МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра програмної інженерії

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему: **«РОЗРОБКА ВЕБ-СИСТЕМИ 3D-ВІЗУАЛІЗАЦІЇ ІСТОРИЧНОЇ БУДІВЛІ З ФУНКЦІЄЮ ВІРТУАЛЬНОЇ ЕКСКУРСІЇ НА ОСНОВІ THREE.JS»**

Виконав: студент 4 курсу, групи 6.1212-пі-1
спеціальності 121 інженерія програмного забезпечення
(шифр і назва спеціальності)
освітньої програми програмна інженерія
(назва освітньої програми)
І.О. Сідорук
(ініціали та прізвище)
Керівник доцент кафедри програмної інженерії, доцент, к.т.н. Мухін В.В.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)
Рецензент доцент кафедри комп'ютерних наук, доцент, к.т.н. Решевська К.С.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя – 2026

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математичний

Кафедра програмної інженерії

Рівень вищої освіти бакалавр

Спеціальність 121 інженерія програмного забезпечення
(шифр і назва)

Освітня програма програмна інженерія

ЗАТВЕРДЖУЮ
Завідувач кафедри програмної інженерії, к.ф.-м.н., доцент

_____ Лісняк А.О.
(підпис)

“ ____ ” _____ 2025 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Сідоруку Ігорю Олексійовичу
(прізвище, ім’я та по-батькові)

1. Тема роботи Розробка веб-системи 3D-візуалізації історичної будівлі з функцією віртуальної екскурсії на основі Three.js

керівник роботи Мухін Віталій Вікторович, к.т.н., доцент
(прізвище, ім’я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 02 » грудня 2025 року № 1741-с

2. Строк подання студентом роботи 01.06.2026 р.

3. Вихідні дані до роботи 1. Постановка задачі.
2. Перелік літератури.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Аналітичний огляд сфери 3D-візуалізації історичних будівель.
2. Проєктування вебсистеми 3D-візуалізації історичної будівлі.
3. Реалізація та тестування вебдодатку на основі Three.js.

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)
презентація за темою доповіді

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання 09.12.2025 р.

К А Л Е Н Д А Р Н И Й П Л А Н

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи.	19.12.2025	
2.	Збір вихідних даних.	14.01.2026	
3.	Обробка методичних та теоретичних	09.02.2026	
	джерел.		
4.	Розробка першого та другого розділу.	23.03.2026	
5.	Розробка третього розділу.	18.05.2026	
6.	Оформлення та нормоконтроль	25.05.2026	
	кваліфікаційної роботи бакалавра.		
7.	Захист кваліфікаційної роботи.	16.06.2026	

Студент _____ І.О. Сідорук
(підпис) (ініціали та прізвище)

Керівник роботи _____ В.В. Мухін
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ А.В. Столярова
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Кваліфікаційна робота бакалавра «Розробка веб-системи 3D-візуалізації історичної будівлі з функцією віртуальної екскурсії на основі Three.js»: 69 с., 21 рис., 23 джерела, 1 додаток.

ВЕБДОДАТОК, ВЕБСИСТЕМА, ВІРТУАЛЬНА ЕКСКУРСІЯ, ІНТЕРАКТИВНА НАВІГАЦІЯ, ІСТОРИЧНА БУДІВЛЯ, КОРИСТУВАЦЬКИЙ ІНТЕРФЕЙС, THREE.JS, WEBGL.

Об’єкт дослідження – процес створення веборієнтованої 3D-візуалізації архітектурних об’єктів.

Мета роботи – розробити вебсистему для 3D-візуалізації історичної будівлі з можливістю інтерактивного перегляду та віртуальної екскурсії на основі Three.js.

Методи дослідження – аналіз джерел, порівняльний аналіз технологій 3D-візуалізації, 3D-моделювання, вебпрограмування, проєктування інтерфейсу, тестування та оцінювання продуктивності вебдодатку.

У роботі розглянуто розробку вебсистеми для 3D-візуалізації історичної будівлі. Проаналізовано підходи до цифрового представлення архітектурних об’єктів, онлайн-екскурсій та використання WebGL і Three.js у вебсередовищі. У процесі роботи визначено вимоги до системи, розроблено структуру вебдодатку, підготовлено сценарій екскурсії та організовано роботу з 3D-моделями у форматі GLB. Реалізована система містить каталог історичних будівель, сторінки опису, 3D-переглядач, точки інтересу, екскурсійний режим і перемикання теми. Практичне значення роботи полягає у створенні вебдодатку для ознайомлення користувачів з історичними будівлями в інтерактивному форматі.

SUMMARY

Bachelor’s qualifying paper “Development of a Web-Based 3D Visualization System of a Historical Building with a Virtual Tour Feature Using Three.js”: 69 pages, 21 figures, 23 references, 1 supplement.

WEB APPLICATION, WEB SYSTEM, VIRTUAL TOUR, HISTORICAL BUILDING, INTERACTIVE NAVIGATION, USER INTERFACE, THREE.JS, WEBGL.

The object of the study is the process of creating web-oriented 3D visualization of architectural objects.

The aim of the study is to develop a web system for 3D visualization of a historical building with interactive viewing and a virtual tour function based on Three.js.

The methods of research include source analysis, comparative analysis of 3D visualization technologies, 3D modeling, web programming, user interface design, testing and performance evaluation of the web application.

The paper considers the development of a web system for 3D visualization of a historical building. Approaches to digital representation of architectural objects, online tours, and the use of WebGL and Three.js in a web environment are analyzed. During the work, system requirements were defined, the web application structure was developed, the tour scenario was prepared, and work with 3D models in GLB format was organized. The implemented system includes a catalog of historical buildings, description pages, a 3D viewer, points of interest, a tour mode, and theme switching. The practical value of the work lies in creating a web application for introducing users to historical buildings in an interactive format.

ЗМІСТ

Завдання на кваліфікаційну роботу 2
Реферат 4
Summary 5
Вступ 10
1 Аналітичний огляд сфери 3D-візуалізації історичних будівель 12
1.1 Історичні будівлі як об’єкти цифрової реконструкції 12
1.2 Віртуальні музеї та онлайн-екскурсії 13
1.3 Огляд сучасних технологій 3D-візуалізації у вебсередовищі 13
1.4 Порівняння можливостей Three.js з альтернативними технологіями 14
1.5 Аналіз існуючих проєктів 3D-турів 15
2 Проєктування вебсистеми 3D-візуалізації історичної будівлі 17
2.1 Вибір історичної будівлі та збір матеріалів 17
2.1.1 Критерії вибору історичних будівель 17
2.1.2 Підготовка текстових і графічних матеріалів 18
2.1.3 Підготовка матеріалів для 3D-візуалізації 18
2.2 Визначення функціональних і нефункціональних вимог 19
2.2.1 Функціональні вимоги до вебсистеми 19
2.2.2 Нефункціональні вимоги до вебсистеми 20
2.2.3 Вимоги до зручності користування 20
2.3 Архітектура вебсистеми 21
2.3.1 Загальна структура вебдодатку 21
2.3.2 Взаємодія основних сторінок системи 22
2.3.3 Організація даних для будівель та екскурсій 22
2.4 Структура вебдодатку та організація файлів 23
2.4.1 HTML-сторінки вебсистеми 23
2.4.2 CSS-стилі та адаптивне оформлення 24
2.4.3 JavaScript-файли та логіка роботи системи 24
2.4.4 Каталоги 3D-моделей, зображень і текстових даних 24

2.5 Розробка сценарію віртуальної екскурсії 25
2.5.1 Формування маршруту екскурсії 25
2.5.2 Опис точок інтересу 26
2.5.3 Підготовка інформаційного супроводу 26
2.6 Проєктування інтерфейсу користувача 27
2.6.1 Інтерфейс головної сторінки 27
2.6.2 Інтерфейс сторінки опису історичної будівлі 28
2.6.3 Інтерфейс сторінки 3D-візуалізації 28
2.6.4 Перемикання світлої та темної теми 28
2.7 Вибір форматів 3D-моделей і текстур 29
2.7.1 Особливості формату GLB/GLTF 29
2.7.2 Вимоги до текстур і матеріалів 30
2.7.3 Переваги використання GLB у вебсередовищі 30
2.8 Підготовка 3D-моделі історичної будівлі 30
2.8.1 Перевірка масштабу та положення моделі 31
2.8.2 Оптимізація моделі для браузера 31
2.8.3 Експорт і підключення моделі до вебсистеми 32
3 Реалізація та тестування вебдодатку на основі Three.js 33
3.1 Створення сцени, камери та освітлення 33
3.1.1 Ініціалізація сцени Three.js 34
3.1.2 Налаштування камери та рендерера 34
3.1.3 Налаштування освітлення сцени 35
3.2 Імпорт 3D-моделі у форматі GLB/GLTF 36
3.2.1 Завантаження моделі за допомогою GLTFLoader 37
3.2.2 Масштабування та позиціонування моделі 38
3.2.3 Перевірка коректності відображення моделі 39
3.3 Налаштування матеріалів і текстур 39
3.3.1 Перевірка матеріалів після імпорту 40
3.3.2 Перевірка відображення текстур після імпорту моделі 40
3.3.3 Оптимізація матеріалів для продуктивності 41
3.4 Реалізація навігації користувача 41

