

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ДЕРЖАВНИЙ ЗАКЛАД
„ЛУГАНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА”

Навчально-науковий інститут
математики та інформаційних технологій

Ліцкевич Микита Володимирович

РОЗРОБКА СИСТЕМИ БРОНЮВАННЯ КВИТКІВ

кваліфікаційна робота

**здобувача вищої освіти першого (бакалаврського) рівня
освітньої програми « Комп’ютерні науки та інформаційні технології »
за спеціальністю 122 Комп’ютерні науки**

Особистий підпис  Микита ЛІЦКЕВИЧ

Науковий керівник  Владислав КОЗУБ, доктор філософії

Завідувач кафедри _____ Микола СЕМЕНОВ, кандидат
педагогічних наук, доцент

Лубни – 2026

Міністерство освіти і науки України
Державний заклад „Луганський національний університет
імені Тараса Шевченка”
Інститут математики та інформаційних технологій

Кафедра
Освітні рівень
Напрямок підготовки
(спеціальність)
Галузь знань

Інформаційних технологій та систем
бакалавр
122 “Комп’ютерні науки”
12 “Інформаційні технології”

ЗАТВЕРДЖУЮ
Завідувач кафедри ІТС

Микола СЕМЕНОВ

“ ” 2026 р.

**ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ**

Ліцкевич Микита Володимирович
(прізвище, ім’я, по батькові)

1. Тема Розробка системи бронювання квитків
Керівник кваліфікаційної роботи Козуб В.Ю. доктор філософії, доцент
затверджена наказом по від “ ” 2026 року №
університету

2. Строк подання здобувачем вищої освіти проєкту ____

3. Вихідні дані до роботи(проєкту)

Об’єкт розробки: вебсистема онлайн-бронювання квитків на екскурсії (музеї, зоопарки, замки м. Познань).

Технологічний стек: HTML, CSS, JavaScript, PHP, СУБД MySQL. Prototyping — Figma.

Основні вимоги: адаптивний інтерфейс, покрокове оформлення замовлення, динамічне відображення вільних місць, генерація квитків у форматі PDF.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

Аналіз вимог до програмного забезпечення. Аналіз та проєктування програмного забезпечення Проєктування інфологічної та даталогічної моделей бази даних, оптимізація структури збереження інформації про заклади та замовлення. Аналіз зв’язків між таблицями реляційної бази даних, забезпечення транзакційної цілісності та консистентності даних. Програмна реалізація вебсистеми бронювання квитків (HTML, CSS, JavaScript, PHP, MySQL), інтеграція модуля генерації електронних квитків у форматі PDF, налаштування кросплатформного фронтенду.

5. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Розділ 1			
Розділ 2			
Розділ 3			

6. Дата видачі завдання ____ " ____ 2026 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
.	Вибір теми роботи, вивчення наукової літератури, затвердження теми та керівника.	до 1 лютого	
.	Аналіз літературних джерел за темою роботи. Розробка та апробація методики дослідно-експериментальної роботи. Подання структури теоретичної частини роботи та плану експериментальних досліджень.	до 20 березня	
.	Робота над теоретичною частиною. Подання теоретичної частини роботи для першого читання науковим керівником.	до 1 квітня	
.	Усунення зауважень, урахування рекомендацій наукового керівника. Подання теоретичної частини роботи на друге читання.	до 15 квітня	
.	Проведення експериментальної роботи. Поетапний аналіз та обговорення її результатів. Перевірка стану виконання роботи.	до 30 квітня	
.	Урахування рекомендацій наукового керівника, усунення недоліків, підготовка варіанта роботи до передзахисту. Розробка презентації.	до 15 травня	
.	Попередній захист роботи на кафедрі	до 30 травня	
.	Доопрацювання роботи з урахуванням рекомендацій після передзахисту. Подання роботи науковому керівникові та рецензентові на підготовку відгуку та рецензії	за 10 днів до державної атестації	
.	Подання на кафедру остаточного варіанта роботи, переплетеного та підписаного автором, науковим керівником і рецензентом.	за 5 днів до державної атестації	

Здобувач вищої освіти



Микита Ліцкевич

Керівник проєкту (роботи)



Владислав КОЗУБ

АНОТАЦІЯ

Ліцкевич М.В.

Тема: Розробка системи бронювання квитків

Спеціальність: 122 „Комп’ютерні науки”

Установа: ДЗ ЛНУ імені Т.Шевченка, 2026 р.

Кваліфікаційна робота містить: 53 с., 16 рис., 5 табл., 10 додат., 16 джерел.

Об’єкт дослідження - процес онлайн-бронювання квитків на культурно-розважальні об’єкти.

Предмет дослідження - методи, засоби та технології веброзробки бронювання квитків для культурно-розважальної системи.

Мета роботи - розробка сайту для бронювання квитків на екскурсії музеями, зоопарками та замками міста Познані в Польщі.

Розроблено вебсистему бронювання квитків, яка забезпечує перегляд інформації про культурно-розважальні заклади, вибір дати та часу відвідування, оформлення бронювання та генерацію електронного квитка у форматі PDF. Впроваджена система дозволяє осучаснити процес резервування квитків, спростити взаємодію споживача із закладами та забезпечити зручний доступ до нагальної інформації через вебінтерфейс.

Створено структуру бази даних для збереження інформації про заклади, типи квитків, екскурсійні групи та бронювання користувачів. Також реалізовано адаптивний інтерфейс, який забезпечує правильну роботу системи на персональних комп’ютерах і мобільних девайсах.

Розроблена система може використовуватися як основа для подальшого розвитку повноцінного сервісу резервування квитків із підключенням платіжних систем, адмініструванням закладів та автоматизованою обробкою замовлень.

Ключові слова. ВЕБСИСТЕМА, БРОНЮВАННЯ КВИТКІВ, HTML, CSS, JAVASCRIPT, PHP, MYSQL, БАЗА ДАНИХ, ЕЛЕКТРОННИЙ КВИТОК, ВЕБРОЗРОБКА, АВТОМАТИЗАЦІЯ.

ABSTRACT

Litskevych M.

Theme: Development of a Ticket Booking System

Specialty: 122 "Computer Science"

Object of study: the process of online ticket booking for cultural and entertainment venues.

Subject of study: methods, tools, and web development technologies for ticket booking systems in the cultural and entertainment sector.

Purpose of the work: the development of a website for booking tickets to tours of museums, zoos, and castles in the city of Poznań, Poland.

A web-based ticket reservation system has been developed that allows viewing information about cultural and entertainment venues, selecting a date and time for a visit, making a reservation, and generating an electronic ticket in PDF format. The implemented system allows you to streamline the ticket reservation process, simplify customer interaction with venues, and provide convenient access to urgent information via a web interface.

A database structure has been created to store information about establishments, ticket types, tour groups, and user reservations. An adaptive interface has also been implemented to ensure the system works properly on personal computers and mobile devices.

The developed system can be used as a basis for further development of a full-fledged ticket reservation service with the connection of payment systems, establishment administration and automated order processing.

Keywords: WEB SYSTEM, TICKET BOOKING, HTML, CSS, JAVASCRIPT, PHP, MYSQL, DATABASE, ELECTRONIC TICKET, WEB DEVELOPMENT, AUTOMATION.

ЗМІСТ

ВСТУП	7
РОЗДІЛ 1. АНАЛІЗ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ	8
1.1. Еволюція наукової думки та аналіз підходів до автоматизації систем бронювання	8
1.2. Аналіз відомих технічних рішень та технологічних процесів	10
1.3. Аналіз існуючих програмних продуктів та аналогів системи.	14
РОЗДІЛ 2. АНАЛІЗ ТА ПРОЄКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ	17
2.1. Специфікація вимог до програмного забезпечення та визначення пріоритетів розробки	17
2.2. Вибір та аналіз програмного забезпечення для розробки вебсайту	19
2.3. Проєктування користувацького інтерфейсу вебсайту	23
РОЗДІЛ 3. СТВОРЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ	39
3.1. Розробка бази даних	39
3.2. Аналіз зв'язків між таблицями	45
3.3. Розробка сайту	46
ВИСНОВОК	51
СПИСОК ДЖЕРЕЛ	53

ВСТУП

Останнім часом інформаційні технології роблять великі стрибки. Для виконання простих життєвих завдань які люди самі собі ставлять. Вже як буденність використовується інтернет: купівля продуктів, побутової хімії, канцелярії для школярів, одєжі та взуття, бронювання або купівля квитків на поїзд, літак, кіно, автобусний рейс. Достатньо для цього мати телефон та доступ в глобальну мережу або додаток того магазину чи закладу який тобі потрібен за ситуацією.

Більшість послуг переходить у цифровий вигляд, що дає змогу користувачам швидко та зрозуміло отримувати потрібну інформацію, робити придбання та резервування через мережу Інтернет. Особливо важливим є впровадження вебсистем для бронювання квитків на культурно-розважальні події та туристичні локації.

Музеї, замки та зоопарки міста Познань щодня відвідує значна кількість мандрівників та місцевих жителів. Звичні методи купівлі квитків нерідко спричиняють клопоти для українців за кордоном, які пов'язані з володінням мовою, чергами, обмеженим режимом праці кас та труднощами здобуття свіжої довідки про ціну й наявність квитків. Саме тому розробка сучасної системи інтернет-резервування є нагальним завданням.

Напрямом кваліфікаційної роботи є створення вебсистеми резервування перепусток для музеїв, замків та зоопарків міста Познань, яка гарантує користувачам спроможність перегляду відомостей про доступні заклади, види квитків, здійснення резервування та одержання їх електронного варіанту.

Практична цінність кваліфікаційної роботи полягає у створенні зручної та сучасної вебсистеми, яку можна використати для спрощення процесу продажу та резервування квитків для туристично-культурних установ.

РОЗДІЛ 1. АНАЛІЗ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

1.1. Еволюція наукової думки та аналіз підходів до автоматизації систем бронювання

Розвиток наукової думки у сфері автоматизації процесів бронювання пройшов тривалий шлях від найпростіших локальних баз даних до складних розподілених інтелектуальних систем. Одна з перших в Україні систем онлайн бронювання була в «Укрзалізниці». У 2008 році вона ввела систему електронного бронювання за 45 днів до потягу, з можливістю викупу квитка у будь-якій касі «Укрзалізниці» протягом двох днів від дня бронювання.[1] Натомість в Польщі перша згадка про бронювання електронних квитків була у 2001 році, і як це не дивно також пов'язано з залізницею, а саме «РКР Intercity». За інформацією, що розміщена на офіційному сайті, першого вересня 2001 року ця компанія відділилася від групи компаній «РКР», що дало їй можливість розвитку самостійно, хоча вона все ж таки залишилася дочірньою компанією, і саме в 2001 році вони офіційно відкрили можливість лише онлайн бронювання, в випадку як і з «УЗ» після бронювання сам квиток потрібно йти та викуповувати на будь-якій касі «РКР Intercity». У 2002 році вони відкрили кол-центр, що дало змогу дзвонити та бронювати квиток не лише на сайті, що на той момент дало можливість трохи зменшити навантаженість на і так не ідеальну систему. [2]

На початкових етапах розробники були зосереджені на самому факті переходу від паперового обліку до цифрового збереження інформації, де основна увага приділялася надійному зберіганню даних на централізованих серверах. 2003 рік, в «РКР Intercity» з'являється сайт з інформацією про тарифи, сполучення та акції, а вже в 2005 році запускається онлайн продаж квитків. [2]

Проте перші програмні рішення були занадто громіздкими та не дозволяли забезпечувати операційну гнучкість у реальному часі. Роботи попередників у цей період переважно базувалися на архітектурі «клієнт-

сервер», яка хоч і вирішувала питання централізації, проте створювалася значна затримка при одночасному зверненні великої кількості користувачів, що призводило до критичних помилок синхронізації. Критичний аналіз існуючих напрацювань свідчить про те, що значний час спільнота ігнорувала питання адаптивності та кросплатформенності, зосереджуючись виключно на функціональній повноті.

Більшість класичних систем, створених на початку 2010-х років, страждають від надмірної складності інтерфейсів та неоптимального використання обчислювальних ресурсів, що робить їх важкими для підтримки та масштабування. Попри суттєві успіхи у впровадженні хмарних технологій, багато теоретичних моделей все ще залишаються занадто абстрактними та не враховують специфіку вузькопрофільних ринків. На сьогодні залишаються невирішеними питання:

1. повної автоматизації конфліктних транзакцій у децентралізованих середовищах
2. інтеграції систем бронювання з різнорідними зовнішніми сервісами без втрати продуктивності.

Також помітна відсутність обґрунтованих підходів до створення легких архітектур. Такі архітектури повинні були б доступні для малого та середнього бізнесу. Тому важливо звернути увагу на впровадження механізмів динамічного керування потоками запитів та удосконалених алгоритмів перевірки доступності ресурсів у реальному часі. Розробка системи, яка поєднуватиме високу продуктивність із простим та зрозумілим інтерфейсом, є закономірним етапом розвитку сучасної програмної інженерії.

Забезпечення конфіденційності, цілісності та доступності персональних і транзакційних даних користувачів детермінується як один із домінантних векторів розвитку сучасної індустрії інформаційних технологій. Ретроспективний аналіз еволюції програмних систем свідчить, що на ранніх етапах проектування зазначені масиви інформації акумулювалися без застосування релевантних криптографічних та архітектурних механізмів безпеки, що зумовлювало високу уразливість до несанкціонованого доступу

та витоку даних. Крім того, дисипація (розпорошення) уваги розробників щодо питань сумісності, кросплатформової інтеграції із сучасними платіжними шлюзами та державними цифровими екосистемами призвела до виникнення системних колізій під час масштабування, модернізації та комплаєнсу (відповідності) архітектури ПЗ актуальним нормативним стандартам безпеки.

На сьогодні актуальною залишається проблема впровадження легких моделей прогнозовної аналітики, здатних завчасно визначати можливі пікові навантаження на систему бронювання. Це дозволило б автоматизовано перерозподіляти ресурси ще до виникнення критичних ситуацій. Також важливим питанням є оптимізація кросплатформенної взаємодії, оскільки підтримка однакового функціоналу на мобільних і настільних пристроях часто призводить до значного ускладнення структури програмного коду.

У науково-практичному дискурсі проектування сучасних систем бронювання особливої актуальності набуває інтеграція механізмів багатофакторної автентифікації, зорієнтованих на мінімізацію когнітивного й ергономічного навантаження на користувача та збереження високого рівня юзабіліті інтерфейсу. Паралельно критичним аспектом виступає впровадження інструментарію автоматизованого стрес-тестування для верифікації відмовостійкості та стабільності функціонування архітектури в умовах пікових навантажень. Як домінантний вектор розвитку визначено застосування модульних архітектурних рішень (зокрема мікросервісної організації), що забезпечують безперервну інтеграцію (CI/CD) нових платіжних шлюзів та протоколів верифікації без деградації або зупинки працездатності магістральної системи. Такий підхід забезпечує більшу гнучкість програмного забезпечення та підвищує його стійкість до змін технологічного середовища і сучасних вимог безпеки.

1.2. Аналіз відомих технічних рішень та технологічних процесів

Еволюція систем бронювання відбувалася від простих монолітних програм до сучасних розподілених архітектур, побудованих із використанням

хмарних технологій. На перших етапах основою таких рішень були реляційні бази даних із жорстко визначеною структурою, де запити оброблялися послідовно. Через це під час пікових навантажень виникали значні затримки в роботі системи. Основна увага під час розробки приділялася підтриманню цілісності даних, тоді як питання масштабованості залишалися другорядними.

