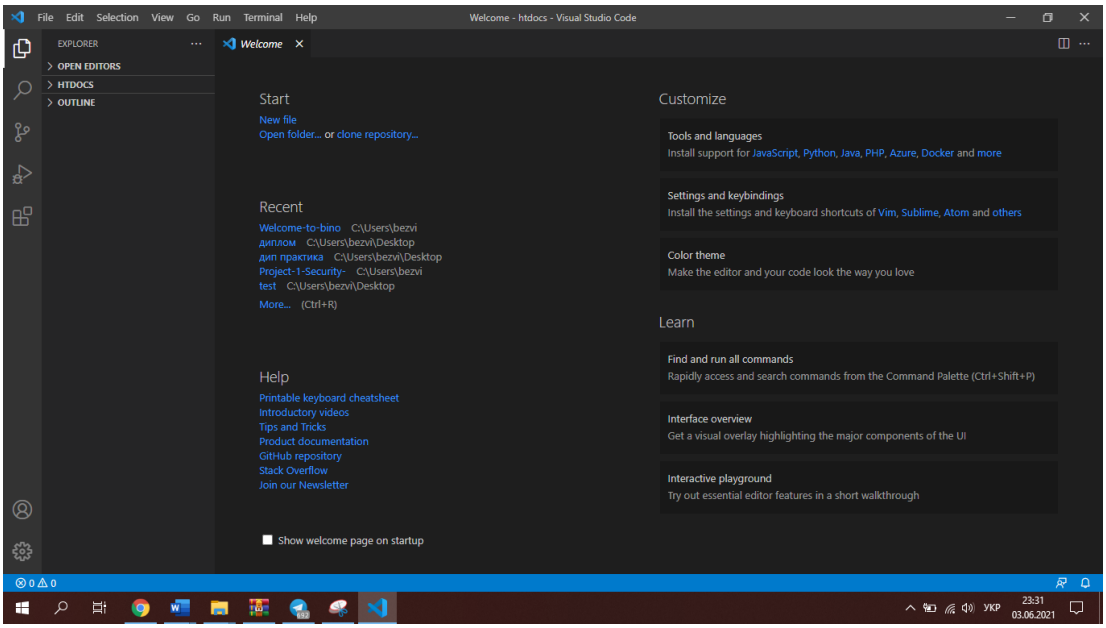

Рис. 2.3. Інтерфейс *Visual Studio Code*

2.1.7. *SQL*

Мова *SQL* є основою багатьох СУБД, тому що відповідає за фізичне структурування й запис даних на диск, а також за читання даних з диска, дозволяє приймати *SQL*-запити від інших компонентів СУБД і користувальницьких додатків. Таким чином, *SQL* – потужний інструмент, що забезпечує користувачам, програмам й обчислювальним системам доступ до інформації, що утримується в реляційних базах даних.

Основними перевагами мови *SQL* є:

1) незалежність від конкретних СУБД;
2) можливість створення інтерактивних запитів;
3) можливість програмного доступу до БД;
4) можливість динамічної зміни і структури БД.

2.2. Діаграма прецедентів

Діаграма прецедентів вважається основним видом діаграми візуального моделювання. Вона описує поведінку системи з погляду на неї користувачем. Діаграму прецедентів ще називають діаграмою варіантів використання або use-case діаграмою. Основні елементи діаграми прецедентів є актор, прецедент і відносини між ними.

Актор – це сутність, яка взаємодіє з системою в процесі своєї роботи з нею. Акторами можуть бути люди, зовнішня система, підсистема, клас. Прецеденти – це певні події, що виконуються системою, і ведуть до спостережуваного актором результату. Прецеденти на діаграмі позначаються еліпсами. На рис. 2.4 зображено діаграму прецедентів веб-додатку «Туристичний гід». На діаграмі зображено три актора, а саме: користувач, зареєстрований користувач та не зареєстрований користувач. Вони зв’язані відношенням узагальнення. Узагальнення – це відношення між предком та нащадком, з’єднаних стрілкою до предка з незафарбованим

трикутником вкінці. В даному випадку предком є користувач, а нащадками – зареєстрований та не зареєстрований користувачі.

Прецедент «Вхід» зв'язаний з прецедентом «Вихід» відношенням включення. Включення означає, що деяка точка базового прецеденту містить поведінку іншого прецедента. Зображується включення залежністю (пунктирна лінія зі стрілкою) зі стереотипом «*include*». При цьому стрілка спрямована у бік включаємого прецедента. Тобто прецедент «Вхід» включає в себе прецедент «Вихід».

Прецеденти «Вірний пароль/логін», «Невірний пароль/логін» зв'язані з прецедентом «Вхід» відношенням розширення. Розширення доповнює прецедент іншими прецедентами, тобто додає у вихідний прецедент послідовність дій, що міститься в іншому прецеденті. Зображується розширення залежністю (пунктирна лінія зі стрілкою) зі стереотипом «*extend*».

Прецедент «Реєстрація» включає в себе прецедент «Вхід», аналогічно прецедентам «Розпочати екскурсію» та «Вибір екскурсії». Прецеденти «Ввід повідомлення» та «Ввід електронної пошти» доповнюють прецедент «Відправка повідомлення».