3.4.1 Керування камерою за допомогою OrbitControls 42
3.4.2 Обмеження руху камери 42
3.4.3 Адаптація керування до різних пристроїв..... 43
3.5 Реалізація точок інтересу 43
3.5.1 Структура даних для точок інтересу..... 44
3.5.2 Відображення маркерів у 3D-сцені 44
3.5.3 Показ інформаційної панелі..... 45
3.5.4 Збір і налаштування координат точок інтересу 46
3.6 Розробка режиму віртуальної екскурсії..... 47
3.6.1 Послідовність переходу між точками..... 48
3.6.2 Анімація переміщення камери 49
3.6.3 Відображення текстового супроводу екскурсії 50
3.6.4 Керування запуском і завершенням екскурсії 50
3.7 Реалізація користувацького інтерфейсу 51
3.7.1 Головна сторінка каталогу 52
3.7.2 Сторінка детального опису будівлі..... 53
3.7.3 Сторінка 3D-візуалізації..... 54
3.7.4 Реалізація перемикання теми інтерфейсу..... 55
3.8 Оптимізація продуктивності вебсцени 56
3.8.1 Оптимізація розміру 3D-моделі..... 56
3.8.2 Оптимізація рендерингу сцени..... 57
3.8.3 Зменшення навантаження на браузер 57
3.9 Адаптація вебдодатку під різні пристрої..... 58
3.9.1 Адаптація сторінок каталогу та опису..... 58
3.9.2 Адаптація 3D-переглядача 58
3.9.3 Перевірка роботи на різних розмірах екрана 59
3.10 Тестування роботи системи 60
3.10.1 Функціональне тестування 60
3.10.2 Тестування інтерфейсу користувача..... 60
3.10.3 Тестування продуктивності 61
3.10.4 Тестування адаптивності..... 61

3.11 Аналіз отриманих результатів 62
3.11.1 Оцінка відповідності системи вимогам 62
3.11.2 Переваги розробленої вебсистеми 63
3.11.3 Обмеження та напрями подальшого розвитку 63
Висновки 64
Перелік посилань 66
Додаток А Структура вебсистеми 3D-візуалізації історичних будівель 68

ВСТУП

Сучасний розвиток інформаційних технологій значно розширив можливості представлення культурної та історичної спадщини у цифровому середовищі. Якщо раніше ознайомлення з архітектурними пам’ятками здебільшого вимагало фізичної присутності відвідувача, то сьогодні вебтехнології дають змогу створювати інтерактивні 3D-візуалізації, віртуальні музеї та онлайн-екскурсії. Такі рішення є особливо актуальними для популяризації історичних будівель, збереження інформації про них та забезпечення доступу до культурних об’єктів незалежно від місця перебування користувача.

Актуальність теми зумовлена зростанням попиту на цифрові реконструкції, онлайн-екскурсії та інтерактивні освітні ресурси. 3D-візуалізація дозволяє не лише показати зовнішній вигляд історичної будівлі, а й організувати зручну взаємодію користувача з об’єктом: оглядати модель з різних боків, переходити між точками інтересу, отримувати текстову інформацію та проходити екскурсійний маршрут.

Одним із ефективних засобів реалізації 3D-графіки у браузері є технологія WebGL, яка забезпечує апаратне прискорення графіки без необхідності встановлення додаткових плагінів [3, 5]. Для спрощення роботи з WebGL широко використовується бібліотека Three.js [1]. Вона надає засоби для створення сцени, камери, освітлення, матеріалів, завантаження 3D-моделей, організації анімації та взаємодії користувача з об’єктами. Саме тому Three.js є доцільним вибором для розробки вебсистеми 3D-візуалізації історичної будівлі.

Мета роботи – розробити вебсистему 3D-візуалізації історичної будівлі з функцією віртуальної екскурсії на основі Three.js.

Для досягнення поставленої мети необхідно виконати такі завдання:

– проаналізувати сучасні підходи до 3D-візуалізації історичних будівель у вебсередовищі;

- розглянути технології WebGL, Three.js та альтернативні засоби створення 3D-візуалізацій;
- проаналізувати існуючі вебпроєкти, пов’язані з віртуальними музеями та 3D-турами;
- визначити функціональні та нефункціональні вимоги до вебсистеми;
- розробити структуру вебдодатку та сценарій віртуальної екскурсії;
- підготувати 3D-моделі історичних будівель для використання у браузері;
- реалізувати вебдодаток з каталогом будівель, сторінками опису та 3D-переглядачем;
- реалізувати точки інтересу, навігацію та екскурсійний режим;
- виконати тестування роботи системи та оцінити отримані результати.

Об’єкт дослідження – процес створення веборієнтованої 3D-візуалізації архітектурних об’єктів.

Предмет дослідження – методи та програмні засоби реалізації інтерактивної 3D-візуалізації історичної будівлі з функцією віртуальної екскурсії у вебсередовищі.

Методи дослідження: аналіз літературних та електронних джерел, порівняльний аналіз технологій, 3D-моделювання, вебпрограмування, проєктування інтерфейсу користувача, функціональне тестування та оцінювання продуктивності вебдодатку.

Практичне значення роботи полягає у створенні вебсистеми, яка може використовуватися для демонстрації історичних будівель, проведення віртуальних екскурсій, ознайомлення користувачів з архітектурними об’єктами та подальшого розширення за рахунок додавання нових моделей, описів і маршрутів.

1 АНАЛІТИЧНИЙ ОГЛЯД СФЕРИ 3D-ВІЗУАЛІЗАЦІЇ ІСТОРИЧНИХ БУДІВЕЛЬ

1.1 Історичні будівлі як об’єкти цифрової реконструкції

Історичні будівлі є важливими об’єктами культурної спадщини, оскільки вони відображають архітектурні особливості певної епохи, розвиток будівельних технологій, художні стилі та історичний контекст. Збереження таких об’єктів має значення не лише для науки та культури, а й для освіти, туризму та формування історичної пам’яті суспільства.

Цифрова реконструкція історичних будівель дає змогу створити віртуальне представлення об’єкта, яке може бути доступним широкому колу користувачів. На відміну від звичайних фотографій або текстових описів, 3D-модель дозволяє розглядати будівлю з різних ракурсів, оцінювати її просторову структуру, архітектурні деталі та загальне розташування елементів. Завдяки цьому користувач отримує більш повне уявлення про об’єкт.

Особливістю історичних будівель як об’єктів цифрової реконструкції є необхідність поєднання візуальної точності та технічної оптимізації. З одного боку, модель повинна передавати характерні архітектурні ознаки будівлі, фасади, форму, пропорції та деталі. З іншого боку, вебдодаток має працювати стабільно у браузері, тому 3D-модель не повинна бути надмірно складною або вимагати занадто великих ресурсів комп’ютера.

У межах даної роботи історична будівля розглядається не лише як окремий 3D-об’єкт, а як частина інтерактивної вебсистеми. Це означає, що користувач може не тільки переглядати модель, але й отримувати додаткову інформацію, переходити між точками інтересу та проходити віртуальну екскурсію. Такий підхід робить цифрову реконструкцію більш змістовною та корисною для користувача.

1.2 Віртуальні музеї та онлайн-екскурсії

Віртуальні музеї та онлайн-екскурсії є одним із напрямів використання цифрових технологій для представлення культурної спадщини [15, 20–22]. Вони дозволяють користувачам знайомитися з музейними експозиціями, архітектурними пам’ятками та історичними просторами без фізичного відвідування об’єкта. Це особливо важливо у випадках, коли будівля знаходиться в іншій країні, має обмежений доступ або потребує збереження від надмірного туристичного навантаження.

Онлайн-екскурсія може бути реалізована різними способами. Найпростішим варіантом є набір фотографій із текстовими описами. Більш складні системи використовують панорамні зображення, відео, 3D-моделі або інтерактивні сцени. Найбільш інформативним є підхід, при якому користувач може самостійно переміщуватися у віртуальному просторі або запускати заздалегідь підготовлений маршрут. Віртуальна екскурсія у 3D-середовищі має кілька переваг. По-перше, вона створює ефект присутності та дозволяє користувачу краще зрозуміти просторову структуру будівлі. По-друге, вона дає змогу поєднати візуальну інформацію з текстовими поясненнями, аудіо або відеоматеріалами. По-третє, така система може бути розширена новими точками інтересу, додатковими будівлями та інтерактивними елементами.

У розроблюваній вебсистемі віртуальна екскурсія реалізується через точки інтересу, інформаційні блоки та керування камерою. Користувач може переглядати 3D-модель, обирати потрібні об’єкти, переходити між частинами будівлі та отримувати описову інформацію. Це дозволяє поєднати функції 3D-переглядача та навчально-екскурсійного ресурсу.

1.3 Огляд сучасних технологій 3D-візуалізації у вебсередовищі

Для створення 3D-візуалізації у вебсередовищі використовуються різні технології та бібліотеки. Основою більшості сучасних рішень є WebGL –

програмний інтерфейс, який дозволяє відображати апаратно прискорену 3D-графіку безпосередньо у браузері [5]. Завдяки WebGL користувачу не потрібно встановлювати додаткові програми або плагіни, оскільки відображення графіки відбувається засобами браузера.

Однак безпосередня робота з WebGL є досить складною, оскільки потребує знання низькорівневих графічних принципів, шейдерів, буферів, матриць та інших технічних елементів. Тому для практичної розробки часто використовуються бібліотеки, які спрощують створення 3D-сцен. Однією з найпоширеніших таких бібліотек є Three.js [1].

Three.js надає готові інструменти для створення сцени, камери, джерел світла, матеріалів, геометрії, анімації та взаємодії користувача з 3D-об’єктами. Також бібліотека підтримує завантаження моделей у форматах GLTF і GLB, що є важливим для розробки вебдодатків із реалістичними 3D-об’єктами [1, 4].

Крім Three.js, існують інші засоби для створення 3D-графіки у браузері, зокрема Babylon.js та A-Frame. Babylon.js є потужним ігровим та візуалізаційним рушієм, який має багато вбудованих можливостей для фізики, матеріалів, анімацій та взаємодії [12]. A-Frame орієнтований на створення WebVR/WebXR-сцен і дозволяє описувати 3D-сцену за допомогою HTML-подібного синтаксису [11, 13].