Більшість ранніх систем використовували синхронну модель обробки запитів, через що користувач був змушений очікувати завершення всієї операції перед отриманням відповіді. В умовах сучасності та високої динаміки онлайн-сервісів такий підхід виявився недостатньо ефективним. Експоненційне зростання аудиторії та інтенсифікація використання сучасних вебплатформ зумовили об'єктивну необхідність ревізії традиційних парадигм проектування систем резервування. Особливої науково-практичної гостроти набула проблема забезпечення детермінованої відмовостійкості сервісів у періоди пікового (масового) надходження транзакційних запитів. Зазначений чинник виступив каталізатором переходу до динамічних архітектурних концепцій, базовими векторами яких є горизонтальна масштабованість, висока толерантність до відмов, а також оптимізація латентності при обробці великих масивів даних (Big Data) та високої щільності вхідного трафіку.

Ретроспективний аналіз технологічних рішень попереднього десятиліття свідчить про домінування монолітних і закритих систем, модернізація та супровід яких супроводжувалися високою ресурсомісткістю обчислювальних процесів. Дослідження архітектурних підходів вказує на те, що застосування реляційних баз даних із песимістичними механізмами блокування табличного рівня детермінувало регулярну генерацію взаємоблокувань (колізій типу «deadlock»). Зазначена деструкція виникала за умов високої конкурентності запитів, коли множинні потоки користувачів синхронно зверталися до унітарного обчислювального ресурсу. При цьому імплементація методів серверного кешування виявлялася інваріантною і не забезпечувала повного нівелювання проблеми нерівномірної дисипації навантаження.

Питання можливості у бік навантаження та автоматичного масштабування ресурсів у режимі нагального часу залишаються недостатньо втрішеними, особливо у рамках бюджетних програмних розробок. Традиційні алгоритми диспетчеризації черг вхідних запитів характеризуються дефіцитом урахування просторово-географічної диспозиції клієнтських терміналів та динамічних параметрів мережевої латентності. Таке ігнорування топологічних характеристик мережі призводить до нерівномірного розподілу часу затримки пакетів та деградації обчислювальної ефективності вебресурсу. У підсумку це зумовлює зниження нормативних показників якості обслуговування та суб'єктивної оцінки якості сприйняття системи користувачем.

Першочергової імплементації потребують обчислювальні процеси, що базуються на патернах асинхронної обробки повідомлень та подійно-орієнтованому неблокуючому введенні-виведенні. Задля підтримання константної релевантності та консистентності даних доцільно залучати протоколи повнодуплексного зв'язку в режимі реального часу. Це забезпечує реактивну синхронізацію станів об'єктів резервування та динамічне оновлення клієнтського інтерфейсу без ініціалізації повторних HTTP-запитів користувачем. Фундаментальним вектором розвитку визначено інтеграцію ізольованих мікросервісних рішень із вбудованими механізмами патернів відмовоустійчивості, що гарантує збереження загальної працездатності системи за умов каскадної деградації її окремих компонентів. Розробка гнучких API-інтерфейсів для безпечної взаємодії з платіжними системами та сервісами ідентифікації є ключовим етапом у створенні конкурентоспроможного програмного забезпечення для бронювання квитків. Сучасний процес розробки систем бронювання все більше зміщується у бік використання архітектури Single Page Application (SPA) у поєднанні з потужними інструментами керування станом на стороні клієнта. Це дозволяє перенести значну частину обчислювального навантаження з сервера на пристрій користувача, що забезпечує плавність інтерфейсу під час

інтерактивного вибору місць на схемах залів або під час бронювання білетів на екскурсії до музеїв.

Однак критичний огляд робіт показує, що такий підхід часто супроводжується проблемами з безпекою бізнес-логіки, яка може бути вразливою до маніпуляцій на рівні клієнтського коду. Велика кількість існуючих технічних рішень не враховувала необхідність впровадження наскрізного шифрування на рівні прикладних протоколів та використання токенованої авторизації (наприклад, JSON Web Token (JWT)), що створювало ризики перехоплення сесій під час процесу оформлення квитка. У системі бронювання необхідно не забувати про безпеку клієнта, тому важливо забезпечити перевірку дій користувача зі сторони сервера, за причини того що частина бізнес-логіки може бути змінена, або підроблена у його коді. Якщо про це забути, то виникає ризик несанкціонованого бронювання, зміни вартості квитка або створення тимчасового дублікату профілю для повторного використання відкритої сесії. На даний момент для зменшення таких загроз сучасні вебсистеми використовують захищену передачу даних HTTPS-протокол, токеновану авторизацію JWT, а також механізми перевірки автентичності запитів на сервері. HTTPS це протокол який є розширеною версією протокола HTTP. У протоколі передача даних здійснюється за допомогою використання криптографічних протоколів SSL або TLS. Завдяки їх використанню більша частина інформації, яка передається користувачем серверу, шифрується, зокрема URL-запити, параметри даних та cookie-файли, які можуть містити інформацію про сесію користувача. Назва хоста та номер порту не підлягають шифруванню через те, що необхідні для встановлення коректного мережевого з'єднання протоколами TCP/IP.

Незважаючи на активне впровадження хмарних сервісів, залишається невирішеним питання ефективної контейнеризації компонентів системи для забезпечення їх миттєвого розгортання в ізольованих середовищах. Проблема «холодного старту» окремих модулів системи під час різкого зростання трафіку досі залишається актуальною для багатьох комерційних аналогів, що

призводить до тимчасової недоступності сервісу в моменти відкриття продажів на популярні події. Також залишається недостатньо дослідженим питання автоматизованого забезпечення узгодженості даних у гібридних системах зберігання даних, де реляційні бази використовуються для обробки фінансових операцій, а NoSQL-рішення для швидкого кешування інформації про доступність квитків. У зв'язку з цим, першочергового впровадження потребують технологічні процеси безперервної інтеграції та доставки (CI/CD), які дозволяють проводити оновлення системи без зупинки обслуговування користувачів. Пріоритетним технічним рішенням має стати впровадження механізмів розподіленого кешування на рівні окремих вузлів системи, що дозволить мінімізувати кількість прямих звернень до основної бази даних. Окрім цього, важливим аспектом є використання сучасних протоколів захисту під час взаємодії з банківськими еквайринговими сервісами, що дозволить забезпечити безпечне проведення фінансових операцій. Зосередження уваги на зазначених інженерних рішеннях дасть можливість наблизити розроблювану систему до сучасних галузевих стандартів, забезпечуючи її надійність, стабільність роботи та захищеність у реальних умовах використання.

1.3. Аналіз існуючих програмних продуктів та аналогів системи

Ринок програмного забезпечення для бронювання квитків пройшов значний шлях розвитку — від закритих корпоративних мереж авіакомпаній до сучасних глобальних систем дистрибуції. Початкові етапи розвитку даного напрямку були пов'язані зі створенням централізованих систем обробки інформації, серед яких одними з найвідоміших стали системи Sabre та Amadeus. Зазначені технологічні розробки сформували канонічний базис цифрового обліку ресурсів та автоматизації процесів резервування.

Платформа Sabre виступила однією з перших глобальних комп'ютерних систем дистрибуції, що забезпечила автоматизацію транзакцій авіаперевезень і детермінувала вектор розвитку електронного сегмента індустрії. У свою чергу, система Amadeus була спроектована консорціумом європейських

авіаперевізників як централізований комплекс для консолідованого бронювання пасажирських місць, готельного фонду та об'єктів культурно-розважальної інфраструктури. Системний аналіз сучасних рішень свідчить, що попри фундаментальний внесок у генезис автоматизованих систем, класичні платформи характеризуються високою інтеграційною складністю та вимагають значних фінансових операційних витрат, що обмежує їх імплементацію в секторі малого та середнього бізнесу. Новітні комерційні сервіси (зокрема Booking.com або Tickets.ua) володіють диверсифікованим функціоналом, проте переважно функціонують у межах посередницької парадигми. Це створює архітектурні бар'єри щодо латентності обміну даними та нівелює можливість гнучкої кастомізації системи під специфічні вимоги локальних провайдерів послуг. Попри проліферацію (розмноження) готового програмного забезпечення, низка науково-технічних задач залишається невирішеною. Серед них - надмірна функціональна надмірність систем, яка зумовлює деградацію продуктивності у мобільних браузерів та на терміналах з обмеженими апаратними ресурсами. Більшість існуючих платформ базується на ригідних (жорстких) архітектурних паттернах, що ускладнює адаптацію логіки під нестандартні типи квитків або специфічні топології просторового розміщення (схеми розсадки) без масштабного рефакторингу вихідного коду ПЗ. Крім того, спостерігається дефіцит рішень, які когнітивно поєднують високі критерії криптографічного захисту фінансових транзакцій із прозорими, інтуїтивно зрозумілими інтерфейсними афродансами. У межах проектування запропонованої системи пріоритетну увагу доцільно приділити математичним механізмам динамічної візуалізації доступних ресурсів та оптимізованим алгоритмам диспетчеризації черг конкурентних запитів. Такий підхід дозволить запобігти виникненню транзакційних колізій під час масового синхронного звернення користувачів. Одним із ключових завдань дослідження є синтез легковагової архітектури, здатної мінімізувати час відгуку сервера та забезпечити оперативну реактивну синхронізацію статусів об'єктів між реляційною/нереляційною базою даних та клієнтською частиною.

Реалізація таких рішень дасть можливість усунути ключові недоліки більшості сучасних аналогів, забезпечивши високу швидкодію системи, її масштабованість та зручність використання.

Під час виконання кваліфікаційної роботи було проаналізовано вебсайти музеїв, зоопарків і замків міста Познань. У більшості випадків на цих ресурсах відсутня функція онлайн-бронювання квитків, а користувачам доступний лише актуальний прайс-лист, який змінюється залежно від сезону, дня або місяця. Також на сайтах переважно розміщуються фотографії закладів, проведених заходів та культурних подій.

РОЗДІЛ 2. АНАЛІЗ ТА ПРОЄКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1. Специфікація вимог до програмного забезпечення та визначення пріоритетів розробки

Підходи до формування вимог для систем бронювання з часом суттєво змінилися — від опису базових операцій зі збереження даних до створення комплексних специфікацій, що враховують доступність, безпеку та адаптивність системи. Розвиток методологій у цій сфері відбувався від жорстких моделей планування, де вимоги визначалися лише на початковому етапі розробки, до сучасних гнучких підходів, які дозволяють змінювати функціональність відповідно до потреб користувачів та ринкових умов.

Аналіз попередніх підходів до документування вимог показує, що значна увага приділялася кількісним характеристикам функціоналу, тоді як зручність взаємодії користувача з інтерфейсом часто залишалася другорядною. Як наслідок, у процесі розробки синтезувалися програмні комплекси, що формально відповідали вихідним технічним вимогам, проте під час експериментальної експлуатації демонстрували високу ергономічну складність та втрату стабільності в умовах пікових навантажень. Сучасний науково-методичний інструментарій недостатньо враховує специфіку конвергенції (об'єднання) вебплатформ із гетерогенними платіжними шлюзами та сервісами автоматизованого сповіщення, що суттєво обмежує потенціал наскрізної автоматизації бізнес-процесів. Актуальними невирішеними задачами наразі залишаються гарантування відмовостійкості систем за умов нестабільного мережевого з'єднання, а також підтримка транзакційної консистентності й цілісності даних при паралельному доступі. У чинних технічних специфікаціях спостерігається дефіцит деталізованих сценаріїв обробки виняткових ситуацій при ініціалізації фінансових транзакцій, що виступає тригером колізій на рівні СУБД. Крім того, фіксується істотна дивергенція між теоретичними моделями безпеки персональних даних та їх практичною імплементацією у вебдодатках, що функціонують у

незахищених розподілених мережах. Відсутність уніфікованих стандартів синхронізації станів об'єктів у багатокористувацькому середовищі призводить до латентних архітектурних дефектів, які верифікуються виключно під час навантажувального стрес-тестування. У цьому контексті першочергової детермінації потребують вимоги щодо реактивного (миттєвого) оновлення статусів квитків та забезпечення високого рівня живучості серверної інфраструктури. Пріоритетним є створення специфікації, яка визначатиме параметри швидкодії, вимоги до безпеки автентифікації користувачів та логіку взаємодії з базою даних через асинхронні неблокуючі запити. Також важливо сформулювати вимоги до функцій, що забезпечуватимуть повний цикл бронювання — від вибору місця до автоматичної генерації підтверджувальних документів. Реалізація зазначених пріоритетів дозволить створити надійну основу для подальшої розробки програмного забезпечення відповідно до сучасних вимог якості та надійності.

У межах кваліфікаційної роботи планується розробка вебсайту, який міститиме три основні напрямки бронювання квитків:

1. Музеї
2. Зоопарки
3. Замки

Для кожного напрямку буде створено перелік закладів міста Познань та його околиць. Назви закладів відображатимуться польською мовою, тоді як опис буде представлений українською, оскільки сайт орієнтований переважно на українських користувачів, які планують відвідати культурно-розважальні заклади Познані та прилеглих територій.

Для кожного закладу передбачається окрема сторінка з коротким описом, інформацією про кількість вільних місць в екскурсійній групі, можливістю вибору дати відвідування та актуальними цінами, що були чинними на момент створення кваліфікаційної роботи.

У межах роботи буде реалізовано демонстраційний модуль оплати, оскільки створення повноцінної платіжної системи потребує відповідних

дозволів та офіційної реєстрації підприємницької діяльності. Система бронювання міститиме обов'язкові поля для введення імені та прізвища користувача, а також можливість вибору дати, типу та кількості квитків. Після вибору квитків система автоматично відображатиме загальну вартість замовлення.

Також буде реалізована функція завантаження електронного квитка, у якому зазначатимуться:

1. Назва закладу
2. Ім'я
3. Прізвище
4. Дата
5. Номер екскурсійної групи
6. Тип та кількість квитків
7. Ціна
8. Повідомлення про завчасне прибуття до місця події

Оскільки заклади можуть відвідувати діти, у системі передбачатиметься інформаційне попередження про те, що діти без супроводу дорослих не можуть знаходитися на екскурсії самостійно. Таке повідомлення допоможе користувачам уникнути помилок під час оформлення квитків для дітей та забезпечить можливість перебування дітей в одній екскурсійній групі разом із батьками або супроводжуючими особами.

2.2. Вибір та аналіз програмного забезпечення для розробки вебсайту

Основою кваліфікаційної роботи є створення програмного продукту, орієнтованого на спрощення процесу бронювання квитків до музеїв, зоопарків та замків. Вебсайт, який планується розробити, буде спрямований на реалізацію зручного та доступного механізму бронювання для користувачів. Туристичні вебсайти на яких в даний момент можна забронювати квитки до музеїв, зоопарків та замків, пропонують бронювати лише комплексні види екскурсій. Це не завжди є доцільним, особливо коли людина не бачила лише один музей з переліку і він стоїть останнім. Через це програмний продукт буде

розроблений на можливості вибору бронювання екскурсії для одного конкретного музею, зоопарку, замку. Майже всі заклади мають дні в які є знижки на квитки або взагалі безкоштовні дні відвідування, і про це туристичні сайти навіть не повідомляють.