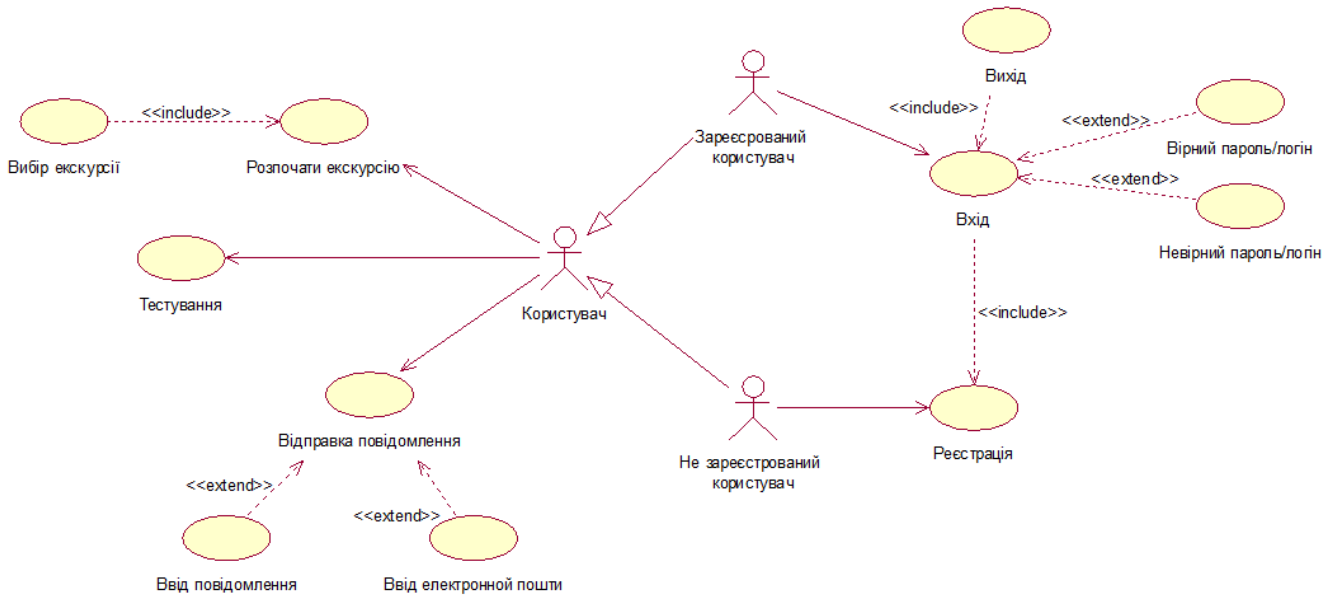

Рис. 2.4. Діаграма прецедентів

2.3. Діаграма класів

Діаграма класів – це набір статичних, декларативних елементів моделі. Елементами діаграмі класів можуть бути інтерфейси, пакети, відносини, об'єкти та зв'язки.

Клас на діаграмі зображується у вигляді прямокутника, розділеного горизонтальними лініями на три секції. Перша секція містить назву класу. Як правило, ім'я класу складається з одного, максимум двох слів. Друга містить перелік атрибутів класу, які характеризують той чи інший об'єкт цього класу в моделі предметної області. Третя містить перелік операцій, що відображають його поведінку в моделі предметної області.

Атрибут – змістовна характеристика класу, що описує множину значень, які можуть приймати окремі об'єкти цього класу.

Операція – це сервіс, що надається кожним екземпляром або об'єктом класу на вимогу своїх клієнтів, в якості яких можуть виступати інші об'єкти, в тому числі і екземпляри даного класу.

Інтерфейс – іменована множина операцій, які характеризують поведінку окремого елемента моделі. Інтерфейс в контексті мови *UML* є спеціальним випадком класу, у якого є операції, але відсутні атрибути.

На рис. 2.5 зображено діаграму класів веб-додатку «Туристичний гід». Клас «Головне вікно» є інтерфейсом, тому він не має атрибутів, а містить лише операції. З інтерфейсом зв’язані класи «Кабінет», «Повідомлення», «Екскурсії» та «Факти». Класи «Реєстрація» та «Вхід» пов’язані з класом «Кабінет» відношенням «один до багатьох».