Для даної роботи обрано Three.js, оскільки ця бібліотека поєднує достатню функціональність, гнучкість, велику кількість прикладів, підтримку GLB/GLTF-моделей і можливість повного контролю над структурою вебдодатку [2, 4].

1.4 Порівняння можливостей Three.js з альтернативними технологіями

Під час вибору інструментів для реалізації вебсистеми 3D-візуалізації доцільно порівняти Three.js з альтернативними технологіями. Основними критеріями порівняння є складність використання, можливості роботи з 3D-

моделями, підтримка інтерактивності, продуктивність, гнучкість і придатність для створення віртуальної екскурсії.

Three.js є універсальною бібліотекою для роботи з 3D-графікою у браузері [1]. Вона не нав’язує жорстку структуру проєкту, тому розробник може самостійно організовувати файли, логіку завантаження моделей, взаємодію з інтерфейсом та поведінку сцени. Це є перевагою для дипломного проєкту, оскільки дозволяє реалізувати власну архітектуру вебсистеми.

Babylon.js має ширший набір вбудованих можливостей і більше схожий на повноцінний 3D-рушій [12]. Він добре підходить для складних інтерактивних сцен, ігор або симуляцій. Проте для вебсистеми, основним завданням якої є демонстрація історичної будівлі, каталог об’єктів, сторінки опису та екскурсійний режим, використання Babylon.js може бути надмірним.

A-Frame зручний для швидкого створення VR-сцен і має простий декларативний підхід [13]. Проте він менш гнучкий для нестандартної логіки інтерфейсу, складного керування камерою та детальної інтеграції з власною структурою вебдодатку.

Таким чином, Three.js є оптимальним вибором для цієї роботи, оскільки забезпечує баланс між функціональністю, продуктивністю та гнучкістю. Бібліотека дозволяє створити інтерактивний 3D-переглядач, реалізувати точки інтересу, екскурсійний режим, роботу з GLB-моделями та інтеграцію з HTML/CSS/JavaScript-інтерфейсом [1, 4].

1.5 Аналіз існуючих проєктів 3D-турів

Існуючі проєкти 3D-турів і віртуальних музеїв демонструють різні підходи до представлення історичних та культурних об’єктів [19, 20]. Частина таких систем базується на панорамних фотографіях, де користувач переміщується між заздалегідь підготовленими точками огляду. Інші рішення використовують повноцінні 3D-моделі, які можна обертати, масштабувати та досліджувати у реальному часі [16, 23].

Перевагою панорамних турів є висока фотографічна реалістичність і простота створення маршруту. Проте такий підхід має обмежену інтерактивність, оскільки користувач не може вільно змінювати простір або розглядати об’єкт з довільного ракурсу. У свою чергу, 3D-моделі забезпечують більшу свободу взаємодії, але потребують підготовки оптимізованих моделей, текстур і налаштування сцени.

Аналіз існуючих рішень показує, що для зручності користувача важливо поєднувати декілька елементів: зрозумілий інтерфейс, якісний опис об’єкта, інтерактивні точки інтересу, можливість керування камерою та стабільну роботу на різних пристроях. Саме такі принципи були враховані під час розробки вебсистеми.

У розроблюваному проєкті реалізовано каталог історичних будівель, сторінки з описами, 3D-переглядач та екскурсійний режим. Такий підхід дозволяє користувачу спочатку ознайомитися з короткою інформацією про будівлю, а потім перейти до інтерактивної 3D-візуалізації.

2 ПРОЄКТУВАННЯ ВЕБСИСТЕМИ 3D-ВІЗУАЛІЗАЦІЇ ІСТОРИЧНОЇ БУДІВЛІ

2.1 Вибір історичної будівлі та збір матеріалів

Вибір історичної будівлі є одним із перших етапів розробки вебсистеми 3D-візуалізації. Від правильного вибору об’єкта залежить змістове наповнення системи, складність підготовки 3D-моделі, кількість інформаційних точок та загальна якість віртуальної екскурсії.

У межах роботи передбачено використання історичних будівель як основних об’єктів демонстрації. Для кожної будівлі необхідно підготувати текстовий опис, зображення, 3D-модель та інформацію для точок інтересу. Такий підхід дозволяє створити не просто окремий 3D-переглядач, а повноцінну вебсистему з каталогом об’єктів і можливістю розширення.

2.1.1 Критерії вибору історичних будівель

Під час вибору історичних будівель для вебсистеми враховувались декілька критеріїв. Насамперед об’єкт повинен мати історичну або архітектурну цінність, оскільки основна мета системи полягає у візуальному представленні культурної спадщини. Також важливо, щоб будівля мала виразний зовнішній вигляд, характерні архітектурні елементи та була цікавою для користувача з точки зору віртуального огляду.

Другим важливим критерієм є наявність матеріалів для опису будівлі. Для створення змістовної сторінки об’єкта необхідні короткі історичні відомості, опис архітектурних особливостей, фотографії або інші візуальні матеріали. Без таких даних 3D-візуалізація буде менш інформативною.

Третім критерієм є можливість використання або підготовки 3D-моделі. Модель повинна бути придатною для відображення у браузері, мати

прийнятний розмір файлу та не створювати надмірного навантаження на систему. Тому під час вибору об’єктів враховувалась не лише їх історична цінність, а й технічна можливість реалізації у вебсередовищі.

2.1.2 Підготовка текстових і графічних матеріалів

Для кожної історичної будівлі необхідно підготувати текстові та графічні матеріали. Текстові матеріали використовуються для сторінки детального опису, інформаційних блоків та точок інтересу у режимі віртуальної екскурсії. Вони повинні бути зрозумілими, стислими та відповідати тематиці конкретного об’єкта.

Графічні матеріали використовуються для представлення будівель у каталозі та на сторінках опису. До них можуть належати фотографії, зображення фасаду, загальні види будівлі або ілюстрації, що допомагають користувачу швидко зрозуміти, який об’єкт він обирає. Зображення також впливають на візуальну привабливість інтерфейсу.

Підготовка матеріалів передбачає їх відбір, структурування та розміщення у відповідних каталогах проєкту. Це дозволяє надалі легко підключати інформацію до HTML-сторінок і підтримувати зрозумілу структуру вебдодатку.

2.1.3 Підготовка матеріалів для 3D-візуалізації

Матеріали для 3D-візуалізації включають 3D-модель будівлі, текстури, дані про положення камери, точки інтересу та описові тексти для екскурсії. Вони повинні бути підготовлені таким чином, щоб забезпечити коректне завантаження та відображення об’єкта у браузері.

Основним форматом для 3D-моделі обрано GLB, оскільки він дозволяє зберігати геометрію, матеріали та текстури в одному файлі [4]. Це спрощує

підключення моделі до Three.js і зменшує кількість помилок, пов’язаних із шляхами до окремих ресурсів [1].

Окремо готуються дані для віртуальної екскурсії. До них належать назви точок інтересу, текстові пояснення, координати маркерів, положення камери та напрямок огляду. Завдяки такій структурі екскурсійний режим можна редагувати без повної зміни програмного коду.

2.2 Визначення функціональних і нефункціональних вимог

Перед початком реалізації вебсистеми необхідно визначити її вимоги. Вимоги описують, які можливості повинна мати система, як вона має працювати та яким очікуванням користувача повинна відповідати.

У даній роботі вимоги поділяються на функціональні, нефункціональні та вимоги до зручності користування. Такий поділ дозволяє окремо розглянути логіку роботи системи, якісні характеристики вебдодатку та особливості взаємодії користувача з інтерфейсом.

2.2.1 Функціональні вимоги до вебсистеми

Функціональні вимоги визначають основні можливості, які має надавати вебсистема користувачу. Насамперед система повинна містити головну сторінку з каталогом історичних будівель. Користувач має мати змогу переглядати доступні об’єкти, обирати конкретну будівлю та переходити до її детального опису.

Другою важливою функцією є сторінка опису історичної будівлі. На ній повинна розміщуватися інформація про об’єкт, зображення та кнопка переходу до 3D-візуалізації. Такий підхід дозволяє користувачу спочатку ознайомитися з будівлею, а потім перейти до інтерактивного перегляду.

Третьою функціональною вимогою є наявність 3D-переглядача. Він повинен забезпечувати завантаження GLB-моделі, відображення сцени, керування камерою, роботу точок інтересу та запуск режиму віртуальної екскурсії. Також система повинна підтримувати перемикання теми інтерфейсу та коректне відображення даних на різних сторінках.

2.2.2 Нефункціональні вимоги до вебсистеми

Нефункціональні вимоги описують якість роботи вебсистеми. Однією з головних вимог є продуктивність. Оскільки 3D-сцена виконується у браузері, система повинна забезпечувати стабільне відображення моделі, плавну навігацію та прийнятний час завантаження.

Також важливою вимогою є сумісність із сучасними браузерами. Вебдодаток має працювати без встановлення додаткових програм або плагінів, використовуючи стандартні можливості браузера та технологію WebGL. Це робить систему доступнішою для користувачів.

До нефункціональних вимог також належать підтримуваність і можливість розширення. Структура проєкту повинна бути організована таким чином, щоб у майбутньому можна було додавати нові будівлі, нові 3D-моделі, додаткові точки інтересу та нові сценарії екскурсій без повної переробки системи.

2.2.3 Вимоги до зручності користування

Зручність користування є важливою характеристикою вебсистеми, оскільки користувач повинен швидко розуміти, як переглядати каталог, відкривати сторінки будівель і запускати 3D-візуалізацію. Інтерфейс не повинен бути перевантаженим зайвими елементами.