Для створення вебсайту буде використовуватися низка програм таких як:

1. Sublime Text 3



Рисунок 2.1 Емблема програми Sublime Text 3

2. XAMPP



Рисунок 2.2 Емблема програми XAMPP

1.2.1. Apache



Рисунок 2.3 Емблема вебсерверу Apache

1.2.2. MySQL



Рисунок 2.4 Емблема програми MySQL

3. Google Chrome



Рисунок 2.5 Емблема браузера Google Chrome

Sublime Text 3 – це швидкий кросплатформенний текстовий редактор. Підтримує плагіни, розроблені за допомогою мови програмування Python.[5] Його інтерфейс простий та зручний, він підтримує написання html, css, php, javascript, що є необхідним для комфортної розробки вебсайту. Додатковою

перевагою редактора є висока швидкість роботи навіть при відкритті великої кількості файлів, а також підтримка підсвічування синтаксису та автоматичного форматування коду. Це значно спрощує процес написання та редагування програмного коду під час створення вебпроєкту.

HTML – це мова гіпертекстової розмітки для створення вебсайту, основа та фундамент сайту. Саме HTML відповідає за структуру вебсторінки, розміщення тексту, кнопок, таблиць, зображень та інших елементів інтерфейсу. За допомогою HTML формується логічна структура сторінок вебсайту, що забезпечує правильне відображення контенту у браузері користувача.

CSS – відповідає за візуальну складову та зовнішній вигляд вебсторінок, шрифти, колір, розмір тексту, блоків, малюнків, таблиць, та інших елементів. CSS дозволяє створити єдиний стиль оформлення всіх сторінок вебсайту. Також використовується для створення анімацій, ефектів наведення та покращення загального дизайну користувацького інтерфейсу.

PHP – це популярна мова програмування, особливо серед веб-розробників. Автором початкової версії є Расмус Лердорф, ідея якого полягала у розробці набору інструментів спрощення процесу створення динамічних веб-сторінок. Незважаючи на те, що сучасний PHP є мовою загального призначення, найчастіше його використовують як серверний інструмент для генерації HTML-коду, який потім інтерпретується браузером.[6]

JavaScript (JS) – це високорівнева скриптова мова програмування, яка широко використовується для створення інтерактивних застосунків і сайтів. Вона може створювати анімацію, спливаючі повідомлення і змушує інтерфейс реагувати на наші дії.[7] У зв'язці з HTML і CSS JavaScript надає змогу розробити повнофункціональний вебсайт із динамічним оновленням інформації без перезавантаження сторінки. У межах проєкту JavaScript буде використовуватись для перевірки правильності введення даних у формах бронювання, автоматичного обчислення вартості квитків, взаємодії з елементами інтерфейсу та покращення загального користувацького досвіду.

XAMPP – це простий в налаштуванні та безкоштовний локальний вебсервер. Така назва не просто так, це є аббревіатура яка розшифровується на компоненти.

X – визначає її кросплатформенність, вона працює на Windows, Linux, MacOS.

A – Apache, компонент програми, безкоштовний вебсервер із відкритим кодом, який є центральним пакетом програми.

M – MySQL (MariaDB), компонент програми, це система керування реляційними базами даних. Завдяки XAMPP можна працювати з MySQL в зручному інтерфейсі phpMyAdmin.



Рисунок 2.6 Емблема інтерфейсу phpMyAdmin

P – PHP, компонент програми, який допомагає пов'язати базу даних з сайтом.

P – Perl, компонент програми, мова програмування для автоматизації та розробки сайтів, однак цю мову витіснила мова PHP. Цей компонент програми не знадобиться для розробки сайту.

Google Chrome – це безкоштовний веббраузер, розроблений компанією Google на основі браузера з відкритим кодом Chromium та іншого відкритого програмного забезпечення.[8] Він потрібен для перевірки коректності розміщення блоків сайту, та його вигляду.

2.3. Проектування користувацького інтерфейсу вебсайту

Проектування користувацького інтерфейсу вебсайту є важливою складовою розробки програмного забезпечення про яку неможна забувати. Від якості реалізації залежить швидкість пошуку необхідної інформації, зручність навігації та загальне враження користувача від роботи з вебсайтом. Інтерфейс

повинен бути зрозумілим навіть для користувачів з малим досвідом роботи в мережі Інтернет, оскільки складна структура сторінок або перевантаженість елементами можуть ускладнювати процес бронювання квитків.

Під час проєктування вебсайту потрібно враховувати такі основні аспекти:

1. Простота сторінок
2. Логічність та послідовність розміщення елементів
3. Функціональність інтерфейсу
4. Зручність навігації
5. Адаптивність для різних пристроїв

Основні функції вебсайту повинні бути доступними без виконання великої кількості дій. Особлива увага повинна бути направлена на навігацію сайтом. Користувач повинен мати можливість швидко перейти до необхідного розділу, переглянути доступні заклади, ознайомитися з інформацією про події та здійснити бронювання квитка без складних переходів між сторінками. Для цього потрібно реалізувати зрозуміле меню навігації, логічну структуру сторінок та помітні кнопки переходу між основними розділами системи.

Під час розробки інтерфейсу сайту також необхідно враховувати психологічні аспекти сприйняття інформації. Інформаційна надмірність, деструктивні колористичні схеми та перевантаженість композиційних шаблонів вебсторінок підвищують рівень когнітивного навантаження на користувача, що лімітує швидкість перцепції (сприйняття) та верифікації релевантних даних. З огляду на це, обґрунтованою є імплементація принципів мінімалістичного дизайну, що передбачає нормовану квантифікацію колірної палітри, однозначну семантику елементів керування та логічну кластеризацію контенту. Для фокусування уваги оператора на магістральних функціональних вузлах системи застосовано методи візуального акцентування цільових компонентів інтерфейсу — зокрема інтерактивних модулів ініціалізації резервування, селекторів часових інтервалів (дат) та фінального підтвердження транзакцій. Окремої детермінації потребує адаптивність клієнтської частини. Зважаючи на стійку тенденцію превалювання мобільного

трафіку (смартфонів та планшетних EOM), архітектура фронтенду має гнучко трансформувати просторову топологію елементів відповідно до геометричних параметрів екрана (responsive design), гарантуючи стабільний рівень кросплатформного юзабіліті (\$usability\$). Ефективність взаємодії з платформою також корелює з часом інтерфейсного відгуку. Система повинна оперативно обробляти користувацькі івенти, забезпечуючи низьку латентність (\$latency\$) при динамічній реплікації даних про доступний квитковий фонд, що мінімізує операційні помилки та максимізує користувацький досвід (\$UX\$). Важливою підсистемою є модуль покрокового оформлення (\$wizard-pattern\$) та рендерингу електронних ваучерів (квитків). Процедuru лінійної дистрибуції оптимізовано шляхом декомпозиції на послідовні дискретні етапи без надмірних транзакційних кроків. Інтерфейс забезпечує перманентну валідацію введених параметрів, відображає квоти вільних місць у режимі реального часу, а після завершення сесії автоматично генерує електронний документ у детермінованому форматі (наприклад, PDF) із підтримкою механізмів візуальної ієрархії для швидкого зчитування атрибутів замовлення. Інтерфейсні рішення, що супроводжуються контекстними повідомленнями та валідаційними тригерами, спроектовані в уніфікованому стилі з дотриманням гайдлайнів доступності (accessibility, W3C Web Accessibility Initiative) для різних вікових груп. Прототипування та композиційне моделювання інтерфейсу виконано у середовищі векторного графічного редактора Figma.. Створення макетів у середовищі Figma дозволяє попередньо оцінити зовнішній вигляд вебсайту, структуру сторінок та взаємодію між елементами інтерфейсу ще до початку програмної реалізації. Це значно спрощує процес проєктування та дозволяє своєчасно виявити можливі недоліки структури або дизайну. Також використання макетів допомагає сформулювати чітке уявлення про кінцевий вигляд програмного продукту та забезпечує більш ефективну організацію подальшої розробки вебсайту.

На сайті планується розробити такі сторінки:

1. Головна сторінка

2. Сторінка музеїв
3. Сторінка зоопарків
4. Сторінка замків
5. Загальна сторінка для кожного музею
6. Загальна сторінка для кожного зоопарку
7. Загальна сторінка для кожного замку
8. Сторінка бронювання
9. Сторінка перевірки правильності введеної інформації
10. Сторінка успішності бронювання, з можливістю завантаження PDF варіанту квитка.

Макет головної сторінки виглядає ось так:

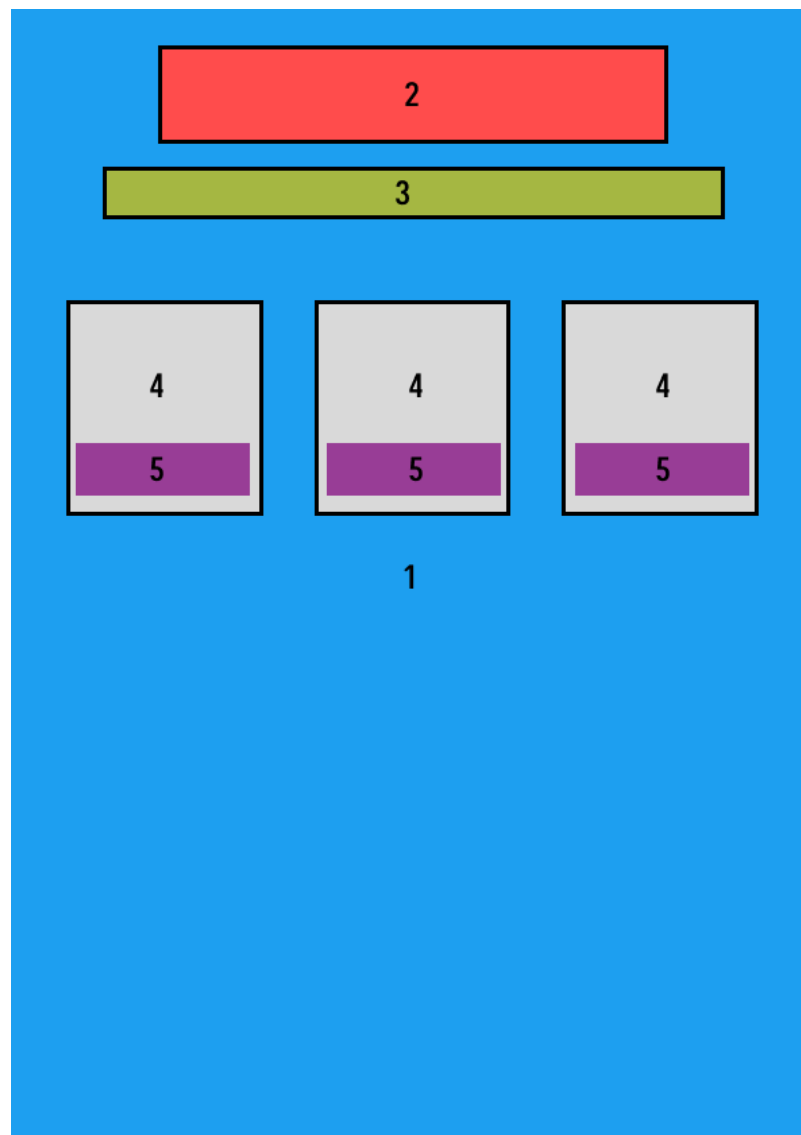


Рисунок 2.7 Макет головної сторінки сайту

Під час проєктування особлива увага приділялася простоті сприйняття інформації, зручності навігації та логічному розміщенню основних елементів

інтерфейсу. Усі основні блоки розташовані таким чином, щоб користувач міг швидко та легко знайти потрібну інформацію та перейти до потрібної категорії закладів без виконання великої кількості дій.

Під номером 1 фон сторінки. Він виконує роль контейнера для всіх блоків сайту та забезпечує цілісність візуального оформлення вебсторінки. Також фон створює загальний стиль інтерфейсу та впливає на комфортне сприйняття інформації користувачем.

У позиції 2 розташовано ідентифікатор вебресурсу (назву сайту), ключове призначення якого полягає в інформаційній підтримці та спрощенні візуальної навігації користувача. У подальших модифікаціях інтерфейсу допускається масштабування цієї області шляхом розміщення графічного логотипу чи допоміжних елементів керування.

Під номером 3 буде знаходитись короткий опис сайту. Даний блок призначений для ознайомлення користувача з основними можливостями вебресурсу. Це дозволить новим користувачам швидше зрозуміти призначення системи.

Під номером 4 знаходяться картки категорій. Вони відповідають за кожен категорію музеї, зоопарки, замки. В кожній картці буде знаходитися іконка відповідної категорії, що дозволить користувачу швидко візуально відрізнити розділи між собою. Використання карток робить інтерфейс більш сучасним та покращує зручність навігації по вебсайту.

Під номером 5 – назва категорії та її короткий опис. Цей блок дозволяє користувачу швидко ознайомитися зі змістом відповідного розділу та зрозуміти його призначення. Після натискання на картку користувач буде переходити на окрему сторінку обраної категорії, де буде відображатися перелік доступних закладів та можливість переходу до системи бронювання квитків.

Під час створення макета також враховується адаптивність вебсайту для різних типів пристроїв. Розташування елементів інтерфейсу повинно автоматично змінюватися залежно від розміру екрана користувача, що

забезпечить комфортне користування вебсайтом як на персональних комп'ютерах, так і на смартфонах або планшетах.

Макет сторінки категорії виглядає ось так:

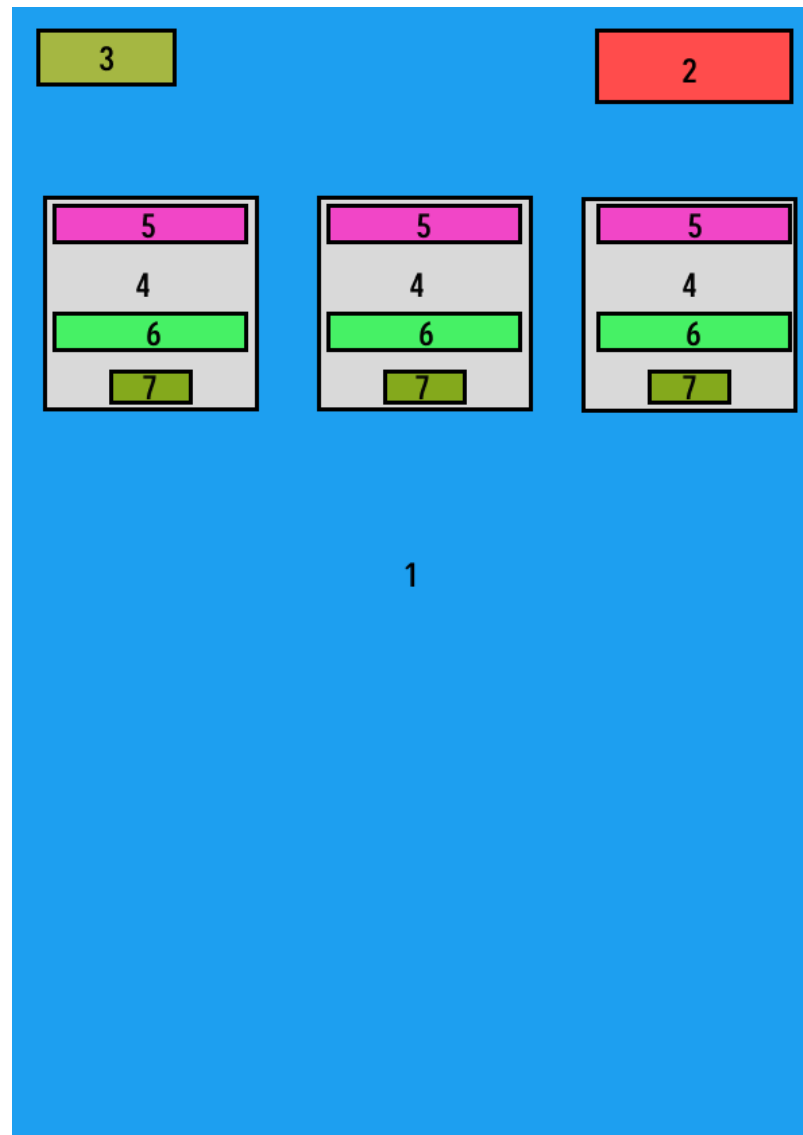


Рисунок 2.8 Макет сторінки категорії сайту

Підчас проектування даної сторінки особлива увага приділялася зручності перегляду списку доступних закладів та швидкому переходу до більш детальної інформації про кожний заклад. Структура сторінки проектується таким чином, щоб користувач міг швидко знайти потрібний заклад та перейти до оформлення бронювання.