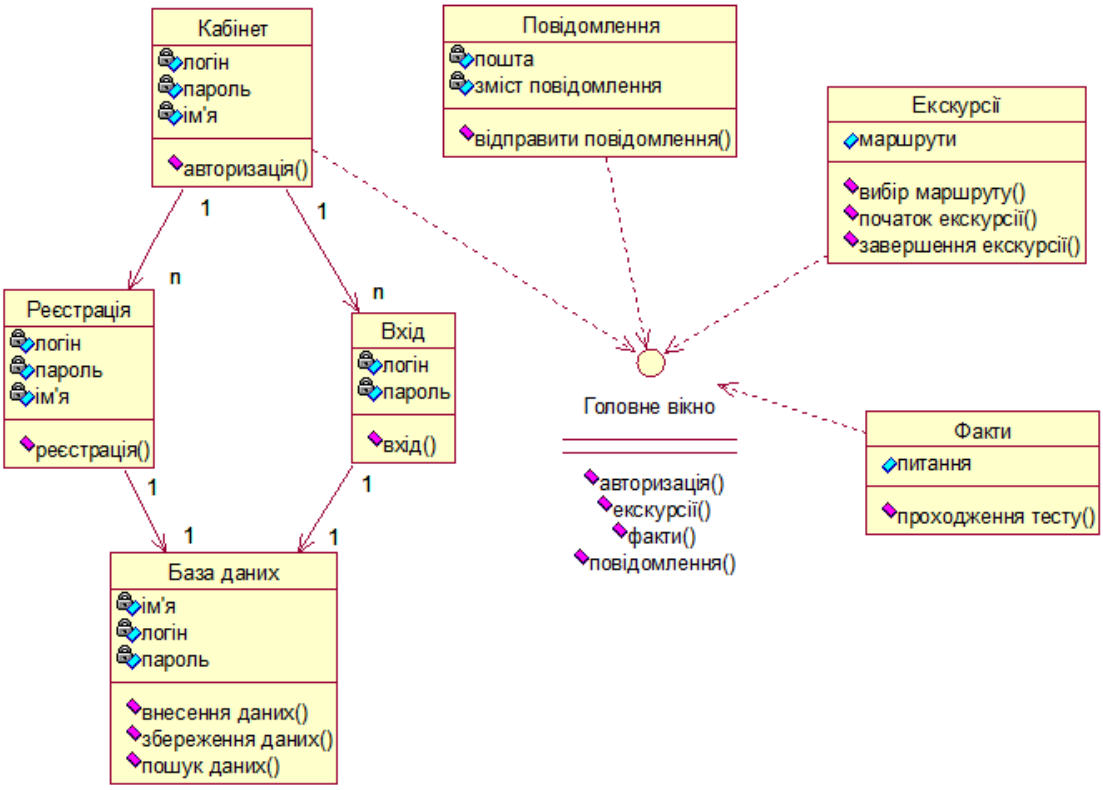

Рис. 2.5. Діаграма класів

2.4. Діаграма кооперації

Діаграма кооперації показує структурну організацію об'єктів, що посилають і отримують повідомлення. На діаграмі кооперації розміщуються об'єкти, що представляють собою екземпляри класів, зв'язки між ними, які в свою чергу є екземплярами асоціацій та повідомлення. Зв'язки доповнюються стрілками повідомлень, при цьому показуються тільки ті об'єкти, які беруть участь у реалізації модельованої кооперації. Далі, як і на діаграмі класів, показуються структурні відносини між об'єктами у вигляді різних з'єднувальних ліній. Зв'язки можуть доповнюватися іменами ролей, які грають об'єкти в даному взаємозв'язку. Зображуються також динамічні взаємозв'язки – потоки повідомлень у формі стрілок з вказівкою напрямку поряд зі сполучними лініями між об'єктами, при цьому задаються імена повідомлень та їх порядкові номери в загальній послідовності повідомлень.

На рис. 2.6 зображено діаграму кооперації для типового випадку використання веб-додатку «Туристичний гід» - проходження онлайн екскурсії. На діаграмі показано послідовність дій користувача та їх взаємодію з класами.

Рис. 2.6. Діаграма кооперації для типового використання веб-додатку «Туристичний гід»

2.5. Алгоритм реєстрації користувача

За допомогою блок-схеми можна відобразити принцип роботи певного алгоритму, що дає повне розуміння процесу який відбувається. На рис. 2.7 зображено схему алгоритму реєстрації користувача. При реєстрації користувача виконуються наступні дії:

1) ввід користувачем логіну, імені та паролю;
2) перевірка даних на правильність вводу;
3) з’єднання з базою даних;
4) внесення даних користувача в базу.

Рис. 2.7. Схема алгоритму реєстрації користувача

На рис. 2.8 показано продовження схеми алгоритму реєстрації користувача.

Рис. 2.8. Продовження схеми алгоритму реєстрації користувача

2.6. Висновки до розділу

В цьому розділі було описано та продемонстровано як можна реалізувати веб-додаток. Використовуючи якісне програмне забезпечення та правильні і зручні технології. Було використано велику кількість технологій, в першу чергу це мова розмітки *HTML*, таблиці стилів *CSS* та об’єктно-орієнтовані мови програмування *JavaScript* та *PHP*.

Використовуючи уніфіковану мову моделювання – *UML*, за допомогою діаграм прецедентів, класів та кооперації було створено загальну структуру та алгоритм роботи веб-додатку. Кожна з діаграм характеризує та описує веб-додаток з різних його сторін та точок зору. Діаграма прецедентів створена для опису додатка з погляду на неї звичайним користувачем. Діаграма класів створена для представлення

статичної моделі додатку, що показує взаємодію його компонентів у вигляді класів та зв’язків між ними. Діаграма кооперацій була створена для відображення типового перебігу подій для варіанта використання додатку, а саме – проходження онлайн екскурсії його користувачем. За допомогою блок-схеми було описано алгоритм реєстрації користувача у веб-додатку.

РОЗДІЛ 3
ПРОГРАМА ВЕБ-ДОДАТКУ

3.1. Розробка інтерфейсу користувача

Інтерфейс веб-додатку – невід'ємна частина та водночас одна з головних, якій слід приділяти максимальну увагу при розробці, оскільки інтерфейс – це те, що бачить та з чим взаємодіє користувач. Інтерфейс веб-додатку розділений на два блоки: статичний і динамічний. Статичний блок інтерфейсу - це модулі, які відображаються на кожній сторінці порталу. Даний блок складається з двох *php* файлів: *header* і *footer*. *Header* - це навігаційна панель із посиланнями на головну сторінку, сторінку з екскурсіями, сторінку з фактами, контактами, особовим кабінетом та логотипом. *Footer* містить текст логотипу та рік розробки додатку. Динамічний блок – це контент, кожної з веб-сторінок. На рис. 3.1 зображено головну сторінку веб-додатка.