На головній сторінці користувач має одразу бачити перелік історичних будівель. Картки об’єктів повинні бути зрозумілими та містити основну інформацію: назву, зображення і короткий опис. На сторінці будівлі кнопка переходу до 3D-візуалізації повинна бути помітною.

У 3D-переглядачі елементи керування повинні бути розташовані так, щоб не заважати огляду моделі. Інформаційна панель точок інтересу має відображати текст у зручному форматі. Також важливо, щоб користувач міг легко повернутися до попередніх сторінок або змінити режим перегляду.

2.3 Архітектура вебсистеми

Архітектура вебсистеми визначає загальну структуру додатку, взаємодію його сторінок, спосіб зберігання даних і логіку переходів між основними частинами системи. У межах роботи обрано клієнтську архітектуру, де основна логіка реалізується засобами HTML, CSS і JavaScript.

Такий підхід є зручним для дипломного проєкту, оскільки система може запускатися локально через браузер або простий локальний сервер. Це спрощує демонстрацію роботи та не потребує складного серверного розгортання.

2.3.1 Загальна структура вебдодатку

Загальна структура вебдодатку складається з декількох основних частин. Перша частина – головна сторінка, яка виконує роль каталогу історичних будівель. Вона дозволяє користувачу обрати об’єкт для перегляду.

Друга частина – сторінка детального опису будівлі. Вона містить розширену інформацію про об’єкт, зображення та елемент переходу до 3D-візуалізації. Ця сторінка виконує інформаційну функцію і готує користувача до перегляду моделі.

Третя частина – сторінка 3D-візуалізації. Саме на ній реалізується інтерактивна сцена Three.js, завантаження моделі, керування камерою, точки інтересу та екскурсійний режим. Таким чином, вебдодаток поєднує інформаційний сайт і 3D-переглядач.

2.3.2 Взаємодія основних сторінок системи

Основні сторінки системи взаємодіють між собою через переходи користувача та передачу інформації про обрану будівлю. На головній сторінці користувач обирає об’єкт, після чого відкривається сторінка його опису. З цієї сторінки користувач може перейти до 3D-переглядача.

Логіка переходів побудована таким чином, щоб користувач рухався від загального огляду до детального. Спочатку він бачить каталог, потім читає інформацію про конкретну будівлю, а після цього запускає інтерактивну 3D-сцену. Такий сценарій є зрозумілим і відповідає звичайній логіці користування вебресурсами.

Для зручності користувача передбачено можливість повернення до попередніх сторінок. Це дозволяє переглядати різні будівлі, порівнювати їх та запускати 3D-візуалізацію для потрібного об’єкта.

2.3.3 Організація даних для будівель та екскурсій

Дані для будівель і віртуальних екскурсій доцільно зберігати окремо від основної програмної логіки. Такий підхід спрощує редагування інформації та дозволяє додавати нові об’єкти без значної зміни коду.

Для кожної будівлі можуть зберігатися назва, короткий опис, повний опис, шлях до зображень, шлях до 3D-моделі та набір точок інтересу. Дані для екскурсії містять назви точок, текстові пояснення, координати маркерів, положення камери та напрямок огляду.

Окрема організація даних робить систему більш гнучкою. Наприклад, якщо необхідно змінити текст опису або додати нову точку інтересу, це можна зробити у відповідному файлі даних без повного переписування логіки 3D-переглядача.

2.4 Структура вебдодатку та організація файлів

Структура файлів вебдодатку має бути зрозумілою та логічною. Це полегшує розробку, тестування, редагування та подальший розвиток системи. У проєкті окремо виділяються файли сторінок, стилів, скриптів, моделей, зображень і текстових даних.

Такий підхід дозволяє уникнути змішування різних частин системи в одному файлі. Кожна група файлів виконує свою роль і відповідає за окрему частину функціональності вебдодатку.

2.4.1 HTML-сторінки вебсистеми

HTML-сторінки формують основну структуру вебсистеми [7]. Головна сторінка використовується для відображення каталогу історичних будівель. Вона містить основні блоки інтерфейсу, картки об’єктів та елементи керування.

Сторінка опису будівлі призначена для відображення більш детальної інформації про обраний об’єкт. На ній розміщуються зображення, текстовий опис і кнопка переходу до 3D-візуалізації. Ця сторінка є проміжною між каталогом і 3D-переглядачем.

Сторінка 3D-візуалізації містить область для відображення сцени Three.js і елементи керування екскурсійним режимом. Саме на цій сторінці користувач взаємодіє з 3D-моделлю історичної будівлі.

2.4.2 CSS-стилі та адаптивне оформлення

CSS-файли відповідають за зовнішній вигляд вебдодатку [8]. У них визначаються кольори, шрифти, розміри блоків, розташування елементів, оформлення карток, кнопок, панелей і режимів теми. Адаптивне оформлення є важливою частиною інтерфейсу [8]. Система повинна коректно відображатися на різних розмірах екрана, зокрема на комп’ютерах, ноутбуках, планшетах і смартфонах. Для цього використовуються гнучкі сітки, відносні розміри та медіазапити. Окремо реалізується підтримка світлої та темної теми. Це дозволяє користувачу змінювати вигляд інтерфейсу відповідно до власних уподобань і умов використання.

2.4.3 JavaScript-файли та логіка роботи системи

JavaScript-файли відповідають за динамічну логіку роботи вебсистеми [6]. До їх функцій належить завантаження даних про будівлі, обробка дій користувача, перемикання сторінок, зміна теми інтерфейсу та запуск 3D-візуалізації. Окрема частина JavaScript-коду пов’язана з Three.js. Вона відповідає за створення сцени, камери, рендерера, освітлення, завантаження GLB-моделі, керування камерою та відображення точок інтересу. Також у цьому коді реалізується екскурсійний режим.

Розділення логіки на окремі файли робить проєкт більш зручним для підтримки. Якщо потрібно змінити лише інтерфейс або лише поведінку 3D-сцени, це можна зробити у відповідному файлі, не змінюючи всю систему.

2.4.4 Каталоги 3D-моделей, зображень і текстових даних

Для зберігання ресурсів у проєкті використовуються окремі каталоги. 3D-моделі розміщуються в каталозі моделей, зображення – у каталозі

графічних матеріалів, а текстові дані та сценарії екскурсій – в окремих файлах даних.

Такий спосіб організації дозволяє швидко знаходити потрібні ресурси та зменшує ризик плутанини у структурі проєкту. Наприклад, якщо потрібно замінити зображення будівлі або оновити GLB-модель, достатньо змінити відповідний файл у потрібному каталозі.

Окреме зберігання текстових даних також спрощує редагування описів. Це важливо для дипломного проєкту, оскільки зміст екскурсії або опис будівлі може уточнюватися під час підготовки роботи.

2.5 Розробка сценарію віртуальної екскурсії

Сценарій віртуальної екскурсії визначає, як саме користувач буде знайомитися з історичною будівлею у 3D-просторі. Він складається з послідовності точок інтересу, описових текстів і положень камери.

Правильно побудований сценарій дозволяє зробити перегляд моделі більш змістовним. Користувач не просто обертає 3D-об’єкт, а послідовно переходить між важливими частинами будівлі та отримує пояснення.

2.5.1 Формування маршруту екскурсії

Маршрут віртуальної екскурсії формується на основі логічної послідовності перегляду будівлі. Зазвичай екскурсія починається із загального вигляду об’єкта, після чого користувач переходить до окремих архітектурних елементів, фасаду, входу, внутрішніх частин або інших важливих зон.

Для кожної точки маршруту визначається положення камери та напрямок огляду. Це потрібно для того, щоб користувач одразу бачив потрібну частину будівлі. Такий підхід робить екскурсію керованою і зрозумілою.

Маршрут не повинен бути занадто довгим або перевантаженим. Оптимальним є такий сценарій, у якому представлені основні частини будівлі, але користувач не втрачає увагу через надмірну кількість переходів.

2.5.2 Опис точок інтересу

Точки інтересу є основними інформаційними елементами віртуальної екскурсії. Кожна точка відповідає певній частині будівлі або важливому архітектурному елементу. Наприклад, це може бути фасад, вхід, башта, внутрішній простір, декоративна деталь або оглядова позиція.

Для кожної точки інтересу задається назва та короткий опис. Назва повинна бути зрозумілою, а опис – достатньо інформативним, щоб пояснити користувачу значення відповідного елемента. При цьому текст не повинен бути надто великим, оскільки він відображається в інтерфейсі 3D-переглядача.

Опис точок інтересу дозволяє поєднати візуальну та інформаційну складові системи. Завдяки цьому користувач не лише бачить модель, а й отримує додаткові відомості про історичну будівлю.

2.5.3 Підготовка інформаційного супроводу

Інформаційний супровід віртуальної екскурсії включає текстові пояснення, які відображаються під час переходу між точками інтересу. Його завдання полягає в тому, щоб надати користувачу коротку та зрозумілу інформацію про об’єкт.

Під час підготовки текстів необхідно враховувати формат відображення в інтерфейсі. Довгі фрагменти тексту можуть бути незручними для читання, тому описи мають бути стислими, логічними та поділеними за окремими точками.

У майбутньому інформаційний супровід може бути розширений аудіо, відео або додатковими ілюстраціями. У межах даної роботи основним способом супроводу є текстовий опис, який відображається у відповідній інформаційній панелі.

2.6 Проєктування інтерфейсу користувача

Інтерфейс користувача є важливою складовою вебсистеми, оскільки саме через нього користувач взаємодіє з каталогом будівель, сторінками опису та 3D-переглядачем.

Інтерфейс повинен бути простим, зрозумілим і не перевантаженим зайвими елементами.