Під номером 1 фон сторінки. Він виконує роль контейнера для блоків сторінки сайту. Фон сторінки планується робити в одному стилі для забезпечення єдиного стилістичного оформлення сайту. Використання

єдиного фону для всіх сторінок дозволить створити єдиний цілісний дизайн вебсайту та покращує сприйняття інформації користувачем.

Під номером 2 буде знаходитись назва категорії, яку вибрав користувач. Даний блок допоможе користувачеві швидко зрозуміти, в якому саме розділі сайту він буде знаходитись. Назва блоку буде зроблена великим та помітним шрифтом для покращення навігації та зручності користувача.

Під номером 3 буде знаходитись кнопка повернення на головну сторінку сайту. Наявність такої кнопки надає можливість користувачеві швидко переходити між сторінками сайту та спрощує навігацію по ньому. Це особливо важливо, коли користувач хоче змінити категорію блоку в якому буде знаходитись.

Під номером 4 знаходяться картки закладів вибраної категорії: музеї, зоопарки, замки. Саме вони виконують роль головних інформаційних елементів сторінки та забезпечують компактне відображення інформації про доступні заклади. Використання карток дозволяє зробити структуру сторінки більш зрозумілою та сучасною.

Під номером 5 буде знаходитись назва картки закладу. Назва дозволяє швидко ідентифікувати заклад або знайти потрібний серед запропонованих. Назви планується робити мовою оригіналу.

Під номером 6 короткий опис закладу. У даному блоці буде розміщена коротка інформація про музей, зоопарк або замок, що дозволить користувачу попередньо ознайомитися з особливостями обраного місця без необхідності переходу на окрему сторінку.

Під номером 7 буде знаходитись кнопка для переходу на сторінку закладу для ознайомлення з ним більш детально.

Під час проєктування сторінки категорії також враховується адаптивність інтерфейсу для різних типів пристроїв. Розташування карток закладів повинно автоматично змінюватися залежно від ширини екрана користувача, що забезпечить комфортне користування вебсайтом як на персональних комп'ютерах, так і на мобільних пристроях.

Макет сторінки опису закладу виглядає ось так:

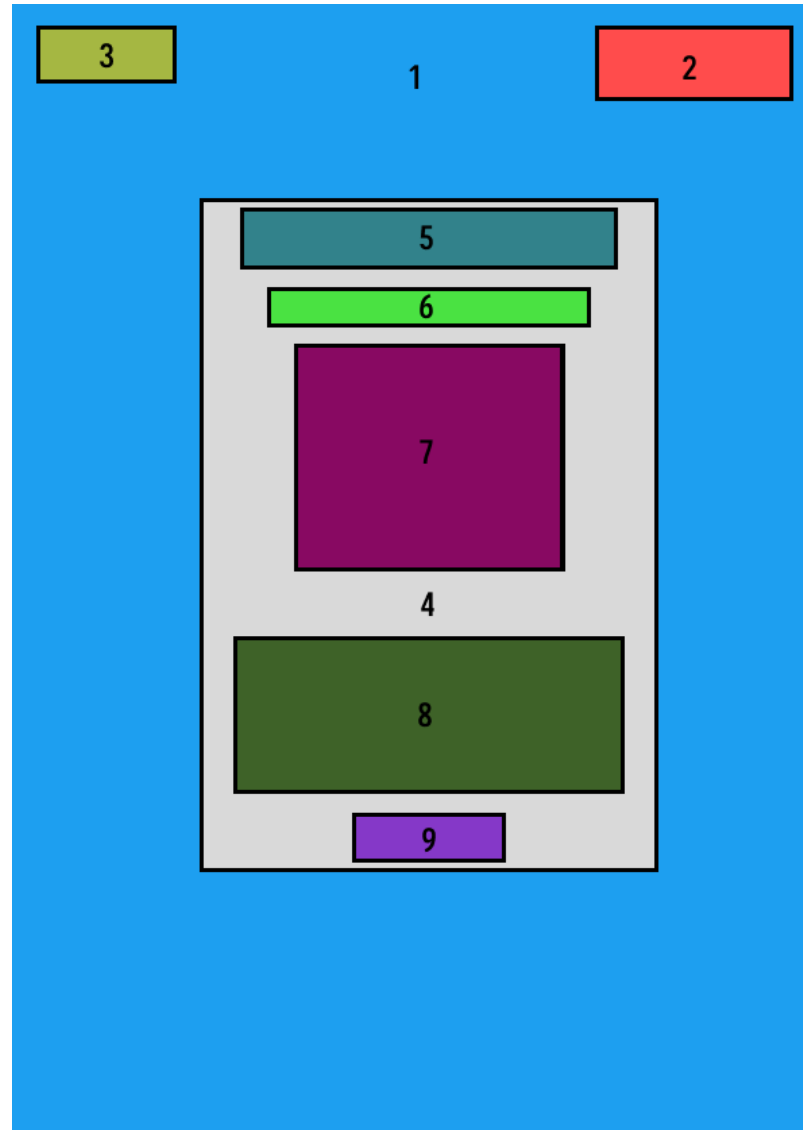


Рисунок 2.9 Макет сторінки опису закладу

Дана сторінка призначена для детального ознайомлення користувача з вибраним закладом та подальшого переходу до системи бронювання квитків. Під час проєктування сторінки основна увага приділялася зручному розташуванню інформації, простоті навігації та забезпеченню комфортного сприйняття контенту користувачем.

Під номером 1 фон сторінки. Він виконує роль контейнера для блоків сторінки сайту. Фон сторінки планується робити в одному стилі для забезпечення єдиного стилістичного оформлення сайту. Використання єдиного фону для всіх сторінок дозволить створити єдиний цілісний дизайн вебсайту та покращує сприйняття інформації користувачем.

У позиції 2 розташовано ідентифікатор вебресурсу (назву сайту), ключове призначення якого полягає в інформаційній підтримці та спрощенні візуальної навігації користувача. У подальших модифікаціях інтерфейсу допускається масштабування цієї області шляхом розміщення графічного логотипу чи допоміжних елементів керування.

Під номером 3 буде знаходитись кнопка повернення на попередню сторінку сайту. Наявність такої кнопки надає можливість користувачеві швидко переходити між сторінками сайту та спрощує навігацію по ньому. Це суттєво важливо, коли користувач хоче повернутися до попереднього розділу вебсайту.

Під номером 4 буде знаходитись картка закладу обрана користувачем, з більшою кількістю інформації. Даний блок є основним елементом сторінки та містить всю необхідну інформацію про заклад.

У позиції 5 відображається автентична назва установи мовою оригіналу. Такий підхід забезпечує збереження номінативної автентичності об'єктів (музеїв, зоопарків, замкових комплексів тощо), а також оптимізує процес просторової орієнтації та пошуку локацій користувачем безпосередньо під час перебування в урбаністичному просторі міста Познань.

Під номером 6 назва закладу українською або короткий опис. Даний блок допомагає користувачам краще зрозуміти призначення закладу та отримати коротку інформацію про нього без необхідності читання повного опису.

Під номером 7 детальний опис закладу. У цьому блоці планується розміщення інформації про історію закладу, особливості екскурсій, доступні експозиції та інші важливі дані. Це дозволить користувачеві більш детально ознайомитися із закладом перед оформленням бронювання квитків.

У позиції 8 інтегровано фотоматеріали досліджуваного об'єкта. Залучення автентичного візуального контенту оптимізує когнітивне сприйняття вебсторінки та сприяє формуванню у користувача цілісного уявлення про екстер'єрні характеристики локації. Візуалізація в даному

контексті поєднує в собі інформативну та естетико-декоративну функції, що безпосередньо підвищує ергономічність, сучасність та атрактивність користувацького інтерфейсу.

У позиції 9 локалізовано інтерактивний елемент (кнопку) для ініціалізації процедури бронювання квитків. Зазначений компонент є ключовим елементом цільової конверсії сторінки, оскільки забезпечує архітектурний перехід користувача до субсистеми оформлення транзакцій. З метою оптимізації користувацької уваги та мінімізації часу пошуку, даний керуючий елемент потребує високого рівня візуальної контрастності та інтуїтивної зрозумілості, а його просторове розміщення має гарантувати миттєвий доступ до функціоналу бронювання.

Під час проектування сторінки опису закладу також враховується адаптивність інтерфейсу для різних типів пристроїв. Усі елементи сторінки повинні коректно змінювати своє розташування залежно від розміру екрана користувача, що забезпечить комфортне користування вебсайтом як на персональних комп'ютерах, так і на смартфонах або планшетах.

Макет сторінки бронювання виглядає ось так:

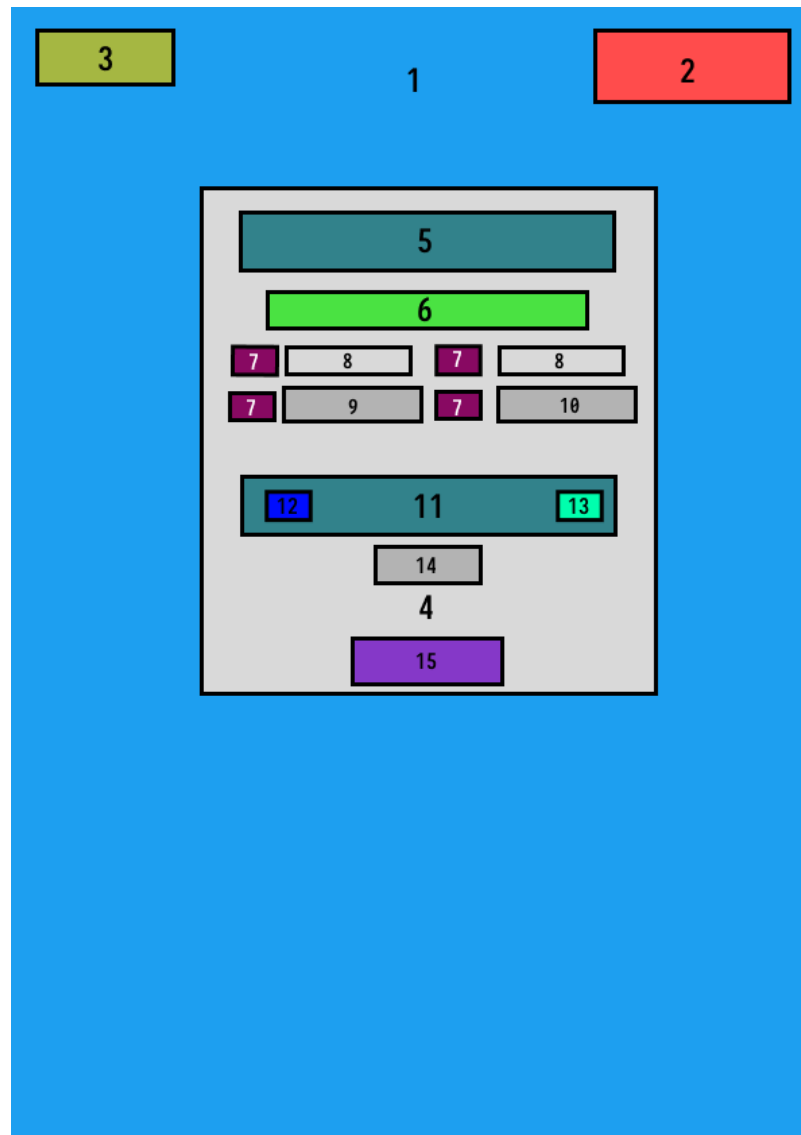


Рисунок 2.10 Макет сторінки бронювання

Дана сторінка призначена для бронювання квитків. Під час проєктування сторінки основна увага приділялася зручному розташуванню інформації, простоті навігації та забезпеченню комфортного сприйняття контенту користувачем. Інтерфейс сторінки повинен забезпечувати швидке та зрозуміле оформлення бронювання без виконання зайвих дій. Усі основні елементи розташовуються у логічній послідовності для покращення взаємодії користувача із системою.

Під номером 1 фон сторінки. Він виконує роль контейнера для блоків сторінки сайту. Фон сторінки планується робити в одному стилі для забезпечення єдиного стилістичного оформлення сайту. Застосування

уніфікованого фонового рішення для всього масиву вебсторінок забезпечує формування цілісної візуально-графічної архітектури вебресурсу та інтенсифікує процеси перцептивного сприйняття інформації користувачем. Системна однорідність фонового простору нівелює когнітивний дисонанс при переході між розділами, оптимізуючи взаємодію суб'єкта з контентом.

У позиції 2 розташовано ідентифікатор вебресурсу (назву сайту), ключове призначення якого полягає в інформаційній підтримці та спрощенні візуальної навігації користувача. У подальших модифікаціях інтерфейсу допускається масштабування цієї області шляхом розміщення графічного логотипу чи допоміжних елементів керування.

Позиція 3 відведена під елемент навігації (кнопку) повернення до попереднього стану інтерфейсу. Наявність цього компонента оптимізує реверсивну навігацію вебресурсу, мінімізує часові витрати користувача при міжсторінкових переходах та спрощує загальну архітектуру взаємодії, що є критично важливим за потреби оперативного повернення суб'єкта до попереднього інформаційного розділу.

Позиція 4 містить інтерактивну форму бронювання, яка виступає функціональним ядром сторінки. Зазначений блок акумулює повний масив релевантної інформації, необхідної для оформлення квитків, та забезпечує мінімізацію ергономічних зусиль при введенні персональних даних.

Позиція 5 репрезентує номінативний маркер установи мовою оригіналу. Таке інженерне рішення спрямоване на збереження автентичності власних назв історико-культурних та інфраструктурних об'єктів (музеїв, зоопарків, замкових комплексів тощо), а також оптимізує процеси просторової орієнтації та верифікації локацій користувачем під час перебування в урбаністичному середовищі міста Познань.

Позиція 6 відведена під україномовний аналог найменування або лаконічний дескриптор об'єкта. Цей функціональний елемент забезпечує інформаційну валідацію, допомагаючи користувачеві верифікувати правильність вибору цільового закладу.

Позиція 7 інтегрує текстові блоки експлікації (підписи полів), які виконують когнітивно-навігаційну функцію та координують дії користувача у суміжних зонах інтерфейсу (а саме - спонукають до введення ідентифікаційних даних та вибору дати візиту). Використання експліцитних маркерів текстових полів підвищує семантичну прозорість форми та знижує ймовірність помилкових дій оператора під час заповнення.

Позиція 8 поєднує два дискретні текстові поля для введення персональних даних (імені та прізвища) з метою ініціалізації процедури резервування. Зазначені поля детерміновані як обов'язкові атрибути для генерації електронного документа (квитка) та забезпечення подальшої автентифікації користувача в системі.

Позиція 9 представлена інтерфейсним компонентом типу Time Picker (вибір часу), за допомогою якого здійснюється селекція відповідного часового слота на основі динамічного моніторингу доступного квотного ліміту квитків.