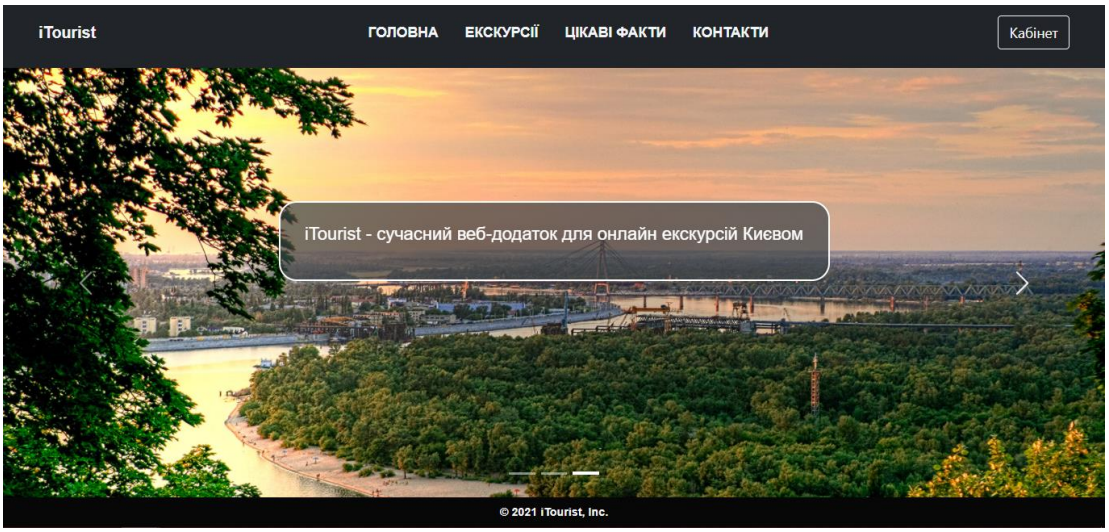

Рис. 3.1. Головна сторінка веб-додатку

Кафедра КСУ				НАУ 21 04 18 000 ПЗ			
Виконав	*Безвіконний О.В.*			Програма веб-додатку	***Літера***	***Аркуш***	***Аркушів***
Керівник	*Масловський Б.Г.*				*Д*	*34*	*46*
Консульт.					СП-435 123		
Норм. контр.	*Тупота Є. В.*						
Зав. Каф.	*Литвиненко О.Є.*						

Важливим фактором при створенні веб-додатку є його адаптивність – здатність підлаштовувати контент для різних розрішень екранів. Адже клієнт може користуватися додатком як в смартфоні з маленьким екраном, так і на ноутбуці з екраном набагато більшим, порівняно з телефоном. Через це можуть виникати проблеми з відображенням контенту, наприклад: поява горизонтальної прокрутки, нечитабельність тексту, зліплення елементів сайту, тощо. Отже, відштовхуючись від цього потрібно підлаштовувати контент додатку для різних пристроїв. Адаптивність реалізовується за допомогою таблиці каскадних стилів. Використовуючи медіа запит, створимо адаптивне меню. При розрішенні екрану менше 768 пікселів, створимо медіа запит, в якому задаємо стилі деяким класам меню (рис. 3.2). За допомогою псевдоелементів *before* та *after* створюємо іконку для відображення меню. При закритому меню іконка складається з трьох паралельних горизонтальних ліній. При відкритому меню іконка трансформується в хрестик. Для роботи меню (відкривання, закривання, перехід за посиланнями) необхідно застосувати скрипт. На рис. 3.3 зображено скрипт, який відслідковує подію натискання на іконку меню і додає до нього додатковий клас «*active*» в якому прописані основні стилі для оформлення адаптивного меню.

```
@media screen and (max-width: 767px) {
    nav{
        align-items: center;
    }
    .menu{
        flex-direction: column;
    }
        .menu-burger{
        display: block;
        z-index: 4;
        position: relative;
        width: 30px;
        height: 18px;
        cursor: pointer;
    }
    .menu-burger::before, .menu-burger::after, .menu-burger>span{
        left: 0;
        position: absolute;
        height: 2px;
        width: 100%;
        transition: all 0.3s ease 0s;
        background-color: rgb(255, 255, 255);
    }
}
```

Рис. 3.2. Приклад використання медіа запиту

```
const menuIcon = document.querySelector('.menu-burger');
const menuHeader = document.querySelector('.menu');

if(menuIcon){
    menuIcon.addEventListener('click', function (e){
        document.body.classList.toggle('lock');
        menuIcon.classList.toggle('active');
        menuHeader.classList.toggle('active');
    });
};
```

Рис. 3.3. *JavaScript* код для роботи адаптивного меню

В результаті при розрішенні екрану менше 768 пікселів буде з’являтися іконка меню (рис. 3.4), при натисканні на яку воно буде відкриватися (рис. 3.5).