Під час проєктування інтерфейсу враховувались основні сценарії використання системи: вибір будівлі, перегляд опису, запуск 3D-сцени, взаємодія з точками інтересу та проходження віртуальної екскурсії.

2.6.1 Інтерфейс головної сторінки

Головна сторінка виконує роль каталогу історичних будівель. Її основним завданням є надати користувачу швидкий огляд доступних об’єктів і можливість обрати потрібну будівлю для подальшого перегляду.

Кожна будівля представлена у вигляді картки. Картка містить зображення, назву та короткий опис. Такий формат є зручним, оскільки користувач може швидко порівняти об’єкти та перейти до того, який його зацікавив.

Для покращення зручності головна сторінка може містити фільтри, категорії або інші елементи навігації. Це особливо корисно, якщо у системі буде багато історичних будівель.

2.6.2 Інтерфейс сторінки опису історичної будівлі

Сторінка опису історичної будівлі призначена для більш детального ознайомлення з об’єктом. Вона містить розширений текстовий опис, зображення та кнопку переходу до 3D-візуалізації.

На цій сторінці користувач отримує основну інформацію про будівлю перед тим, як перейти до 3D-перегляду. Це дозволяє краще зрозуміти історичний і архітектурний контекст об’єкта.

Кнопка переходу до 3D-візуалізації повинна бути помітною та зрозумілою. Вона є головним елементом, який пов’язує інформаційну частину вебсистеми з інтерактивною 3D-сценою.

2.6.3 Інтерфейс сторінки 3D-візуалізації

Сторінка 3D-візуалізації є основною інтерактивною частиною системи. Вона містить область відображення 3D-сцени, елементи керування камерою, кнопки запуску екскурсії та інформаційну панель.

Інтерфейс цієї сторінки повинен бути компактним, оскільки основна увага користувача має бути зосереджена на 3D-моделі. Елементи керування не повинні перекривати важливі частини сцени або заважати огляду будівлі.

Інформаційна панель використовується для відображення опису поточної точки інтересу. Вона допомагає користувачу зрозуміти, яку частину будівлі він переглядає і чому вона важлива.

2.6.4 Перемикання світлої та темної теми

Перемикання світлої та темної теми є додатковою функцією інтерфейсу, яка підвищує комфорт користування вебсистемою. Світла тема зручна для денного перегляду, а темна – для роботи в умовах слабкого освітлення.

Реалізація перемикання теми дозволяє змінювати кольори фону, тексту, кнопок і панелей без зміни основної структури сторінок. Це робить інтерфейс більш сучасним і гнучким.

Крім практичного значення, ця функція демонструє можливість динамічної зміни оформлення вебдодатку за допомогою JavaScript і CSS [6, 8].

2.7 Вибір форматів 3D-моделей і текстур

Вибір формату 3D-моделей і текстур має велике значення для роботи вебсистеми. Формат повинен підтримувати необхідні дані, бути сумісним із Three.js і забезпечувати зручне завантаження у браузері.

У межах даної роботи для моделей використовується формат GLB/GLTF, який є одним із найпоширеніших форматів для веборієнтованої 3D-графіки [4].

2.7.1 Особливості формату GLB/GLTF

GLTF є форматом для передачі та відображення 3D-моделей у реальному часі [4]. Його часто називають стандартом для 3D-графіки у вебсередовищі. Формат підтримує геометрію, матеріали, текстури, анімації та інші дані, необхідні для відображення моделі.

GLB є бінарною версією GLTF [4]. Його перевага полягає в тому, що всі основні ресурси можуть зберігатися в одному файлі. Це зручно для вебдодатку, оскільки зменшує кількість окремих файлів і спрощує підключення моделі.

Three.js має підтримку GLB/GLTF через GLTFLoader, тому цей формат добре підходить для реалізації 3D-візуалізації історичних будівель [1, 4, 9].

2.7.2 Вимоги до текстур і матеріалів

Текстури і матеріали впливають на зовнішній вигляд 3D-моделі. Для історичних будівель важливо, щоб вони передавали характер поверхонь, кольори, архітектурні деталі та загальну атмосферу об’єкта.

Водночас текстури не повинні бути надмірно великими, оскільки це може збільшити час завантаження і навантаження на браузер. Тому під час підготовки моделі необхідно використовувати оптимальні розміри текстур.

Матеріали також повинні бути налаштовані з урахуванням продуктивності. Занадто складні матеріали або велика кількість ефектів можуть погіршити роботу сцени, особливо на слабших пристроях.

2.7.3 Переваги використання GLB у вебсередовищі

Використання формату GLB має кілька переваг для вебсистеми [4]. По-перше, він дозволяє зберігати модель, матеріали та текстури в одному файлі. Це спрощує структуру проєкту і зменшує кількість помилок під час завантаження.

По-друге, GLB добре підтримується сучасними 3D-бібліотеками, зокрема Three.js. Це дозволяє швидко інтегрувати модель у вебсцену без складної конвертації.

По-третє, формат GLB підходить для оптимізації. Моделі можна стискати, зменшувати кількість полігонів і налаштовувати текстури, щоб забезпечити кращу продуктивність у браузері.

2.8 Підготовка 3D-моделі історичної будівлі

Підготовка 3D-моделі є важливим етапом, оскільки від якості моделі залежить загальне враження від вебсистеми.

Модель повинна бути достатньо деталізованою, але водночас оптимізованою для використання у браузері.

Перед підключенням до вебдодатку модель необхідно перевірити, налаштувати її масштаб, положення, матеріали та текстури.

2.8.1 Перевірка масштабу та положення моделі

Після підготовки або завантаження 3D-моделі необхідно перевірити її масштаб і положення. Якщо модель занадто велика або занадто мала, це може ускладнити налаштування камери та навігації.

Також важливо, щоб модель була правильно розташована відносно центру сцени. Якщо об’єкт зміщений занадто далеко, камера може фокусуватися неправильно, а точки інтересу буде складніше налаштувати.

Перевірка масштабу і положення дозволяє забезпечити коректне відображення будівлі в 3D-переглядачі та спростити подальшу роботу з екскурсійним маршрутом.

2.8.2 Оптимізація моделі для браузера

Оптимізація моделі необхідна для стабільної роботи вебсистеми. Надмірна кількість полігонів, складні матеріали або великі текстури можуть призвести до зниження продуктивності, довгого завантаження та затримок під час навігації.

Під час оптимізації необхідно зменшити зайву деталізацію, яка не впливає суттєво на візуальне сприйняття будівлі. Також варто перевірити текстури, матеріали та загальну структуру моделі.

Оптимізована модель повинна зберігати основні архітектурні особливості будівлі, але при цьому бути достатньо легкою для відображення у браузері.

2.8.3 Експорт і підключення моделі до вебсистеми

Після перевірки та оптимізації модель експортується у формат GLB [4, 14]. Цей файл розміщується у відповідному каталозі вебдодатку і підключається до 3D-переглядача за допомогою GLTFLoader [9].

Підключення моделі передбачає вказання правильного шляху до файлу, додавання моделі до сцени, налаштування масштабу, позиції та освітлення. Після цього перевіряється коректність відображення моделі у браузері.

На цьому етапі також можна налаштовувати початкове положення камери, точки інтересу та параметри екскурсії, щоб користувач одразу отримував зручний вигляд будівлі.

3 РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ ВЕБДОДАТКУ НА ОСНОВІ THREE.JS

3.1 Створення сцени, камери та освітлення

Першим етапом реалізації 3D-переглядача було створення базової сцени Three.js [1]. У файлі viewer.js було налаштовано сцену, перспективну камеру, WebGL-рендерер та джерела освітлення. Ці елементи забезпечують відображення 3D-моделі історичної будівлі у браузері (див. рис. 3.1).

```
const scene = new THREE.Scene();
scene.background = new THREE.Color(0x10141c);

const camera = new THREE.PerspectiveCamera(
  60,
  window.innerWidth / window.innerHeight,
  0.1,
  5000
);
camera.position.set(4, 3, 8);

const renderer = new THREE.WebGLRenderer({ antialias: true });
renderer.setPixelRatio(Math.min(window.devicePixelRatio, 2));
renderer.setSize(window.innerWidth, window.innerHeight);
renderer.outputColorSpace = THREE.SRGBColorSpace;
document.body.appendChild(renderer.domElement);

const ambientLight = new THREE.AmbientLight(0xffffff, 1.15);
scene.add(ambientLight);

const dirLight1 = new THREE.DirectionalLight(0xffffff, 1.2);
dirLight1.position.set(8, 10, 6);
scene.add(dirLight1);
```

Рисунок 3.1 – Створення сцени, камери, рендерера та освітлення

Сцена використовується як основний контейнер для моделі, точок інтересу та допоміжних елементів. Камера відповідає за формування ракурсу перегляду, а рендерер виводить готове зображення на сторінку viewer.html. Освітлення було налаштовано таким чином, щоб модель будівлі була достатньо видимою та зберігала об’ємність.

3.1.1 Ініціалізація сцени Three.js

Ініціалізація сцени виконується у файлі viewer.js. У розробленій системі сцена Three.js використовується для розміщення завантаженої GLB-моделі, маркерів точок інтересу та допоміжних об’єктів.

Після створення сцени до неї додаються основні елементи, необхідні для відображення історичної будівлі.

Для коректної роботи 3D-переглядача також задаються базові параметри сцени та підготовлюються змінні для подальшого завантаження моделі, побудови точок інтересу і запуску екскурсійного режиму. Такий підхід дозволяє об’єднати всі елементи 3D-візуалізації в одному робочому середовищі.

3.1.2 Налаштування камери та рендерера

У вебдодатку використовується перспективна камера, оскільки вона дозволяє природно відображати архітектурний об’єкт у тривимірному просторі.