Позиція 10 містить елемент вибору календарної дати (Date Picker). Диференціація часових параметрів у формі окремого візуального компонента оптимізує моніторинг доступних варіантів та спрощує алгоритм фінального оформлення замовлення.

Позиція 11 локалізує блок селекції категорій квитків. У межах цього модуля реалізовано можливість диференціації тарифних планів (зокрема, дорослий, дитячий, пільговий), що дозволяє системі в автоматичному режимі обчислювати інтегральну вартість замовлення на основі обраних пресетів.

Позиція 12 функціонує як інформаційне поле класу квитка, призначене для візуалізації специфікацій та умов обраної категорії, що підвищує рівень поінформованості користувача під час прийняття рішень.

Позиція 13 містить інтерактивне поле введення кількісних показників квитків для резервування. На рівні програмного забезпечення реалізовано модуль валідації вхідних значень для блокування некоректного або аномального введення даних.

Позиція 14 репрезентує блок динамічного калькулювання сукупної вартості замовлених послуг. Цей елемент UI-дизайну забезпечує миттєвий зворотний зв'язок (візуалізацію підсумкової суми), що суттєво підвищує ергономічність та зручність експлуатації системи бронювання.

Позиція 15 відведена під фінальний керуючий елемент (кнопку підтвердження), який завершує транзакційну процедуру та ініціює перехід до етапу верифікації або генерації вихідного електронного квитка. Задля оптимізації перцептивної уваги користувача даний елемент має виражений візуальний акцент (акцентний колір, розмір) на тлі інших компонентів сторінки.

Додатково, у процесі архітектурного проектування сторінки бронювання було враховано вимоги інженерної психології щодо обов'язкової інтеграції модулів модальних сповіщень, системних попереджень та інформаційних тригерів для забезпечення безпомилкової взаємодії суб'єкта з вебплатформою. Наприклад, система повинна повідомляти про необхідність перевірки правильності введених даних, обмеження щодо дитячих квитків або відсутність вільних місць на обрану дату. Це дозволить підвищити надійність роботи системи та зменшити кількість помилок під час оформлення бронювання.

Окрему увагу приділено адаптивності сторінки бронювання для різних типів пристроїв. Усі елементи форми повинні коректно змінювати своє розташування залежно від ширини екрана користувача, що забезпечить комфортне користування системою бронювання як на персональних комп'ютерах, так і на смартфонах або планшетах.

Макет сторінки для перевірки інформації виглядає ось так:

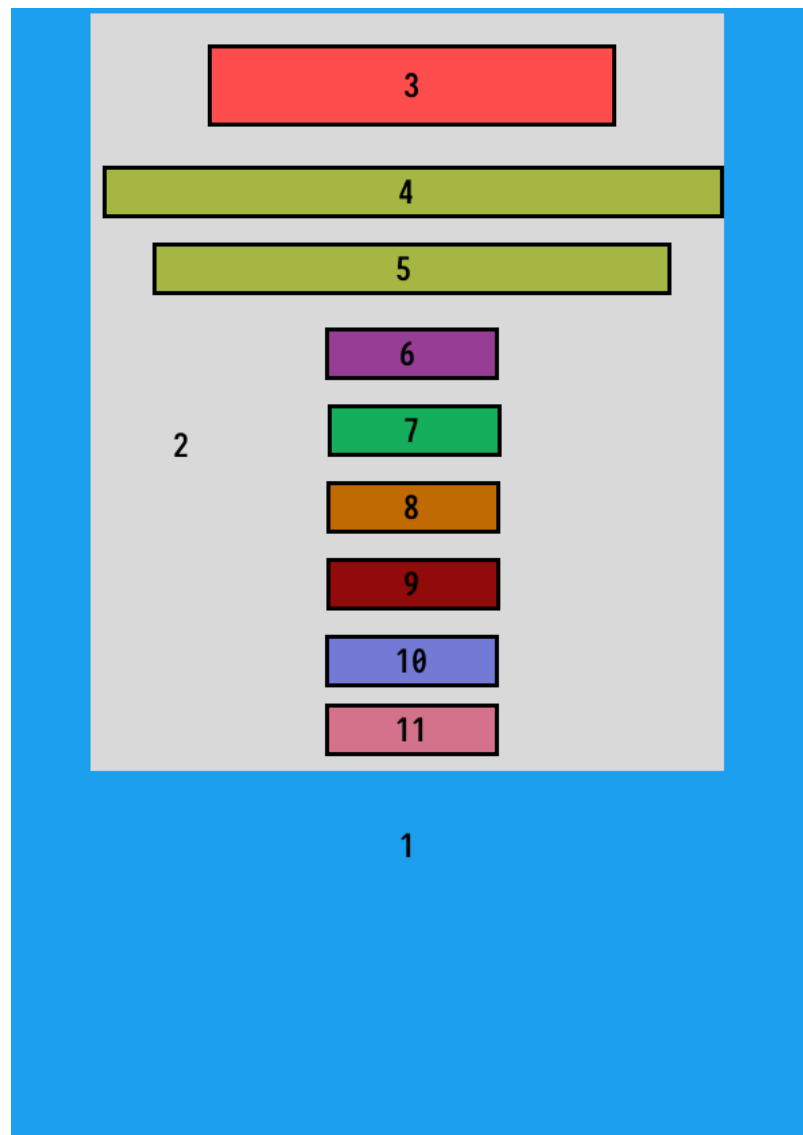


Рисунок 2.11 Макет сторінки для перевірки інформації

Під номером 1 фон сторінки. Він виконує роль контейнера для блоків сторінки сайту. Фон сторінки планується робити в одному стилі для забезпечення єдиного стилістичного оформлення сайту. Використання єдиного фону для всіх сторінок дозволить створити єдиний цілісний дизайн вебсайту та покращує сприйняття інформації користувачем.

Під номером 2 буде знаходитись форма для перевірки інформації введеної користувачем. Даний блок є основним елементом сторінки та містить інформацію яку ввів користувач для бронювання квитків.

Під номером 3 назва блоку.

Під номером 4 назва музею.

Під номером 5 ім'я та прізвище написане англійською.

Під номером 6 час на який буде бронюватися квиток.

Під номером 7 дата на яку буде бронюватися квиток.

Під номером 8 номер групи в якій буде заброньовано квиток.

Під номером 9 буде кількість та тип квитків які планується забронювати.

Під номером 10 загальна вартість за квитки.

Під номером 11 кнопка бронювати.

Наступна сторінка це сторінка підтвердження бронювання, вона буде така сама як і для перевірки інформації тільки там вже буде кнопка, яка дозволить скачати в PDF форматі квиток, який забронював користувач.

РОЗДІЛ 3. СТВОРЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1. Розробка бази даних

Представлена структура бази даних описує інформаційну систему бронювання квитків для музеїв або інших туристичних/екскурсійних об'єктів. Основне призначення системи – забезпечення зручного процесу вибору локації, типу квитка, часу відвідування та створення бронювання користувачем.

Система побудована за реляційною моделлю даних та складається з декількох взаємопов'язаних таблиць.

Основні функції системи:

1. зберігання інформації про музеї та місця відвідування;
2. поділ місць за категоріями;
3. управління типами квитків;
4. контроль часу екскурсій;
5. обмеження максимальної кількості відвідувачів;
6. створення та облік бронювань;
7. зв'язок між квитками, музеями та групами відвідування.

Для проєктування структури бази даних системи бронювання квитків було використано ER-модель рис 3.1 (Entity-Relationship Model).

Дана модель дозволяє описати основні сутності системи, їх атрибути та зв'язки між ними. На основі ER-моделі була побудована реляційна структура бази даних, яка складається із взаємопов'язаних таблиць. ER-діаграма містить основні сутності системи: `booking`, `excursion_group`, `place`, `place_type` та `ticket_type`, а також первинні та зовнішні ключі, що забезпечують цілісність і коректність зв'язків між даними.

Використання ER-моделі дозволило логічно організувати структуру даних, уникнути дублювання інформації та забезпечити масштабованість системи бронювання.

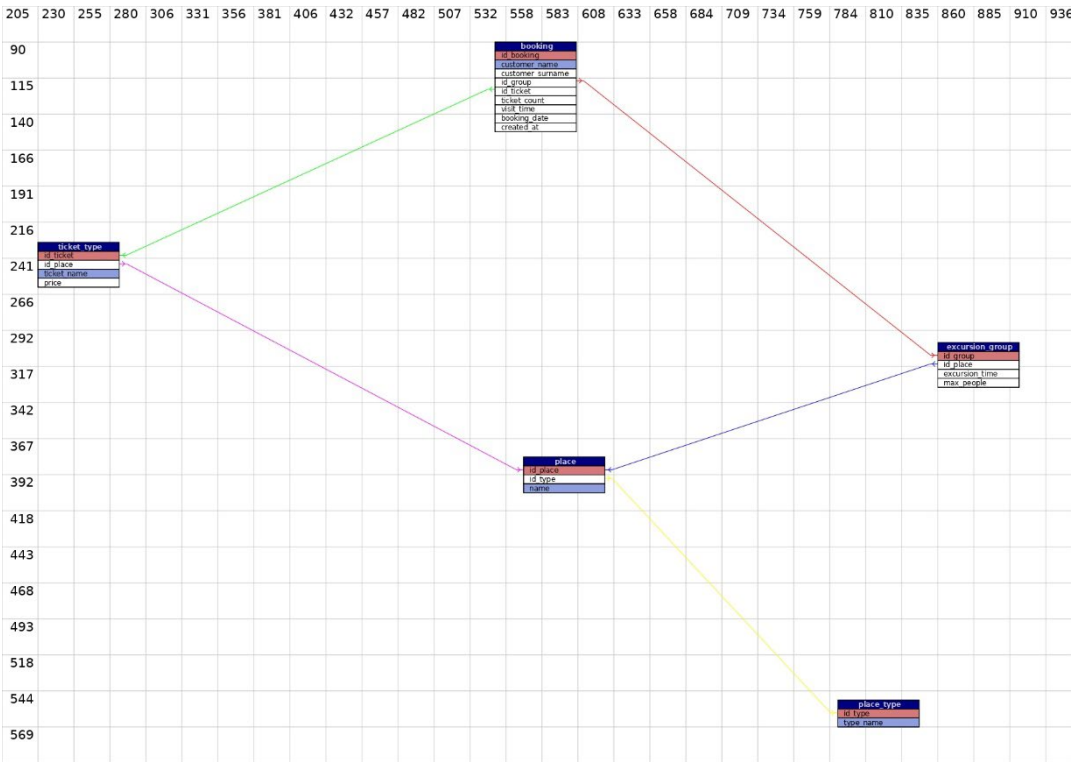


Рисунок 3.1 ER-модель бази даних

У системі присутні 5 основних таблиць:

- 1. booking
- 2. excursion_group
- 3. place
- 4. place_type
- 5. ticket_type

Таблиця booking є центральною таблицею системи. Вона зберігає інформацію про всі створені бронювання.

Таблиця 3.1 Таблиця booking

Поле	Тип	Призначення
id_booking	Int	Унікальний ідентифікатор бронювання

customer_name	Varchar	Ім'я користувача
customer_surname	Varchar	Прізвище користувача
id_group	Int	Посилання на групу екскурсії
id_ticket	Int	Посилання на тип квитка
ticket_count	Int	Кількість квитків
visit_time	Time	Обраний час відвідування
booking_date	Date	Дата відвідування
created_at	Timestamp	Дата та час створення бронювання

Таблиця booking (Таблиця 3.1) призначена для зберігання інформації про бронювання квитків користувачами на відвідування музею або екскурсії. Вона містить дані про користувача, тип квитка, кількість придбаних квитків, дату та час відвідування, а також дату створення бронювання. Кожен запис у таблиці є окремим бронюванням і має унікальний ідентифікатор. Поле `id\_group` використовується для зв'язку з таблицею груп екскурсій, а `id\_ticket` — для зв'язку з таблицею типів квитків. Поле `created\_at` автоматично фіксує дату та час створення запису, що дозволяє вести облік усіх бронювань у системі.

Таблиця booking забезпечує:

1. реєстрацію користувачів;
2. створення бронювання;
3. зв'язок із конкретною екскурсією;
4. зв'язок із типом квитка;
5. контроль кількості квитків;
6. можливість ведення статистики;
7. перевірку доступності місць.

Первинний ключ в таблиці booking є поле id_booking. Первинний ключ використовується для:

1. унікальної ідентифікації бронювання;
2. швидкого пошуку записів;
3. створення зв'язків між таблицями.

Зовнішні ключі таблиці booking є поля id_group та id_ticket.

Поле id_group посилається на таблицю excursion_group.

Функція:

1. визначає групу або часовий слот екскурсії;
2. дозволяє контролювати кількість місць.

Поле id_ticket посилається на таблицю ticket_type.

Функція:

1. визначає тип квитка;
2. дозволяє використовувати різні ціни.

Таблиця excursion_group відповідає за зберігання інформації про часові групи екскурсій. Кожен запис у таблиці представляє окремий часовий слот для проведення екскурсії у певному місці відвідування. Це дозволяє організовувати екскурсії за розкладом та контролювати кількість учасників у кожній групі.

Поля таблиці (Таблиця 3.2)

Таблиця 3.2 Таблиця excursion_group

Поле	Тип	Призначення
id_group	Int	Унікальний номер групи
id_place	Int	Посилання на місце відвідування
excursion_time	Time	Час проведення екскурсії
max_people	Int	Максимальна кількість людей

Таблиця забезпечує:

1. управління часовими слотами;
2. обмеження кількості відвідувачів;
3. контроль перевантаження музею;

4. планування екскурсій;
5. можливість створення декількох сеансів на день.

Первинний ключ таблиці `excursion_group` є поле `id_group`

Використовується для:

1. ідентифікації екскурсійної групи;
2. створення зв'язку з бронюванням.

Зовнішній ключ таблиці `excursion_group` є поле `id_place` і посилається він на таблицю `place`.

Функція:

1. визначає музей або локацію;
2. пов'язує часовий слот із конкретним місцем.

Таблиця `place` містить інформацію про місця відвідування.

Поля таблиці (Таблиця 3.3)

Таблиця 3.3 Таблиця `place`

Поле	Тип	Призначення
<code>id_place</code>	<code>Int</code>	Унікальний номер місця
<code>id_type</code>	<code>Int</code>	Посилання на тип місця
<code>name</code>	<code>Varchar</code>	Назва музею або локації

Функціонал таблиці:

1. зберігання списку музеїв;
2. групування за категоріями;
3. зв'язок із квитками;
4. зв'язок із часовими слотами.

Первинний ключ таблиці `place` є поле `id_place`.

Ключ дозволяє:

1. ідентифікувати музей;
2. зв'язувати інші таблиці з конкретним місцем.

Зовнішній ключ таблиці `place` є поле `id_type` він посилається на таблицю `place_type`.

Функція зовнішнього ключа полягає в визначенні категорії місця.

Таблиця place_type зберігає типи або категорії місць.

Поля таблиці (Таблиця 3.4)

Таблиця 3.4 Таблиця place_type

Поле	Тип	Призначення
id_type	Int	Унікальний номер типу
type_name	Varchar	Назва категорії

Таблиця дозволяє:

1. фільтрувати місця;
2. групувати музеї;
3. спрощувати навігацію;
4. створювати категорії на сайті.

Таблиця ticket_type містить інформацію про типи квитків.