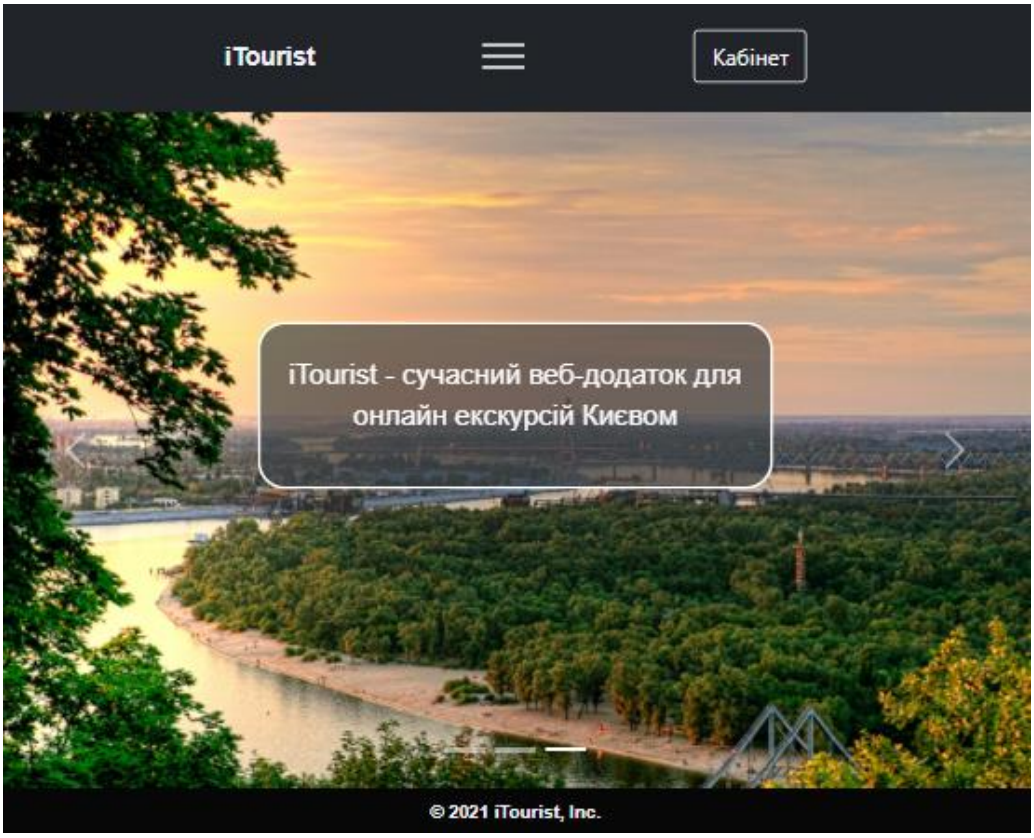

Рис. 3.4. Відображення іконки скритого меню

Рис. 3.5. – Відображення меню при натисканні на іконку

3.2. Підключення бази даних

За допомогою графічного інтерфейсу для роботи з базами даних *phpMyAdmin* на сервері була створена база даних *registration-db*. Для подальшої роботи з базою було створено таблицю *users* (рис. 3.6), в яку будуть заноситься дані про користувачів. В нашому випадку – це *id*, *login*, *pass*, *name*. Кожен зареєстрований користувач матиме свій унікальний номер – *id*.

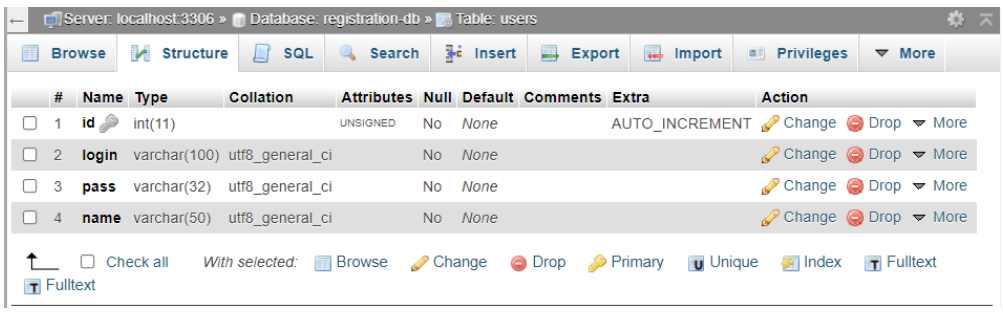

Рис. 3.6. Таблиця *users*

Для обробки даних користувача необхідно створити файл, що буде містити *php*-код (рис. 3.7). Назвемо файл *check.php*. Скрипт приймає дані, які ввів користувач та записує їх в змінні *login*, *name* та *pass*. Після чого дані перевіряються на коректність вводу, якщо перевірку пройдено, то можна підключатись до бази даних використовуючи функцію «*new mysqli*» в параметрах якої вказати адресу, ім'я користувача, пароль, та назву таблиці. Використовуючи *SQL*-запит вносимо дані в таблицю, після чого закриваємо запит до бази та переходимо на головну сторінку.

check.php

validation-form > check.php

```php
<?php
    $login = filter_var(trim ($_POST['login']), FILTER_SANITIZE_STRING);
    $name = filter_var(trim ($_POST['name']), FILTER_SANITIZE_STRING);
    $pass = filter_var(trim ($_POST['pass']), FILTER_SANITIZE_STRING);

    if(mb_strlen($login) < 3 || mb_strlen($login) > 90){
     echo "Довжина логіна повинна бути від 3 до 90 символів";
     exit();
    }
    else if(mb_strlen($name) < 1 || mb_strlen($name) > 50){
     echo "Довжина імені повинна бути від 1 до 50 символів";
     exit();
    }
    else if(mb_strlen($pass) < 6 || mb_strlen($pass) > 50){
     echo "Довжина пароля повинна бути від 6 до 50 символів";
     exit();
    }

    //хешування для пароля
    $pass = md5($pass."oiuyt557rdcv3bnm");
    //Підключення до бази даних mysql
    $mysql = new mysqli('localhost', 'root', 'root', 'registration-db');
    $mysql->query("INSERT INTO users (login, pass, name) VALUES('$login', '$pass', '$name')");
    $mysql->close();
    header('Location: /');
?>
```

Рис. 3.7. *php*-код файлу *check.php*

3.3. Інструкції користувача

Додаток працює в веб-браузері, тому для його користування необхідний доступ до інтернету. Додаток працює з усіма браузерами та операційними системами. Проте для користування максимальним функціоналом в таблицях 3.2 та 3.3 представлені мінімальні вимоги до систем.