Положення камери налаштовується відносно завантаженої моделі, щоб користувач одразу бачив будівлю у зручному ракурсі.

На рисунку 3.2 наведено фрагмент коду налаштування камери, рендерера та параметрів відображення 3D-сцен.

```
const camera = new THREE.PerspectiveCamera(
  60,
  window.innerWidth / window.innerHeight,
  0.1,
  5000
);
camera.position.set(4, 3, 8);

const renderer = new THREE.WebGLRenderer({ antialias: true });
renderer.setPixelRatio(Math.min(window.devicePixelRatio, 2));
renderer.setSize(window.innerWidth, window.innerHeight);
renderer.outputColorSpace = THREE.SRGBColorSpace;
document.body.appendChild(renderer.domElement);

const controls = new OrbitControls(camera, renderer.domElement);
controls.enableDamping = true;
controls.dampingFactor = 0.06;
controls.target.set(0, 1, 0);
controls.update();
```

Рисунок 3.2 – Налаштування камери, рендерера та керування сценою

Рендерер відповідає за відображення 3D-сцени на сторінці viewer.html. Його розмір налаштовується відповідно до розміру вікна браузера. При зміні розміру екрана параметри камери та рендерера оновлюються, що дозволяє уникнути спотворення зображення та забезпечити коректну роботу 3D-переглядача на різних пристроях.

3.1.3 Налаштування освітлення сцени

Для кращого відображення історичної будівлі у сцені було використано джерела освітлення. Освітлення дозволяє зробити модель більш об’ємною, підкреслити її форму та архітектурні деталі. Без налаштування світла модель може виглядати занадто темною або втрачати частину візуальної інформації.

На рисунку 3.3 наведено фрагмент коду налаштування джерел освітлення у 3D-сцені.

```
const ambientLight = new THREE.AmbientLight(0xffffff, 1.15);
scene.add(ambientLight);

const dirLight1 = new THREE.DirectionalLight(0xffffff, 1.2);
dirLight1.position.set(8, 10, 6);
scene.add(dirLight1);

const dirLight2 = new THREE.DirectionalLight(0xffffff, 0.65);
dirLight2.position.set(-8, 6, -6);
scene.add(dirLight2);
```

Рисунок 3.3 – Налаштування освітлення 3D-сцени

У розробленому 3D-переглядачі освітлення налаштовано так, щоб користувач міг комфортно розглядати модель з різних ракурсів. Це особливо важливо для режиму віртуальної екскурсії, де камера переходить між підготовленими точками огляду.

3.2 Імпорт 3D-моделі у форматі GLB/GLTF

Для відображення історичних будівель у вебсистемі використовуються 3D-моделі у форматі GLB [14, 15]. Цей формат зручний для вебсередовища, оскільки дозволяє зберігати геометрію, матеріали та текстури в одному файлі.

У розробленому проєкті моделі зберігаються у каталозі models, а шлях до відповідної моделі вказується у файлі data/buildings.js.

На рисунку 3.4 наведено фрагмент коду визначення обраної будівлі та шляху до її 3D-моделі.

Завантаження моделі виконується на сторінці viewer.html за допомогою GLTFLoader. Після завантаження модель додається до сцени Three.js, центрується та готується до подальшої взаємодії з користувачем. Після цього виконується фокусування камери на моделі та створення точок інтересу.

```
const params = new URLSearchParams(window.location.search);
const buildingId = params.get("id");
const building = getBuildingById(buildingId);

if (!building) {
  alert("Будівлю не знайдено");
  location.href = "index.html";
}

function loadModel() {
  if (!building?.modelPath) {
    console.error("У будівлі не вказаний modelPath");
    return;
  }

  clearOldModel();
  loader.load(building.modelPath, onModelLoaded);
}
```

Рисунок 3.4 – Визначення обраної будівлі та шляху до 3D-моделі

3.2.1 Завантаження моделі за допомогою GLTFLoader

У системі завантаження 3D-моделі реалізовано через GLTFLoader [9]. Після вибору будівлі користувач переходить на сторінку 3D-візуалізації, де за ідентифікатором об’єкта визначається потрібний шлях до GLB-файлу. Завантажувач отримує цей шлях, імпортує модель і додає її до сцени.

Після додавання моделі обчислюються її межі за допомогою Box3. Це потрібно для визначення центру моделі та подальшого правильного фокусування камери. Такий підхід дозволяє коректно працювати з різними моделями, які можуть відрізнятися розмірами та положенням у просторі.

На рисунку 3.5 наведено фрагмент коду завантаження GLB-моделі за допомогою GLTFLoader.

```
loader.load(
  building.modelPath,
  (gltf) => {
    loadedModel = gltf.scene;
    scene.add(loadedModel);
    modelBounds = new THREE.Box3().setFromObject(loadedModel);
    const center = new THREE.Vector3();
    modelBounds.getCenter(center);
    loadedModel.position.sub(center);
    focusOnModel();
    buildPoints();
  }
);
```

Рисунок 3.5 – Завантаження 3D-моделі у форматі GLB

Використання GLTFLoader спрощує процес імпорту, оскільки розробнику не потрібно вручну створювати геометрію будівлі у коді. Достатньо підготувати модель у відповідному форматі та правильно підключити її до вебдодатку.

3.2.2 Масштабування та позиціонування моделі

Після завантаження модель центрується відносно сцени. Це необхідно для того, щоб камера, елементи керування та точки інтересу працювали стабільно незалежно від початкового положення моделі у GLB-файлі. Якщо модель не центрувати, вона може відображатися зі зміщенням або бути незручною для перегляду. На рисунку 3.6 наведено фрагмент коду центрування, масштабування та позиціонування 3D-моделі у сцені.

У розробленій вебсистемі після визначення центру моделі її позиція коригується, після чого виконується функція фокусування на об’єкті. Завдяки цьому користувач одразу бачить модель у зручному ракурсі після відкриття сторінки 3D-візуалізації.

```
modelBounds = new THREE.Box3().setFromObject(loadedModel);

const center = new THREE.Vector3();
modelBounds.getCenter(center);

loadedModel.position.sub(center);

modelBounds = new THREE.Box3().setFromObject(loadedModel);

focusOnModel();
```

Рисунок 3.6 – Центрування та позиціонування 3D-моделі у сцені

3.2.3 Перевірка коректності відображення моделі

Після імпорту моделі було виконано перевірку її відображення у браузері. Перевірялась наявність геометрії, правильність завантаження матеріалів і текстур, а також зручність початкового ракурсу. У разі потреби модель можна оглянути з різних боків за допомогою керування камерою.

Після завантаження 3D-модель відображається безпосередньо у вебпереглядачі в середовищі Three.js. Це дозволяє оцінити коректність її розміщення у сцені, перевірити масштаб, орієнтацію та загальний вигляд об’єкта. Користувач може змінювати ракурс, наближати або віддаляти модель для детальнішого огляду.

3.3 Налаштування матеріалів і текстур

Матеріали та текстури 3D-моделей імпортуються разом із GLB-файлами. Після завантаження моделі у вебсистемі перевіряється, чи коректно відображаються поверхні, кольори та архітектурні деталі. Для історичних будівель це важливо, оскільки зовнішній вигляд моделі має передавати основні особливості об’єкта.

У процесі перевірки було оцінено, чи модель не виглядає занадто темною, чи не втрачаються текстури, чи правильно видно фасади та окремі елементи будівлі. Освітлення сцени було підібрано таким чином, щоб матеріали моделі виглядали достатньо виразно у браузері.

3.3.1 Перевірка матеріалів після імпорту

Після імпорту GLB-моделі було перевірено, чи правильно відображаються її матеріали. У разі некоректного освітлення або неправильних параметрів матеріалів модель може виглядати плоскою. Тому після завантаження моделі важливо оцінити її.

У розробленій системі матеріали не створюються вручну для кожного об’єкта, а використовуються ті, що вже містяться у GLB-моделі. Це спрощує підключення моделей і дозволяє зберігати їх оригінальний вигляд.

3.3.2 Перевірка відображення текстур після імпорту моделі

Текстури 3D-моделі завантажуються разом із GLB-файлом, тому окреме ручне налаштування текстур у коді вебдодатку не виконувалося. Основним завданням на цьому етапі була перевірка того, чи коректно відображаються поверхні моделі після імпорту у сцену Three.js.

На рисунку 3.7 наведено результат відображення 3D-моделі історичної будівлі після її завантаження у вебпереглядач.