Поля таблиці (Таблиця 3.5)

Таблиця 3.5 Таблиця ticket_type

Поле	Тип	Призначення
id_ticket	Int	Унікальний номер квитка
id_place	Int	Посилання на музей
ticket_name	Varchar	Назва квитка
price	Decimal	Вартість квитка

Таблиця забезпечує:

1. різні типи квитків;
2. різні ціни;
3. підтримку пільг;
4. гнучке управління тарифами.

Приклади типів квитків:

1. дорослий;
2. пільговий;
3. дитячий/молодь;

Зовнішній ключ таблиці ticket_type це поле id_place він посилається на таблицю place.

Функція зовнішнього ключа, визначає музей, для якого діє квиток.

3.2. Аналіз зв'язків між таблицями

1. place_type → place

Тип зв'язку:

Один до багатьох (1:N)

Один тип місця може містити багато музеїв.

Приклад:

- Категорія “музей” може містити:
- Muzeum Narodowe;
- Muzeum Archeologiczne;
- Muzeum Pyry.

2. place → ticket_type

Тип зв'язку:

Один до багатьох (1:N)

Один музей може мати багато типів квитків.

3. place → excursion_group

Тип зв'язку:

Один до багатьох (1:N)

Один музей може мати багато часових слотів.

4. excursion_group → booking

Тип зв'язку:

Один до багатьох (1:N)

Одна екскурсійна група може містити багато бронювань.

5. ticket_type → booking

Тип зв'язку:

Один до багатьох (1:N)

Один тип квитка може використовуватись у багатьох бронюваннях.

3.3. Розробка сайту

Розробка вебсайту системи бронювання квитків здійснювалася з метою створення зручного та сучасного сервісу для перегляду інформації про музеї, вибору квитків і оформлення бронювання в онлайн-режимі.

Під час створення вебсайту основна увага приділялася:

1. простоті використання;
2. зрозумілому інтерфейсу;
3. швидкій навігації;
4. адаптивному дизайну;
5. зручності оформлення бронювання.

Для реалізації клієнтської частини вебсайту були використані технології HTML, CSS та JavaScript.

HTML використовувався для створення структури сторінок вебсайту та розміщення основних елементів інтерфейсу, таких як:

1. навігаційне меню;
2. картки музеїв;
3. форми бронювання;
4. кнопки взаємодії;
5. інформаційні блоки.

CSS застосовувався для стилізації сторінок та створення сучасного дизайну. За допомогою CSS було реалізовано:

1. кольорове оформлення;
2. адаптивне розташування елементів;
3. оформлення кнопок і карток;
4. анімації та ефекти наведення;
5. структурування контенту.

JavaScript використовувався для реалізації динамічного функціоналу вебсайту. За допомогою JavaScript було реалізовано:

1. завантаження інформації про музеї;
2. обробку параметрів URL;

3. вибір типів квитків;
4. підрахунок вартості замовлення;
5. перевірку введених даних;
6. передачу інформації між сторінками;
7. генерацію електронного квитка.

Інформація про музеї зберігається у вигляді JavaScript-об'єкта, що містить:

1. назву музею;
2. короткий опис;
3. детальний опис;
4. зображення;
5. типи квитків;
6. вартість квитків.

На головній сторінці вебсайту (Рис. 3.2) розташовані основні категорії доступних місць для відвідування, а також навігаційні елементи системи.

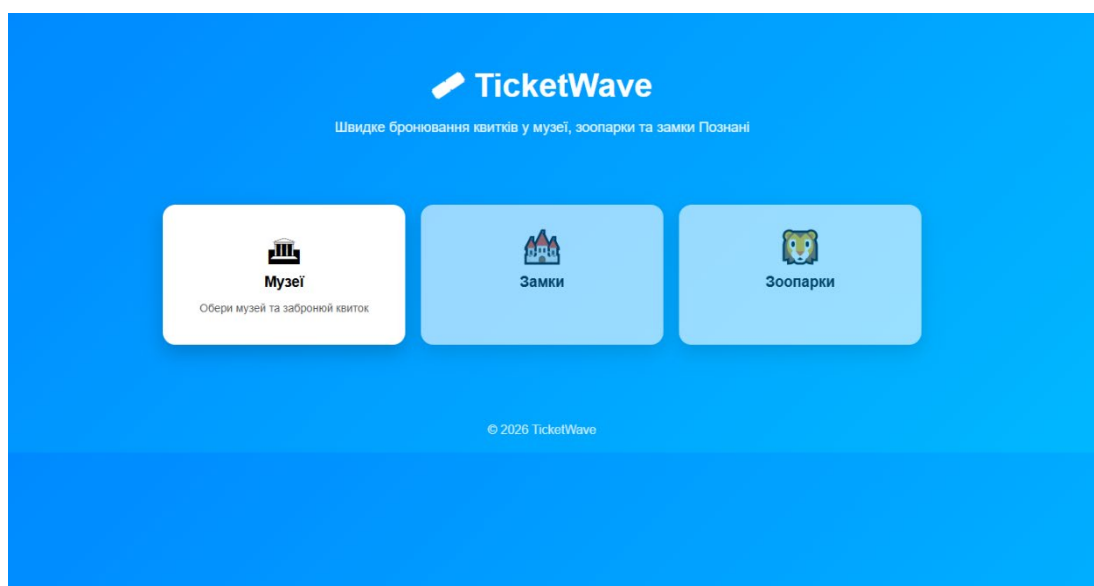


Рисунок 3.2 Головна сторінка вебсайту

Інтерфейс сторінки розроблено у сучасному стилі з використанням карток категорій та адаптивного дизайну.

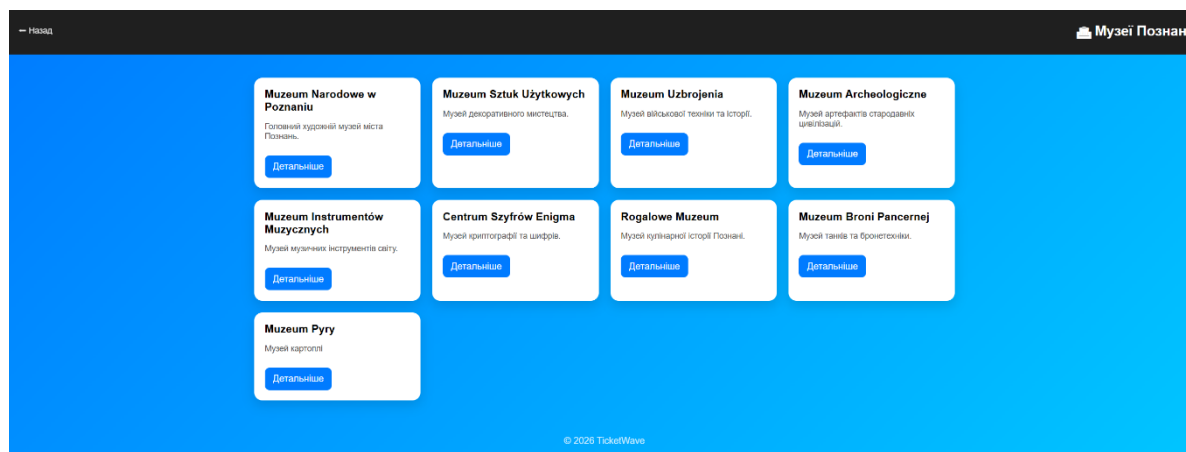


Рисунок 3.3 Сторінка вибору музеїв

На рисунку 3.3 зображено сторінку вибору музеїв. Дана сторінка призначена для перегляду доступних музеїв та переходу до детальної інформації про кожен об'єкт. У верхній частині сторінки розташована навігаційна панель, яка містить кнопку повернення на попередню сторінку та назву поточного розділу.

Основна частина сторінки реалізована у вигляді адаптивної сітки карток.

Кожна картка музею містить:

1. назву музею;
2. короткий опис;
3. кнопку переходу до детальної сторінки.

На сторінці представлені різні музеї Познані, зокрема художні, археологічні, військові та тематичні музеї. Для покращення зручності користування було реалізовано компактне розташування елементів, що дозволяє швидко переглядати доступні варіанти.

Дизайн сторінки виконаний у сучасному стилі з використанням градієнтного фону, карткового інтерфейсу та інтерактивних кнопок.

Кнопка «Детальніше» забезпечує перехід до сторінки конкретного музею, де користувач може переглянути розширену інформацію та перейти до оформлення бронювання квитків.

Для кожного музею реалізовано окрему сторінку з детальною інформацією (Рис 3.4).



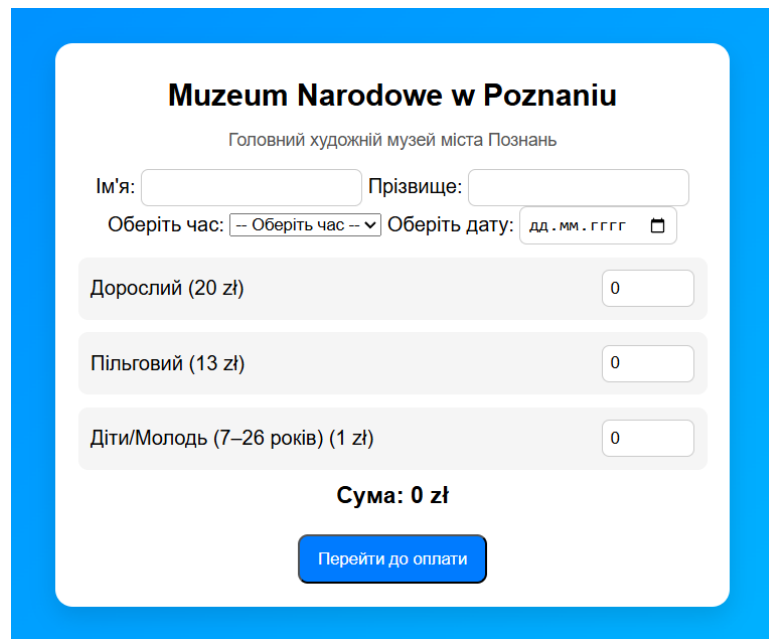
Рисунок 3.4 Сторінка одного з музеїв

На сторінці відображаються:

1. фотографія музею;
2. назва;
3. короткий опис;
4. розширена інформація;
5. доступні типи квитків;
6. кнопка переходу до бронювання.

Система бронювання (Рис 3.5) дозволяє користувачу:

1. обрати музей;
2. вибрати тип квитка;
3. вказати кількість квитків;
4. вибрати дату та час відвідування;
5. переглянути загальну вартість замовлення;
6. підтвердити бронювання.



The image shows a web form for booking tickets at the Muzeum Narodowe w Poznaniu. The form is titled 'Muzeum Narodowe w Poznaniu' and 'Головний художній музей міста Познань'. It includes input fields for 'Ім'я:' (Name) and 'Прізвище:' (Surname). Below these are dropdown menus for 'Оберіть час:' (Select time) and 'Оберіть дату:' (Select date) with a calendar icon. There are three rows for ticket types: 'Дорослий (20 zł)' (Adult 20 zł), 'Пільговий (13 zł)' (Reduced 13 zł), and 'Діти/Молодь (7–26 років) (1 zł)' (Children/Youth 7–26 years 1 zł). Each row has a numeric input field with '0' entered. The total is displayed as 'Сума: 0 zł'. A blue button at the bottom says 'Перейти до оплати' (Go to payment).

Рисунок 3.5 Сторінка системи бронювання

Для підвищення зручності користування було реалізовано адаптивний дизайн, який дозволяє коректно відображати вебсайт як на персональних комп'ютерах, так і на мобільних пристроях.

Під час розробки структури даних вебсайту було використано ER-модель (Entity-Relationship Model), яка дозволила описати основні сутності системи, їх атрибути та зв'язки між ними. На основі ER-моделі була створена реляційна структура бази даних для зберігання інформації про музеї, квитки, часові слоти та бронювання користувачів.

У результаті розробки було створено вебсайт, який забезпечує зручний процес онлайн-бронювання квитків, швидкий доступ до інформації про музеї та сучасний інтерфейс взаємодії з користувачем.

ВИСНОВОК

У процесі виконання кваліфікаційної роботи було проведено аналіз сучасних підходів до створення систем онлайн-бронювання квитків, досліджено особливості існуючих програмних рішень та визначено основні проблеми, які виникають під час організації електронного резервування культурно-розважальних заходів. Особливу увагу було приділено питанням зручності користування, швидкодії системи, безпеки передачі даних та адаптивності вебресурсів для різних типів пристроїв.

У результаті виконаної роботи було спроектовано та реалізовано вебсистему бронювання квитків для музеїв, замків та зоопарків міста Познань. Створена система дозволяє користувачам переглядати інформацію про доступні заклади, обирати дату та час відвідування, тип квитка, кількість місць, а також оформлювати бронювання з можливістю отримання електронного квитка у форматі PDF. Під час розробки було реалізовано структуру бази даних, яка забезпечує коректне зберігання інформації про заклади, типи квитків, екскурсійні групи та бронювання користувачів.

Під час роботи було використано сучасні вебтехнології HTML, CSS, JavaScript, PHP та MySQL, які дозволили створити функціональний та зрозумілий інтерфейс користувача. Також було враховано принципи адаптивного дизайну із забезпеченням коректної роботи вебсайту як на персональних комп'ютерах, так і на мобільних пристроях. Окрему увагу звернено у бік логічності структури сторінок, простоті навігації та мінімізації кількості дій, необхідних для оформлення бронювання.

Під час виконання кваліфікаційної роботи було підтверджено, що автоматизація процесу бронювання квитків значно спрощує взаємодію користувачів із культурними закладами, зменшує навантаження на каси та дозволяє швидко отримувати актуальну інформацію про доступність квитків. Розроблена система є особливо актуальною для українців, які перебувають у Польщі та потребують зрозумілого україномовного сервісу для планування культурного відпочинку та екскурсій.

Практична цінність роботи полягає у створенні готового вебпроєкту, який може бути використаний як основа для подальшого розвитку повноцінної системи онлайн-бронювання. У перспективі система може бути доповнена інтеграцією з платіжними сервісами, системою реєстрації користувачів, електронною поштою для автоматичного надсилання квитків, а також механізмами адміністрування закладів та статистики відвідувань.

Отже, поставлена мета кваліфікаційної роботи була досягнута, а всі основні завдання, пов'язані з аналізом, проєктуванням та створенням вебсистеми бронювання квитків, виконані у повному обсязі.