Таблиця 3.1

Вимоги до браузерів

Браузер	**Версія**
Google Chrome	11 і новіша
Internet Explorer	11 і новіша
Firefox	5 і новіша
Safari	5 і новіша
Opera	11 і новіша

Таблиця 3.2

Вимоги до ОС

ОС	**Версія**
Windows	7, 8, 10
Mac OS	10.12.0 або новіша версія
Linux	Сумісний
Android	5.0 або новіша версія
IOS	11 або новіша версія

Покрокова інструкція користування додатком:

1) відкрити веб-додаток на комп’ютері, смартфоні, планшеті, ноутбуці, використовуючи веб-браузер;

2) зайти в особовий кабінет та зареєструватись або здійснити вхід, якщо клієнт вже зареєстрований;

3) перейти на головну сторінку та ознайомитись з інтерфейсом та меню додатку;

4) обрати пункт меню «Екскурсії»;

5) ознайомитись з запропонованими маршрутами екскурсій та обрати один за бажанням;

6) натиснути кнопку «Розпочати»;

7) ознайомитись з точкою проходження маршруту та короткою інформацією про пам’ятку;

8) ознайомитись з точкою проходження маршруту на карті (рис. 3.8);

9) знайти на карті жовтого чоловічка (рис. 3.9) та утримуючи ЛКМ (або затиснувши його на смартфоні) перемістити його до потрібної точки маршруту для відкриття 3*D* тура;

10) включити аудіофайл для прослуховування екскурсії;

11) відкрити карту на весь екран, переміщатись по карті, прослуховуючи екскурсію;

12) повторити пункти 7-11 для всіх точок маршруту;

13) після проходження екскурсії повернутись на головну сторінку;

14) обрати пункт меню «Цікаві факти»;

15) пройти тест на знання Києва та дізнатись свій результат;

16) завершити роботу.

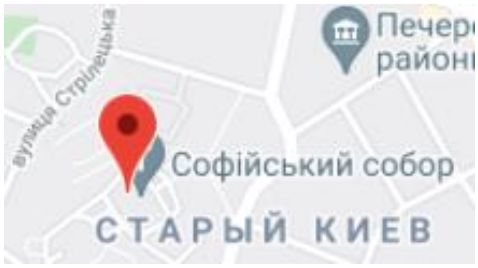

Рис. 3.8. Точка маршруту на карті

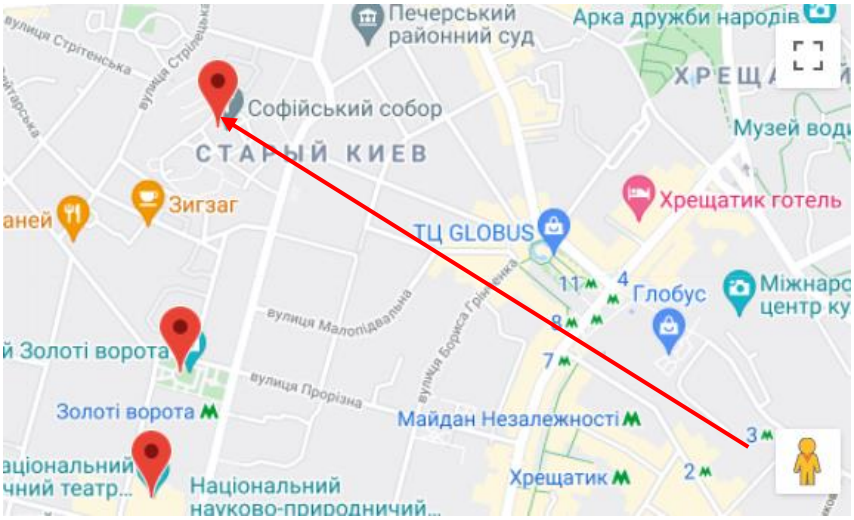

Рис. 3.9. Відкриття 3*D* тура

3.4. Тестування додатку

Проведемо функціональне тестування розробленого адаптивного меню додатка. В таблиці 3.3 представлений набір тестів, вхідні дані, очікуваний результат та результат проходження або не проходження тесту.

Таблиця 3.3

Тестування адаптивного меню

№	Назва тесту	Вхідні дані	Очікуваний результат	Тест пройдено?
1	Відображення головного екрану	Розрішення екрану пристрою 768*px* і ширше	Повноцінне відображення меню	Так
2	Відображення головного екрану	Розрішення екрану пристроя менше 768*px*	Адаптивне відображення меню	Так

Продовження таблиці 3.3

3	Відображення головного екрану	Натискання на кнопку «бургер» при розрішенні екрана менше 768*px*	Відображення меню	Так
4	Відображення головного екрану	Натискання на кнопку «хрестик» при розрішенні екрана менше 768*px*	Вікно головного меню повинно закритися	Так
5	Відображення головного екрану	Перехід по посиланням меню при розрішенні екрана менше 768*px*	Повинен відбутися перехід до розділу із меню	Так
6	Відображення головного екрану	Прокрутка контенту при відкритому меню при розрішенні екрана менше 768*px*	Прокрутка не повинна відбуватись	Так
7	Відображення головного екрану	Зміна іконки меню при відкритті при розрішенні екрана менше 768*px*	Іконка «бургер» повинна змінитись на іконку «хрестик»	Так

Закінчення таблиці 3.3

8	Відображення головного екрану	Зміна іконки меню при закритті при розрішенні екрана менше 768*px*	Іконка «хрестик» повинна змінитись на іконку «бургер»	Так
9	Відображення головного екрану	Натискання на будь-який пункт меню	Плавний перехід до відповідного розділу	Так
10	Відображення головного екрану	Натискання на логотип	Логотип повинен бути посиланням на головну сторінку	Так

3.4. Висновки до розділу

В цьому розділі був описаний процес розробки інтерфейсу, а саме було показано як реалізовувати адаптивність веб-додатку, адже адаптивність – одна з ключових речей при роботі з додатком користувача. Також було продемонстровано підключення бази даних для реєстрації нових користувачів та входу вже існуючих.