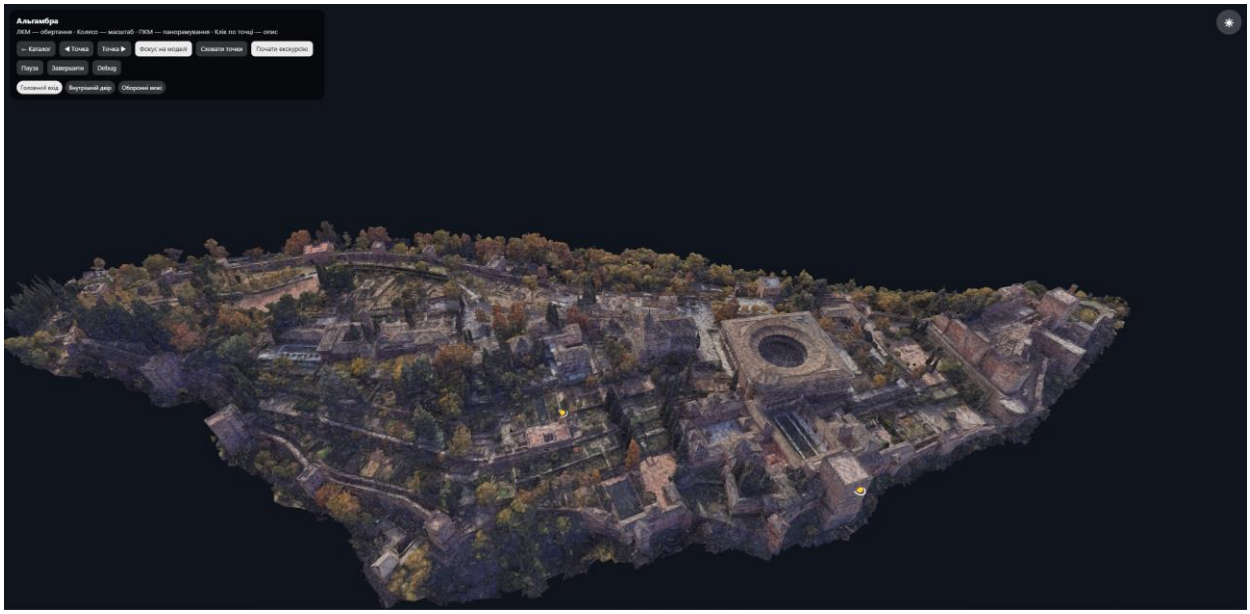


Рисунок 3.7 – Відображення 3D-моделі історичної будівлі у вебсистемі

Після завантаження моделі було перевірено наявність текстур, правильність відображення кольорів, фасадів, даху та інших архітектурних елементів. Також було оцінено, чи не виглядає модель занадто темною або спотвореною у браузері.

3.3.3 Оптимізація матеріалів для продуктивності

Оскільки 3D-сцена працює у браузері, важливо враховувати продуктивність під час відображення моделей. Надмірно великі текстури або складні матеріали можуть збільшувати час завантаження та знижувати плавність роботи сцени. Тому для вебсистеми використовуються моделі у форматі GLB, що дозволяє зручно зберігати ресурси та підключати їх до Three.js.

Під час тестування було перевірено, що модель відкривається у браузері, реагує на керування камерою та не заважає роботі основних елементів інтерфейсу. Це підтверджує, що використані матеріали та текстури придатні для демонстрації у вебсередовищі.

3.4 Реалізація навігації користувача

Для взаємодії користувача з 3D-моделлю у вебсистемі реалізовано керування камерою.

Навігація дозволяє оглядати історичну будівлю з різних боків, змінювати масштаб і самостійно обирати потрібний ракурс. Це є важливою частиною 3D-візуалізації, оскільки користувач повинен не просто бачити модель, а мати можливість досліджувати її.

На рисунку 3.8 наведено фрагмент коду підключення OrbitControls для керування камерою у 3D-сцені.

```
const controls = new OrbitControls(camera, renderer.domElement);

controls.enableDamping = true;
controls.dampingFactor = 0.06;
controls.target.set(0, 1, 0);
controls.minDistance = 1;
controls.maxDistance = 2000;
controls.update();
```

Рисунок 3.8 – Реалізація навігації користувача у 3D-сцені

У проєкті використовується OrbitControls, що забезпечує обертання камери навколо моделі, наближення та віддалення. Завдяки цьому сторінка 3D-візуалізації працює як інтерактивний переглядач архітектурного об’єкта [10].

3.4.1 Керування камерою за допомогою OrbitControls

OrbitControls використовується для організації ручного керування камерою у 3D-сцені. Користувач може обертати модель мишею, змінювати відстань до неї та розглядати будівлю з різних ракурсів. Такий спосіб керування є зручним для архітектурної візуалізації, оскільки модель залишається у центрі уваги.

OrbitControls є зручним рішенням для архітектурної візуалізації, оскільки користувач зберігає фокус на об’єкті, але може змінювати ракурс. Це дозволяє розглядати фасад, дах, бокові частини та загальний вигляд будівлі.

3.4.2 Обмеження руху камери

Для забезпечення зручності перегляду необхідно встановити обмеження руху камери. Без таких обмежень користувач може надто далеко віддалити камеру, занадто наблизитися до моделі або обрати незручний кут огляду.

Обмеження можуть стосуватися мінімальної та максимальної відстані до об’єкта, кута повороту або області, навколо якої обертається камера. Це дозволяє зберегти контроль над переглядом сцени.

На рисунку 3.9 наведено фрагмент коду налаштування обмежень руху камери. Такі налаштування покращують користувацький досвід і роблять роботу з 3D-переглядачем більш передбачуваною.

```
controls.minDistance = 1;
controls.maxDistance = 2000;

controls.enableDamping = true;
controls.dampingFactor = 0.06;

controls.update();
```

Рисунок 3.9 – Реалізація навігації користувача у 3D-сцені

3.4.3 Адаптація керування до різних пристроїв

Оскільки вебдодаток може використовуватися на різних пристроях, керування камерою повинно бути адаптоване до миші, тачпада або сенсорного екрана. Це дозволяє комфортно працювати з системою як на комп’ютері, так і на мобільних пристроях. На комп’ютері користувач може обертати камеру за допомогою миші, масштабувати колесом і змінювати положення огляду. На сенсорних пристроях аналогічні дії можуть виконуватися жестами. Адаптація керування підвищує доступність вебсистеми та дозволяє використовувати її у різних умовах.

3.5 Реалізація точок інтересу

Для підвищення інформативності 3D-візуалізації у вебсистемі реалізовано точки інтересу. Вони позначають важливі частини історичної

будівлі та дозволяють користувачу отримати додаткову текстову інформацію про відповідний елемент моделі.

Дані про точки інтересу зберігаються у JSON-файлах у каталозі texts/tours. Для кожної точки вказуються назва, опис, координати маркера, положення камери та ціль огляду. Така структура дозволяє окремо редагувати сценарій екскурсії без зміни основного коду 3D-переглядача.

3.5.1 Структура даних для точок інтересу

Кожна точка інтересу містить набір даних, необхідний для її відображення у 3D-сцені та використання в екскурсійному режимі. До таких даних належать назва точки, текстовий опис, координати розташування маркера, положення камери та напрямок огляду.

Завдяки такій структурі одна й та сама точка може використовуватися як для ручного вибору користувачем, так і для автоматичного переходу під час віртуальної екскурсії. Це дозволяє зробити систему більш гнучкою та зручною для подальшого розширення.

3.5.2 Відображення маркерів у 3D-сцені

Після завантаження моделі система створює маркери точок інтересу та додає їх до сцени. Маркери розміщуються у координатах, вказаних у даних екскурсії, і візуально показують користувачу, які частини моделі мають додатковий опис.

Маркер повинен бути помітним, але не повинен заважати огляду моделі. Його розмір, колір і положення мають бути підібрані так, щоб користувач легко міг його знайти. На рисунку 3.10 наведено приклад відображення маркера точки інтересу у 3D-сцені.

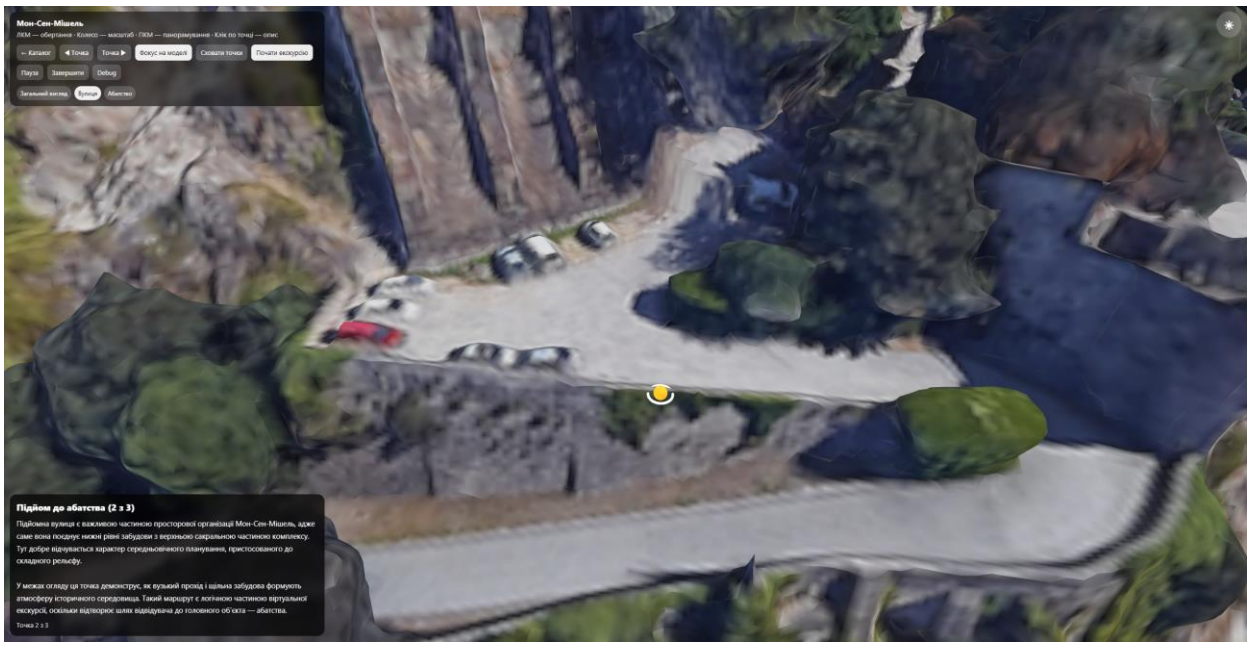


Рисунок 3.10 – Відображення точки інтересу у 3D-сцені

Під час натискання на маркер система повинна відображати інформацію, пов’язану з цією точкою. Це забезпечує інтерактивну взаємодію користувача з 3D-сценою.