СПИСОК ДЖЕРЕЛ

1. Укрзалізниця ввела бронювання квитків через Інтернет [Електронний ресурс] // Finance.ua. – Режим доступу: Finance.ua
2. Historia firmy [Електронний ресурс] // PKP Intercity. – Режим доступу: PKP Intercity
3. Sabre (travel reservation system) [Електронний ресурс] // Wikipedia. – Режим доступу: Wikipedia – Sabre
4. Amadeus IT Group [Електронний ресурс] // Wikipedia. – Режим доступу: Wikipedia – Amadeus IT Group
5. Sublime Text [Електронний ресурс] // Wikipedia. – Режим доступу: Wikipedia – Sublime Text
6. Що таке PHP? [Електронний ресурс] // FreeHost. – Режим доступу: FreeHost – PHP
7. Що таке JavaScript і для чого він потрібен [Електронний ресурс] // GoIT. – Режим доступу: GoIT – JavaScript
8. Google Chrome [Електронний ресурс] // Wikipedia. – Режим доступу: Wikipedia – Google Chrome
9. HTML: HyperText Markup Language [Електронний ресурс] // Mozilla Developer Network. – Режим доступу: MDN – HTML
10. CSS: Cascading Style Sheets [Електронний ресурс] // Mozilla Developer Network. – Режим доступу: MDN – CSS
11. JavaScript Guide [Електронний ресурс] // Mozilla Developer Network. – Режим доступу: MDN – JavaScript Guide
12. PHP Manual [Електронний ресурс] // PHP Documentation Group. – Режим доступу: PHP Manual
13. MySQL Documentation [Електронний ресурс] // Oracle. – Режим доступу: MySQL Documentation
14. XAMPP Installers and Downloads [Електронний ресурс] // Apache Friends. – Режим доступу: XAMPP Official Website
15. Figma: the collaborative interface design tool [Електронний ресурс] // Figma. – Режим доступу: Figma Official Website
16. jsPDF – JavaScript PDF generation library [Електронний ресурс]. – Режим доступу: jsPDF Official Website

Додаток А. Код головної сторінки сайту

```

<!DOCTYPE html>
<html lang="uk">
<head>
<meta charset="UTF-8">
<title>TicketWave</title>
<link rel="stylesheet" href="style.css">
</head>
<body>
<header class="hero"> <!-- Верхній банер (шапка сайту) -->
  <h1>🎫 TicketWave</h1> <!-- Головний заголовок сайту -->
  <p>Швидке бронювання квитків у музеї, зоопарки та замки Познані</p> <!-- Підзаголовок з описом сервісу -->
</header>
<section class="menu"> <!-- Основний блок меню з картками -->
  <a href="museums.html" class="card"> <!-- Картка з посиланням на сторінку музеїв -->
    <div class="icon">🏛️</div>
    <h3>Музеї</h3>
    <p>Обери музей та забронюй квиток</p>
  </a>
  <div class="card disabled"> <!-- Неактивна картка (замки) -->
    <div class="icon">🏰</div>
    <h3>Замки</h3>
  </div>
  <div class="card disabled"> <!-- Неактивна картка (зоопарки) -->
    <div class="icon">🦁</div>
    <h3>Зоопарки</h3>
  </div>
</section>
<footer>© 2026 TicketWave</footer>
</body>
</html>

```

Додаток Б. Код сторінки музеїв

```

<!DOCTYPE html> <!-- Оголошення HTML5 документа -->
<html lang="uk"> <!-- Початок HTML сторінки, мова українська -->
<head> <!-- Початок блоку метаданих -->
<meta charset="UTF-8"> <!-- Кодування символів UTF-8 -->
<title>TicketWave | Музеї</title> <!-- Заголовок вкладки браузера -->
<link rel="stylesheet" href="style.css"> <!-- Підключення стилів -->
</head> <!-- Кінець head -->
<body> <!-- Початок видимого вмісту сторінки -->
<header class="topbar"> <!-- Верхня панель сайту -->
  <a href="index.html" class="back">⬅ Назад</a> <!-- Кнопка повернення на головну -->
  <h2>🏛 Музеї Познані</h2> <!-- Заголовок сторінки -->
</header> <!-- Кінець верхньої панелі -->
<main class="container"> <!-- Основний контейнер сторінки -->
<section class="grid"> <!-- Сітка з картками музеїв -->
<div class="card"> <!-- Картка музею -->
  <h3>Muzeum Narodowe w Poznaniu</h3> <!-- Назва музею -->
  <p>Головний художній музей міста Познань.</p> <!-- Опис музею -->
  <a href="museum.html?id=1" class="btn">Детальніше</a> <!-- Посилання на сторінку музею -->
</div> <!-- Кінець картки -->
<div class="card"> <!-- Картка музею -->
  <h3>Muzeum Sztuk Użytkowych</h3> <!-- Назва музею -->
  <p>Музей декоративного мистецтва.</p> <!-- Опис музею -->
  <a href="museum.html?id=2" class="btn">Детальніше</a> <!-- Посилання -->
</div> <!-- Кінець картки -->
<div class="card"> <!-- Картка музею -->
  <h3>Muzeum Uzbrojenia</h3> <!-- Назва музею -->
  <p>Музей військової техніки та історії.</p> <!-- Опис музею -->
  <a href="museum.html?id=3" class="btn">Детальніше</a> <!-- Посилання -->
</div> <!-- Кінець картки -->
<div class="card"> <!-- Картка музею -->
  <h3>Muzeum Archeologiczne</h3> <!-- Назва музею -->
  <p>Музей артефактів стародавніх цивілізацій.</p> <!-- Опис музею -->
  <a href="museum.html?id=4" class="btn">Детальніше</a> <!-- Посилання -->
</div> <!-- Кінець картки -->
<div class="card"> <!-- Картка музею -->
  <h3>Muzeum Instrumentów Muzycznych</h3> <!-- Назва музею -->
  <p>Музей музичних інструментів світу.</p> <!-- Опис музею -->
  <a href="museum.html?id=5" class="btn">Детальніше</a> <!-- Посилання -->
</div> <!-- Кінець картки -->
<div class="card"> <!-- Картка музею -->
  <h3>Centrum Szyfrów Enigma</h3> <!-- Назва музею -->
  <p>Музей криптографії та шифрів.</p> <!-- Опис музею -->
  <a href="museum.html?id=6" class="btn">Детальніше</a> <!-- Посилання -->
</div> <!-- Кінець картки -->
<div class="card"> <!-- Картка музею -->
  <h3>Rogalowe Muzeum</h3> <!-- Назва музею -->
  <p>Музей кулінарної історії Познані.</p> <!-- Опис музею -->
  <a href="museum.html?id=7" class="btn">Детальніше</a> <!-- Посилання -->
</div> <!-- Кінець картки -->
<div class="card"> <!-- Картка музею -->
  <h3>Muzeum Broni Pancernej</h3> <!-- Назва музею -->
  <p>Музей танків та бронетехніки.</p> <!-- Опис музею -->
  <a href="museum.html?id=8" class="btn">Детальніше</a> <!-- Посилання -->
</div> <!-- Кінець картки -->
<div class="card"> <!-- Картка музею -->
  <h3>Muzeum Pury</h3> <!-- Назва музею -->
  <p>Музей картоплі.</p> <!-- Опис музею -->
  <a href="museum.html?id=9" class="btn">Детальніше</a> <!-- Посилання -->
</div> <!-- Кінець картки -->
</section> <!-- Кінець сітки -->
</main> <!-- Кінець основного контейнера -->

```

```
<footer> <!-- Футер сторінки -->  
  © 2026 TicketWave <!-- Авторські права -->  
</footer> <!-- Кінець футера -->  
</body> <!-- Кінець body -->  
</html> <!-- Кінець HTML документа -->
```

Додаток В. Код сторінки музею

```

<!DOCTYPE html> <!-- Оголошення HTML5 документа -->
<html lang="uk"> <!-- Початок HTML сторінки, мова українська -->
<head> <!-- Початок блоку метаданих -->
<meta charset="UTF-8"> <!-- Кодування сторінки UTF-8 -->
<title>Музей</title> <!-- Заголовок вкладки браузера -->
<link rel="stylesheet" href="style.css"> <!-- Підключення CSS стилів -->
</head> <!-- Кінець head -->
<body> <!-- Початок видимого вмісту сторінки -->
<header class="topbar"> <!-- Верхня панель -->
  <a href="museums.html">⬅ Назад</a> <!-- Кнопка повернення до списку музеїв -->
  <h2>TicketWave</h2> <!-- Назва сайту в шапці -->
</header> <!-- Кінець шапки -->
<div id="content"></div> <!-- Контейнер, куди JS вставляє інформацію про музей -->
<script src="museums.js"></script> <!-- Підключення файлу з даними музеїв -->
<script> <!-- Початок JavaScript логіки -->
const id = new URLSearchParams(location.search).get("id");
<!-- Отримання id музею з URL (наприклад ?id=1) -->
const m = museums[id];
<!-- Пошук музею в об'єкті museums за отриманим id -->
if (!m) {
  <!-- Перевірка: якщо музей не знайдено -->
  document.getElementById("content").innerHTML = "<h2>✗ Не знайдено</h2>";
  <!-- Вивід повідомлення про помилку -->
} else {
  <!-- Якщо музей знайдено -->
  document.getElementById("content").innerHTML = `
    <!-- Вставка HTML-коду динамічно -->
    <div class="card big"> <!-- Велика картка музею -->
      <h1>${m.name}</h1> <!-- Назва музею з даних -->
      <p>${m.desc}</p> <!-- Короткий опис музею -->
      <div class="full-text">${m.fullDesc}</div> <!-- Опис музею -->
      
      <a class="btn" href="bron.html?id=${id}">🛡 Бронювати</a> <!-- Кнопка бронювання -->
    </div>
  `;
} <!-- Кінець умови -->
</script> <!-- Кінець JavaScript -->
</body> <!-- Кінець тіла сторінки -->
</html> <!-- Кінець HTML документа -->

```

Додаток Г. Код сторінки бронювання

```

<!DOCTYPE html>
<html lang="uk">
<head>
<meta charset="UTF-8">
<title>Бронювання</title>
<link rel="stylesheet" href="style.css">
</head>
<body>
<header class="topbar"> <!-- Верхня панель -->
  <a href="museums.html">← Назад</a> <!-- Кнопка повернення -->
  <h2>Бронювання</h2> <!-- Заголовок сторінки -->
</header>
<div class="card big" id="app"></div>
<!-- Основний контейнер, куди JS вставляє форму -->
<script src="museums.js"></script>
<!-- Підключення бази музеїв -->
<script> <!-- Початок JavaScript -->
const id = new URLSearchParams(location.search).get("id");
const m = museums[id];
let total = 0;
const app = document.getElementById("app");
if (!m) {
  app.innerHTML = "✖ Музей не знайдено";
} else {
  app.innerHTML = `
    <h2>${m.name}</h2>
    <p>${m.desc}</p>
    <div class="form">
      <label>Ім'я:</label>
      <input type="text" id="firstName">
      <label>Прізвище:</label>
      <input type="text" id="lastName">
    </div>
    <label>Оберіть час:</label>
    <select id="visitTime">
      <option value="">-- Оберіть час --</option>
      <option value="10:00">10:00</option>
      <option value="13:00">13:00</option>
      <option value="16:00">16:00</option>
    </select>
    <label>Оберіть дату:</label>
    <input type="date" id="visitDate">
    <div id="tickets"></div>
    <h3 id="sum">Сума: 0 zł</h3>
    <button class="btn" onclick="goToPayment()">
      Перейти до оплати
    </button>
  `;
  const box = document.getElementById("tickets");
  m.tickets.forEach(t => {
    box.innerHTML += `
      <div class="ticket">
        <span>${t.name} (${t.price} zł)</span>
        <input type="number" min="0" value="0"
          data-price="${t.price}"
          data-name="${t.name}">
      </div>
    `;
  });
  document.addEventListener("input", () => {
    total = 0;
  });
}

```

```

let count = 0;
document.querySelectorAll("input[type='number']")
  .forEach(i => {
    const val = Number(i.value) || 0;
    count += val;
    total += val * Number(i.dataset.price);
  });
if (count > 15) {
  alert("✗ Максимум 15 квитків на одну групу");
  document.activeElement.value = 0;
  return;
}
document.getElementById("sum").innerText =
  "Сума: " + total + " zł";
});
}
// ГОЛОВНА ФУНКЦІЯ
function goToPayment() {

  const firstName =
    document.getElementById("firstName").value.trim();

  const lastName =
    document.getElementById("lastName").value.trim();
  let time =
    document.getElementById("visitTime").value;
  if (time && time.length === 5) {
    time += ":00";
  }
  const date =
    document.getElementById("visitDate").value;
  if (!firstName || !lastName) {
    alert("Введіть ім'я та прізвище");
    return;
  }
  if (!date) {
    alert("Оберіть дату");
    return;
  }
  const selectedDate = new Date(date);
  if (selectedDate.getDay() === 0) {
    alert("✗ У неділю музей не працює");
    return;
  }
  let tickets = [];
  let count = 0;
  document.querySelectorAll("input[type='number']")
    .forEach(i => {
      if (Number(i.value) > 0) {
        tickets.push({
          name: i.dataset.name,
          count: Number(i.value)
        });
        count += Number(i.value);
      }
    });
  if (count > 15) {
    alert("✗ Максимум 15 квитків на групу");
    return;
  }
  const assignedGroup = Number(document.getElementById("visitTime").selectedIndex);

```

```
location.href =  
`payment.html?museum=${encodeURIComponent(m.name)}&total=${encodeURIComponent(total)}&tickets=${e  
ncodeURIComponent(tickets.map(t => `${t.name} x${t.count}`).join(",  
"))}&time=${encodeURIComponent(time)}&date=${encodeURIComponent(date)}&group=${encodeURIComponent  
ent(assignedGroup)}&firstName=${encodeURIComponent(firstName)}&lastName=${encodeURIComponent(last  
Name)}&id=${encodeURIComponent(id)}`;  
}  
</script>  
</body>  
</html>
```

Додаток Г Код сторінки перевірки даних

```

<!DOCTYPE html>
<html lang="uk">
<head>
<meta charset="UTF-8">
<title>Оплата</title>
<link rel="stylesheet" href="style.css">
</head>
<body>
<div class="card big" id="box"></div>
<script>
const p = new URLSearchParams(location.search);
const museum = p.get("museum");
const total = p.get("total");
const tickets = p.get("tickets");
const time = p.get("time");
const date = p.get("date");
const group = p.get("group");
const id = p.get("id");
const firstName =
  p.get("firstName");
const lastName =
  p.get("lastName");
if (!museum) {
  document.body.innerHTML =
    "✖ Нема даних";
} else {
  document.getElementById("box")
    .innerHTML = `
    <h2>Бронювання</h2>
    <p>
      <b>${museum}</b>
    </p>
    <p>
      👤 ${firstName} ${lastName}
    </p>
    <p>
      ⌚ ${time}
    </p>
    <p>
      📅 ${date}
    </p>
    <p>
      👥 Група №${group}
    </p>
    <p>
      🎫 ${tickets}
    </p>
    <h3>${total} zł</h3>
    <button
      class="btn"
      onclick="pay()"
    >
      Бронювати
    </button>
  `;
}
function pay() {
  const p = new URLSearchParams(location.search);
  const firstName = p.get("firstName");
  const lastName = p.get("lastName");
  const date = p.get("date");

```

```

const time = p.get("time");
const group = p.get("group");
const id = p.get("id");
const museum = p.get("museum");
const total = p.get("total");
const tickets = p.get("tickets");
console.log("PAY CLICKED");
if (!firstName || !lastName || !date || !id) {
    alert("Missing data");
    return;
}
let ticketData = [];
if (tickets) {
    tickets.split(",").forEach(t => {
        const parts = t.trim().split(" x");
        ticketData.push({
            name: parts[0],
            count: Number(parts[1] || 1)
        });
    });
}
const payload = {
    name: firstName,
    surname: lastName,
    date: date,
    time: time,
    group: Number(group),
    id_place: Number(id),
    tickets: ticketData
};
console.log("SENDING:", payload);
console.log(time);
fetch("save.php", {
    method: "POST",
    headers: {
        "Content-Type": "application/json"
    },
    body: JSON.stringify(payload)
})
.then(res => res.json())
.then(data => {
    console.log("SERVER:", data);
    if (data.success) {
        location.href =
`success.html?museum=${encodeURIComponent(museum)}&total=${encodeURIComponent(total)}&tickets=${en
codeURIComponent(tickets)}&time=${encodeURIComponent(time)}&date=${encodeURIComponent(date)}&gro
up=${encodeURIComponent(group)}&firstName=${encodeURIComponent(firstName)}&lastName=${en
codeURIComponent(lastName)}`;
    } else {
        alert("DB ERROR: " + data.error);
    }
})
.catch(err => {
    console.log(err);
    alert("Server error");
});
}
</script>
</body>
</html>