В додатку створено простий та водночас зручний дизайн. Інтерфейс інтуїтивно-зрозумілий, тому користуватися додатком зможе навіть дитина. Також було протестовано роботу адаптивного меню веб-додатку та створено інструкції користувача.

ВИСНОВКИ

В ході виконання дипломного проєкту були описані особливості створення веб-додатків та веб-сайтів, їх відмінності. Аналіз показав, що створення веб-додатків набагато складніший та займає більше часу на створення порівняно з веб-сайтами. Проте функціонал веб-додатків значно перевищує функціонал сайтів. Також був проведений огляд типових туристичних веб-додатків та порівняння їх можливостей. В результаті було поставлено задачу на виконання та розкриття теми дипломного проєкту.

Було проведено вибір інструментів та технологій розробки, обґрунтування їх використання. Розумний вибір інструментів спрощує та полегшує процес розробки, що є дуже важливим фактором для розробника. Створенні діаграми прецедентів, класів та кооперації показали як буде виглядати додаток з точки зору користувача; показали загальну структуру додатку; продемонстрували типовий варіант використання додатку. Також був описаний процес створення інтерфейсу додатку та його підключення до бази даних. Розроблені інструкції користувача.

В результаті написання дипломної роботи була виконана програмна реалізація клієнтської та серверної частин веб додатку «Туристичний гід», основна задача якого полягає в наданні екскурсійних послуг онлайн. Функціональна структура додатку складається з веб-сайту та бази даних, дані елементи взаємодіють між собою.

Позитивними якостями даного веб-додатку є:

- Зручний та інтуїтивно-зрозумілий інтерфейс;
- Можливість проходити екскурсії, використовуючи 3*D* карту;
- Можливість користування додатком на різних пристроях (ПК, ноутбук, смартфон, планшет);
- Можливість користування додатком на різних операційних системах (*MS Windows, Mac OS, Linux, Android, IOS*).

Результатами проєкту зможуть користуватись туристи та подорожуючі, жителі Києва, які зацікавлені в дослідженні свого міста, педагоги, що організовують

просвітницьку та виховну роботу з дітьми, молоддю, туристичні фірми, підприємства туристичної інфраструктури, міська рада.

В подальшому проєкт буде продовжувати розвиватись. Будуть додаватись нові маршрути екскурсій, покращуватись та розширюватись функціонал.

СПИСОК БІБЛІОГРАФІЧНИХ ПОСИЛАНЬ ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Бойченко С.В., Іванченко О.В. Положення про дипломні роботи (проекти) випускників Національного авіаційного університету. – К.: НАУ, 2017. – 63 с.
2. ДСТУ ГОСТ 3008-95 «Документація. Звіти у сфері науки і техніки. Структура і правила оформлення».
3. ДСТУ 3008-95. Документація. Звіти у сфері науки і техніки. правила оформлення. – 88 с.
4. Криза у світовому туризмі [Електронний ресурс]. – *URL: https://www.slovoidilo.ua/2021/01/28/novyna/suspilstvo/kryza-svitovomu-turyzmi-vidstupyt-ne-ranishe-2023-roku-oon*
5. Особливості веб-додатків [Електронний ресурс]. – *URL: http://sites.znu.edu.ua/webprog/lect/1191.ukr.html*
6. Різниця між веб-сайтом та веб-додатком [Електронний ресурс]. - *URL: https://dinarys.com/ru/blog/website-vs.-webapplication*
7. Основи *HTML* [Електронний ресурс]. - *URL: https://developer.mozilla.org/ru/docs/Learn/Getting_started_with_the_web/HTML_basics*
8. Поняття *CSS* [Електронний ресурс]. - *URL:* https://skillbox.ru/media/code/chto_takoe_css/
9. Основи *JavaScript* [Електронний ресурс]. - *URL: https://developer.mozilla.org/ru/docs/Learn/Getting_started_with_the_web/JavaScript_basics*
10. Поняття *PHP* [Електронний ресурс]. - *URL: https://htmlacademy.ru/tutorial/php/basics*
11. *MAMP* [Електронний ресурс]. - *URL: https://www.mamp.info/en/windows/*
12. *Visual Studio Code* [Електронний ресурс]. - *URL: https://htmlacademy.ru/blog/boost/tools/vscode*
13. Переваги SQL [Електронний ресурс]. - *URL: https://life-prog.ru/2_90300_perevagi-movi-SQL.html*

ДОДАТОК А

Частини коду програми створення тесту

```
const headElem = document.getElementById("head");
const buttonsElem = document.getElementById("buttons");
const pagesElem = document.getElementById("pages");

//Клас тесту
class Quiz
{
   constructor(type, questions, results)
   {
      this.type = type;

      //Масив з питаннями
      this.questions = questions;

      //Масив з ймовірними результатами
      this.results = results;

      //Кількість набраних балів
      this.score = 0;

      //Номер результату з масива
      this.result = 0;

      //Номер поточного питання
      this.current = 0;
   }
```

```
Click(index)
{
    //Додаємо бали
    let value = this.questions[this.current].Click(index);
    this.score += value;

    let correct = -1;

    //Якщо було додано бал то відповідь вірна
    if(value >= 1)
    {
        correct = index;
    }
    else
    {
        //Якщо ні - шукаємо вірну відповідь
        for(let i = 0; i < this.questions[this.current].answers.length; i++)
        {
            if(this.questions[this.current].answers[i].value >= 1)
            {
                correct = i;
                break;
            }
        }
    }

    this.Next();

    return correct;
```

```
    }

    //Перехід до наступного питання
    Next()
    {
        this.current++;

        if(this.current >= this.questions.length)
        {
            this.End();
        }
    }

    //Перевірка результату користувача
    End()
    {
        for(let i = 0; i < this.results.length; i++)
        {
            if(this.results[i].Check(this.score))
            {
                this.result = i;
            }
        }
    }
}

//Клас питання
class Question
{
    constructor(text, answers)
```

```
    {
        this.text = text;
        this.answers = answers;
    }

    Click(index)
    {
        return this.answers[index].value;
    }
}

//Класс відповіді
class Answer
{
    constructor(text, value)
    {
        this.text = text;
        this.value = value;
    }
}

//Класс результату
class Result
{
    constructor(text, value)
    {
        this.text = text;
        this.value = value;
    }
```

```
    //Перевірка чи достатньо набрано балів
    Check(value)
    {
        if(this.value <= value)
        {
            return true;
        }
        else
        {
            return false;
        }
    }
}

//Масив з результатами
const results =
[
    new Result("Ти взагалі щось чув про Київ?", 0),
    new Result("Бігом на екскурсії!", 2),
    new Result("Непогано!", 4),
    new Result("Все вірно, ти що підглядав?", 6)
];

//Масив з питаннями
const questions =
[
    new Question("У якому році в Києві сталася епохальна подія: відкрили перший МакДональдс?",
    [
        new Answer("1995", 0),
```

```
        new Answer("1996", 0),
        new Answer("1997", 1),
        new Answer("1998", 0)
    ]),

    new Question("Якого кольору спочатку був найстаріший корпус Київського націон
ального університету?",
    [
        new Answer("Білого", 1),
        new Answer("Червоного", 0),
        new Answer("Жовтого", 1),
        new Answer("Блакитного", 0)
    ]),

    new Question("Чи правда, що мер Києва - Віталій Кличко написав іронічну книгу з
і своїми цитатами?",
    [
        new Answer("Не вірю", 0),
        new Answer("Правда", 1),
    ]),

    new Question("Знайди цитату Кличка.",
    [
        new Answer("Ви, хлопці, засранці", 0),
        new Answer("Маємо те, що маємо", 0),
        new Answer("У мене два заступники, чотири з яких вже місяць лежать в кабміні
", 1),
        new Answer("Новорічна... йолка", 0)
    ]),
```

```
    new Question("Якої з цих вулиць немає у Києві",
    [
        new Answer("Ананасової", 0),
        new Answer("Огіркової", 0),
        new Answer("Абрикосової", 0),
        new Answer("Полуничної", 1)
    ]),

    new Question("Найстаріша вулиця Києва, вік якої сягає більше 1000 років - це",
    [
        new Answer("Володимирська", 1),
        new Answer("Хрещатик", 0),
        new Answer("Нижній вал", 0),
        new Answer("Васильківська", 0)
    ])
];

//Сам тест
const quiz = new Quiz(1, questions, results);

Update();

//Оновлення тесту
function Update()
{
    //Перевіряємо чи залишились питання
    if(quiz.current < quiz.questions.length)
    {
        //Якщо є, змінюємо заголовок з питанням
        headElem.innerHTML = quiz.questions[quiz.current].text;
```

```
        //Видалення старих варантів відповідей
        buttonsElem.innerHTML = "";

        //Створення кнопок для нових варіантів відповідей
        for(let i = 0; i < quiz.questions[quiz.current].answers.length; i++)
        {
            let btn = document.createElement("button");
            btn.className = "button";

            btn.innerHTML = quiz.questions[quiz.current].answers[i].text;

            btn.setAttribute("index", i);

            buttonsElem.appendChild(btn);
        }

        //Вивід номеру питання
        pagesElem.innerHTML = (quiz.current + 1) + " / " + quiz.questions.length;

        Init();
    }
    else
    {
        //Якщо кінець тесту, то виводим результат
        buttonsElem.innerHTML = "";
        headElem.innerHTML = quiz.results[quiz.result].text;
        pagesElem.innerHTML = "Очки: " + quiz.score;
    }
}
```

```
function Init()
{
    //Знаходимо всі кнопки
    let btns = document.getElementsByClassName("button");

    for(let i = 0; i < btns.length; i++)
    {
        //Вішаєм подію на кожну кнопку
        //При натисканні на кнопку визивається функція Click
        btns[i].addEventListener("click", function (e) { Click(e.target.getAttribute("index"));
});
    }
}

function Click(index)
{
    //Отримуємо номер правильної відповіді
    let correct = quiz.Click(index);

    let btns = document.getElementsByClassName("button");

    //Робимо кнопки сірими
    for(let i = 0; i < btns.length; i++)
    {
        btns[i].className = "button button_passive";
    }

    if(quiz.type == 1)
    {
```

```
        if(correct >= 0)
        {
            btns[correct].className = "button button_correct";
        }

        if(index != correct)
        {
            btns[index].className = "button button_wrong";
        }
    }
    else
    {
        btns[index].className = "button button_correct";
    }

    setTimeout(Update, 1000);
}
```