```

Додаток Д Код сторінки для отримання квитка

```

<!DOCTYPE html>
<html lang="uk">
<head>
<meta charset="UTF-8">
<title>Уcnix</title>
<script src="https://cdnjs.cloudflare.com/ajax/libs/jspdf/2.5.1/jspdf.umd.min.js"></script>
<link rel="stylesheet" href="style.css">
</head>
<body>
<div class="card big" id="out"></div>
<script>
const p = new URLSearchParams(location.search);
const museum = p.get("museum");
const total = p.get("total");
const tickets = p.get("tickets");
const time = p.get("time");
const date = p.get("date");
const group = p.get("group");
const firstName = p.get("firstName");
const lastName = p.get("lastName");
if (!museum) {document.body.innerHTML = "✗ Дані не знайдено";}
else {
  document.getElementById("out")
    .innerHTML = `
    <h1>✓ Заброньовано</h1>
    <p>
      🏛️ ${museum}
    </p>
    <p>
      👤 ${firstName} ${lastName}
    </p>
    <p>
      🕒 ${time}
    </p>
    <p>
      📅 ${date}
    </p>
    <p>
      👥 Група №${group}
    </p>
    <p>
      🎫 ${tickets}
    </p>
    <h3>
      💰 ${total} zł
    </h3>
    <button
      class="btn"
      onclick="pdf()"
    >
      Скачати PDF
    </button>
  `;
}
function pdf() {
  const { jsPDF } = window.jspdf;
  const doc = new jsPDF();
  doc.setFont("helvetica", "normal");
  const safeMuseum = museum;
  const ticketLines = tickets

```

```

.split(",")
.map(t => {
  const line = t.trim();
  return line
    .replace(/Дорослий/g, "Adult")
    .replace(/Дитячий/g, "Child")
    .replace(/Пільговий/g, "Discount")
    .replace(/Діти\Молодь \((7-26 років\)/g, "Youth (7-26)")
    .replace(/Діти до 7 років/g, "Children under 7");
});
doc.setFontSize(24);
doc.text("MUSEUM TICKET", 20, 25);
doc.line(20, 32, 190, 32);
doc.setFontSize(15);
doc.text("Museum:", 20, 50);
doc.text(safeMuseum, 70, 50);
doc.text("Visitor:", 20, 65);
doc.text(firstName + " " + lastName, 70, 65);
doc.text("Date:", 20, 80);
doc.text(date, 70, 80);
doc.text("Time:", 20, 95);
doc.text(time, 70, 95);
doc.text("Group:", 20, 110);
doc.text("No " + group, 70, 110);
doc.text("Tickets:", 20, 125);
let yTickets = 125;
ticketLines.forEach(line => {
  doc.text(line, 70, yTickets);
  yTickets += 8;
});
const y = yTickets + 10;
doc.text("Total:", 20, y);
doc.text(total + " zł", 70, y);
doc.setFontSize(11);
doc.text("Please arrive 15 minutes before the visit.", 20, y + 25);
doc.rect(10, 10, 190, y + 35);
doc.save("ticket.pdf");
}
</script>
</body>
</html>

```

Додаток Е Код сторінки для приєднання до бази даних

```
<?php
$host = "localhost";
$user = "root";
$password = "";
$database = "booking_system";
$conn = new mysqli($host, $user, $password, $database);
if ($conn->connect_error) {
    die("DB ERROR: " . $conn->connect_error);
}
$conn->set_charset("utf8");
?>
```

Додаток Є Код сторінки для збереження даних в базі даних

```

<?php
error_reporting(E_ALL);
ini_set('display_errors', 1);
require "connect.php";
header("Content-Type: application/json");
$data = json_decode(file_get_contents("php://input"), true);
if (!$data) {
    echo json_encode([
        "success" => false,
        "error" => "No data received"
    ]);
    exit;}
$name = $data["name"];
$surname = $data["surname"];
$date = $data["date"];
$time = $data["time"];
$groupIndex = $data["group"];
$id_place = $data["id_place"];
$tickets = $data["tickets"];
try {
    foreach ($tickets as $ticket) {
        $ticketName = $ticket["name"];
        $ticketCount = $ticket["count"];
        $stmt = $conn->prepare("
            SELECT id_ticket
            FROM ticket_type
            WHERE id_place = ? AND ticket_name = ?
            LIMIT 1
        ");
        $stmt->bind_param("is", $id_place, $ticketName);
        $stmt->execute();
        $res = $stmt->get_result();
        $ticketData = $res->fetch_assoc();
        if (!$ticketData) {
            continue;
        }
        $id_ticket = $ticketData["id_ticket"];
        $stmt2 = $conn->prepare("
            SELECT id_group
            FROM excursion_group
            WHERE id_place = ?
            ORDER BY id_group
            LIMIT 1 OFFSET ?
        ");
        $offset = $groupIndex - 1;
        $stmt2->bind_param("ii", $id_place, $offset);
        $stmt2->execute();
        $res2 = $stmt2->get_result();
        $groupData = $res2->fetch_assoc();
        if (!$groupData) {
            continue;
        }
        $id_group = $groupData["id_group"];
        $check = $conn->prepare("
        SELECT SUM(ticket_count) as total
        FROM booking
        WHERE id_group = ?
    ");
        $check->bind_param("i", $id_group);
        $check->execute();
        $res = $check->get_result();
    }
}

```

```

$row = $res->fetch_assoc();

$current = $row["total"] ?? 0;
$newTotal = $current + array_sum(array_column($tickets, "count"));
$groupTimes = [
    1 => "10:00",
    2 => "13:00",
    3 => "16:00"
];
$timeLabel = $groupTimes[$groupIndex] ?? "unknown";
if ($newTotal > 15) {
    echo json_encode([
        "success" => false,
        "error" => "На цей час кількість квитків що вам потрібна недоступна",
        "details" => [
            "group_time" => $timeLabel,
            "current_tickets" => $current,
            "requested_tickets" => array_sum(array_column($tickets, "count")),
            "max" => 15
        ]
    ]);
    exit;
}

$stmt3 = $conn->prepare("
    INSERT INTO booking
    (
        customer_name,
        customer_surname,
        id_group,
        id_ticket,
        ticket_count,
        visit_time,
        booking_date
    )
    VALUES (?, ?, ?, ?, ?, ?, ?)
");
$stmt3->bind_param(
    "ssiiiss",
    $name,
    $surname,
    $id_group,
    $id_ticket,
    $ticketCount,
    $time,
    $date
);
if (!$stmt3->execute()) {
    echo json_encode([
        "success" => false,
        "error" => $stmt3->error
    ]);
    exit;
}
}
echo json_encode([
    "success" => true
]);
} catch (Exception $e) {
    echo json_encode([
        "success" => false,
        "error" => $e->getMessage()
    ]);
}
}

```

Додаток Ж Код сторінки стилів сайту

```

body {
  margin: 0;
  font-family: Arial, sans-serif;
  background: linear-gradient(135deg, #007bff, #00c6ff);
  color: white;
}
.topbar {
  display: flex;
  justify-content: space-between;
  align-items: center;
  padding: 15px 20px;
  background: #1f1f1f;
  color: white;
}
.back {
  color: white;
  text-decoration: none;
  font-size: 14px;
  opacity: 0.9;
}
.back:hover {
  opacity: 1;
}
.hero {
  text-align: center;
  padding: 60px 20px 30px;
}

.hero h1 {
  font-size: 42px;
  margin-bottom: 10px;
}
.hero p {
  font-size: 18px;
  opacity: 0.9;
}
.menu {
  display: flex;
  justify-content: center;
  gap: 20px;
  padding: 40px;
  flex-wrap: wrap;
}
.menu .card {
  background: white;
  color: black;
  width: 260px;
  padding: 25px;
  border-radius: 16px;
  text-align: center;
  text-decoration: none;
  transition: 0.3s;
  box-shadow: 0 10px 25px rgba(0,0,0,0.2);
}
.menu .card:hover {
  transform: translateY(-10px);
}
.icon {
  font-size: 40px;
  margin-bottom: 10px;
}

```

```

.disabled {
  opacity: 0.6;
  cursor: not-allowed;
}
.container {
  max-width: 1200px;
  margin: 40px auto;
  padding: 0 20px;
}
.grid {
  display: grid;
  grid-template-columns: repeat(auto-fit, minmax(260px, 1fr));
  gap: 20px;
}
.card {
  background: white;
  color: black;
  padding: 18px;
  border-radius: 14px;
  box-shadow: 0 8px 20px rgba(0,0,0,0.12);
  transition: 0.25s;
}
.card:hover {
  transform: translateY(-6px);
}
.card h3 {
  margin-top: 0;
}

.card p {
  color: #555;
  font-size: 14px;
}
/* big card (museum page) */
.card.big {
  max-width: 500px;
  margin: 40px auto;
  text-align: center;
}
.btn {
  display: inline-block;
  margin-top: 10px;
  padding: 10px 14px;
  background: #007bff;
  color: white;
  border-radius: 8px;
  text-decoration: none;
  transition: 0.2s;
}
.btn:hover {
  background: #0056b3;
}
.ticket {
  display: flex;
  justify-content: space-between;
  align-items: center;
  margin: 10px 0;
  background: #f5f5f5;
  padding: 10px;
  border-radius: 8px;
  color: black;
}
.ticket input {

```

```
    width: 60px;
  }
  input {
    padding: 6px;
    border-radius: 6px;
    border: 1px solid #ccc;
  }
  footer {
    text-align: center;
    padding: 20px;
    color: white;
    opacity: 0.8;
    margin-top: 40px;
  }
  h1, h2, h3 {
    margin: 10px 0;
  }
  .museum-img{
    width:100%;
    height:300px;
    object-fit:cover;
    border-radius:15px;
    margin-bottom:20px;
  }
  .full-text{
    margin-top:15px;
    line-height:1.7;
    color:#333;
    font-size:18px;
    white-space:pre-line;
  }
}
```

Додаток 3 Файл реалізації JavaScript

```

const museums = {
  "1": { name:"Muzeum Narodowe w Poznaniu",
    desc:"Головний художній музей міста Познань",
    fullDesc:`
      Muzeum Narodowe w Poznaniu є одним із найстаріших
      та найважливіших музеїв Польщі.
      У музеї знаходяться:
      • польський живопис;
      • європейське мистецтво;
      • античні експонати;
      • сучасні виставки.
      Музей розташований у центрі Познані
      та щороку приймає тисячі туристів.
    `,
    image:"Muzeum Narodowe w Poznaniu.jpg",
    tickets:[{name:"Дорослий", price:20},{name:"Пільговий", price:13},{name:"Діти/Молодь (7–26 років)",
price:1}],},
  "2": { name:"Muzeum Sztuk Użytkowych",
    desc:"Музей декоративного мистецтва",
    fullDesc:`
      Muzeum Sztuk Użytkowych присвячений
      декоративному мистецтву та дизайну.
      У музеї представлені:
      • старовинні меблі;
      • кераміка;
      • годинники;
      • предмети інтер'єру;
      • вироби XVI–XIX століття.
      Музей знаходиться у Королівському замку
      в Познані та має оглядову вежу.
    `,
    image:"Muzeum Sztuk Użytkowych.jpg",
    tickets:[{name:"Дорослий", price:20},{name:"Пільговий", price:13},{name:"Діти/Молодь (7–26 років)",
price:1}],},
  "3": { name:"Muzeum Uzbrojenia",
    desc:"Музей військової техніки та історії",
    fullDesc:`
      Muzeum Uzbrojenia демонструє
      військову історію Польщі.
      У музеї можна побачити:
      • танки;
      • гармати;
      • бронетехніку;
      • військову форму;
      • історичну зброю.
      Музей особливо популярний
      серед любителів військової техніки.
    `,
    image:"Muzeum Uzbrojenia.jpg",
    tickets:[{name:"Дорослий", price:15},{name:"Пільговий", price:10},{name:"Діти до 7 років", price:0}],},
  "4": { name:"Muzeum Archeologiczne",
    desc:"Музей артефактів стародавніх цивілізацій",
    fullDesc:`
      Muzeum Archeologiczne представляє
      археологічні знахідки різних епох.
      Експозиція містить:
      • стародавні артефакти;
      • єгипетські експонати;
      • монети;
      • прикраси;
      • предмети побуту.
    `
  }
}

```

```

Музей дозволяє відвідувачам
дізнатися більше про розвиток цивілізацій.
,
image:"Muzeum Archeologiczne.jpg",
tickets:[{name:"Дорослий", price:10},{name:"Пільговий", price:6}],
"5": { name:"Muzeum Instrumentów Muzycznych",
desc:"Музей музичних інструментів світу",
fullDesc:`
Muzeum Instrumentów Muzycznych
є унікальним музеєм музики.
У колекції знаходяться:
• піаніно;
• скрипки;
• духові інструменти;
• народні інструменти;
• старовинні музичні експонати.
Відвідувачі можуть дізнатися
про історію музики різних народів світу.
,
image:"Muzeum Instrumentów Muzycznych.jfif",
tickets:[{name:"Дорослий", price:15},{name:"Пільговий", price:10}],
"6": { name:"Centrum Szyfrów Enigma",
desc:"Музей криптографії та шифрів",
fullDesc:`
Centrum Szyfrów Enigma присвячений
історії криптографії та шифрувальних машин.
У центрі можна побачити:
• копії машини Enigma;
• інтерактивні стенди;
• історичні документи;
• матеріали про криптографів.
Музей розповідає про внесок
польських математиків у розшифрування Enigma.
,
image:"Centrum Szyfrów Enigma.jpg",
tickets:[{name:"Дорослий", price:25}],
"7": { name:"Rogalowe Muzeum",
desc:"Інтерактивний музей познанських рогаликів Святого Марціна",
fullDesc:`
Rogalowe Muzeum Poznania —
один із найвідоміших туристичних музеїв міста.
Музей присвячений традиційному
рогалику Святого Марціна,
який є символом Познані.
Під час інтерактивного шоу відвідувачі можуть:
• дізнатися історію рогалика;
• побачити процес приготування;
• взяти участь у майстер-класі;
• скуштувати свіжу випічку;
• почути легенди та познанський діалект.
Музей поєднує гумор,
кулінарні традиції та живі вистави,
тому особливо популярний серед туристів і сімей.
,
image:"Rogalowe Muzeum.png",
tickets:[{name:"Дорослий", price:43},{name:"Діти/Молодь (7–26 років)", price:37}],
"8": { name:"Muzeum Broni Pancernej",
desc:"Музей танків та бронетехніки",
fullDesc:`
Muzeum Broni Pancernej
спеціалізується на бронетехніці.
У музеї представлені:
• танки;

```

```

    • бронемашини;
    • військові автомобілі;
    • історична техніка XX століття.
    Багато експонатів є
    справжніми зразками військової історії.
    ,
    ,
    image:"Muzeum Broni Pancernej.jpg",
    tickets:[{name:"Дорослий", price:32},{name:"Пільговий", price:21}],
    "9": { name:"Muzeum Rygu",
    desc:"Музей картоплі",
    fullDesc:`
        Muzeum Rygu присвячений
        історії та культурі картоплі у Польщі.
        У музеї можна:
        • дізнатися історію картоплі;
        • побачити тематичні експозиції;
        • взяти участь у кулінарних шоу;
        • скуштувати традиційні страви.
        Музей має інтерактивний формат
        та підходить для всієї родини.
        ,
        ,
        image:"Muzeum Rygu.jpg",
        tickets:[{name:"Дорослий", price:25}]
    };

```