3.5.3 Показ інформаційної панелі

Після вибору точки інтересу користувачу відкривається інформаційна панель. У ній відображається назва точки, її порядковий номер і текстовий опис. Це дозволяє поєднати візуальне дослідження моделі з пояснювальною інформацією.

Панель повинна мати зручне розташування і не перекривати основні елементи 3D-сцени. Текст у ній має бути читабельним, а її оформлення повинно відповідати загальному стилю вебдодатку.

На рисунку 3.11 наведено приклад інформаційної панелі, що відображається після вибору точки інтересу.

Завдяки інформаційній панелі користувач отримує змістовний супровід під час перегляду моделі.

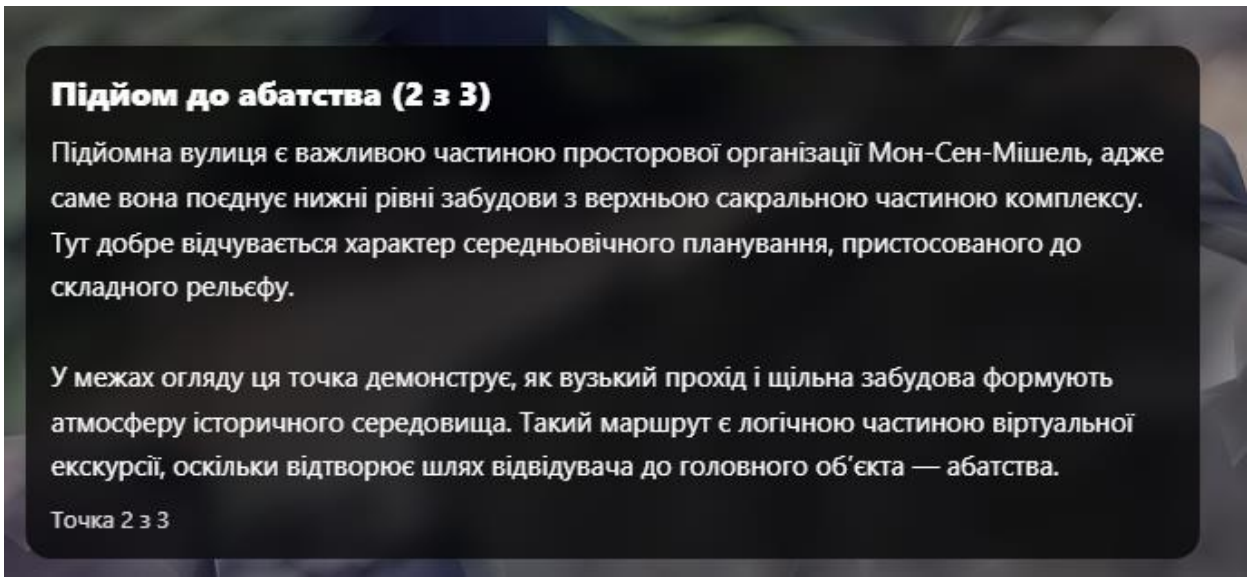


Рисунок 3.11 – Інформаційна панель точки інтересу

3.5.4 Збір і налаштування координат точок інтересу

Для налаштування точок інтересу у розробленій вебсистемі використовувався спеціальний режим отримання координат у 3D-переглядачі. Під час роботи з моделлю користувач обирав потрібний ракурс, після чого у консолі браузера виводилися координати камери, цільової точки огляду та положення маркера точки інтересу.

Отримані координати використовувалися для формування даних екскурсії у JSON-файлах, які зберігаються у каталозі texts/tours. Для кожної точки інтересу вказувалися назва, текстовий опис, координати маркера, положення камери та координати цілі, на яку спрямовано огляд. Такий підхід дозволив вручну підібрати зручні ракурси для окремих частин історичної будівлі.

На рисунку 3.12 наведено приклад структури даних точки інтересу з координатами маркера, положенням камери та напрямком огляду.

Після внесення координат у файл екскурсії вебсистема завантажувала ці дані на сторінці 3D-переглядача. Маркери точок інтересу відображалися у відповідних місцях сцени, а під час запуску віртуальної екскурсії камера

переходила до збережених позицій. Завдяки цьому користувач бачив саме той фрагмент моделі, який відповідав поточному.


```
{
  "name": "Головний вхід",
  "title": "Вхід до комплексу Альгамбра",
  "camera": [
    -5.121,
    2.581,
    1.828
  ],
  "target": [
    -4.068,
    -0.252,
    -3.034
  ],
  "point": [
    -3.909,
    -0.301,
    -2.71
  ],
  "text": "Ця точка демонструє головний вхід до комплексу Альгамбра..."
}
```


Рисунок 3.12 – Приклад даних точки інтересу у файлі екскурсії

3.6 Розробка режиму віртуальної екскурсії

У розробленій вебсистемі режим віртуальної екскурсії реалізовано на сторінці viewer.html. Основна логіка роботи цього режиму знаходиться у файлі js/viewer.js, а дані для екскурсій зберігаються у JSON-файлах каталогу texts/tours. Для кожної історичної будівлі передбачено окремий файл з точками екскурсії, наприклад alhambra.json, big-ben.json, colosseum.json та інші.

Після відкриття сторінки 3D-візуалізації система визначає, яку будівлю обрав користувач, завантажує відповідну GLB-модель і підключає дані точок інтересу для цієї будівлі. Кожна точка містить координати маркера, положення камери, ціль огляду, назву та текстовий опис. Саме ці дані використовуються

для послідовного переходу між частинами моделі під час екскурсії. На рисунку 3.13 наведено фрагмент коду запуску режиму віртуальної екскурсії.

```
async function startTour() {
  const points = getCurrentPoints();

  if (!points.length) {
    setTourStatus("Точки екскурсії відсутні");
    return;
  }

  activeTourRunId += 1;
  const runId = activeTourRunId;

  isTourPlaying = true;
  isTourPaused = false;
  updateTourButtons();

  for (let i = 0; i < points.length; i += 1) {
    if (!isTourPlaying || runId !== activeTourRunId) {
      return;
    }

    await focusPoint(i, true);
  }
}
```

Рисунок 3.13 – Реалізація запуску режиму віртуальної екскурсії

Кнопки керування екскурсією розміщені безпосередньо на сторінці 3D-переглядача. Користувач може запустити екскурсію, після чого система автоматично переходить між підготовленими точками, змінює положення камери та показує текстовий супровід у інформаційній панелі.

3.6.1 Послідовність переходу між точками

Послідовність переходу між точками у вебсистемі задається порядком об’єктів у JSON-файлі екскурсії. Коли користувач запускає екскурсійний режим, файл js/viewer.js отримує список точок для поточної будівлі та починає послідовно переходити від першої точки до останньої.

Для кожної точки використовується її назва, опис, координати камери та координати цілі огляду. Це дозволяє не просто показати маркер на моделі, а одразу встановити камеру в такий ракурс, у якому відповідний елемент будівлі добре видно користувачу.

Послідовність переходів повинна бути зрозумілою і не перевантаженою. Кожна наступна точка має доповнювати попередню та поступово розкривати особливості будівлі.

3.6.2 Анімація переміщення камери

Переміщення камери між точками екскурсії реалізовано у файлі js/viewer.js. Під час переходу до чергової точки система бере з JSON-файлу координати camera та target, після чого змінює положення камери і напрямок огляду. Завдяки цьому користувач бачить саме ту частину моделі, яка описується у поточній точці екскурсії.

У проєкті для кожної точки вручну підібрано зручний ракурс. Спочатку модель відкривалась у 3D-переглядачі, після чого підбирався потрібний вигляд камери. Далі координати камери, цілі огляду та маркера переносилися у відповідний JSON-файл екскурсії. На рисунку 3.14 наведено фрагмент коду анімації переміщення камери між точками інтересу.

```
function toggleTheme() {
  const currentTheme = document.documentElement.getAttribute("data-theme") || "dark";
  const nextTheme = currentTheme === "dark" ? "light" : "dark";
  localStorage.setItem(THEME_STORAGE_KEY, nextTheme);
  applyTheme(nextTheme);
}
function setupThemeButtons() {
  const buttons = document.querySelectorAll("[data-theme-toggle]");
  buttons.forEach((button) => {
    button.addEventListener("click", toggleTheme);
  });
}
```

Рисунок 3.14 – Перехід камери до точки інтересу під час екскурсії

Анімація повинна бути достатньо швидкою, щоб не затримувати користувача, але водночас плавною, щоб зберігати ефект екскурсії.

3.6.3 Відображення текстового супроводу екскурсії

Текстовий супровід екскурсії у вебсистемі також зберігається у JSON-файлах каталогу texts/tours. Для кожної точки інтересу вказано поле з текстом опису. Коли користувач обирає точку або коли екскурсія переходить до неї автоматично, цей текст виводиться в інформаційну панель на сторінці 3D-переглядача.

Таким чином, екскурсія складається не лише з переміщення камери, а й з пояснення того, що саме бачить користувач. Це дозволяє зробити перегляд 3D-моделі більш змістовним. Наприклад, користувач може побачити фасад, башту, вхід або інший архітектурний елемент і одразу прочитати короткий опис.

Завдяки текстовому супроводу віртуальна екскурсія стає більш змістовною та корисною для користувача.

3.6.4 Керування запуском і завершенням екскурсії

Керування режимом віртуальної екскурсії реалізовано через кнопки на сторінці viewer.html. Користувач бачить елементи керування поруч із 3D-сценою і може запустити екскурсію під час перегляду моделі. Після запуску система починає послідовний перехід між точками інтересу.

На рисунку 3.15 наведено елементи керування запуском і завершенням режиму віртуальної екскурсії.

У файлі js/viewer.js для цього використовується логіка запуску екскурсії, перевірки наявності точок, оновлення статусу та переходу між ними.

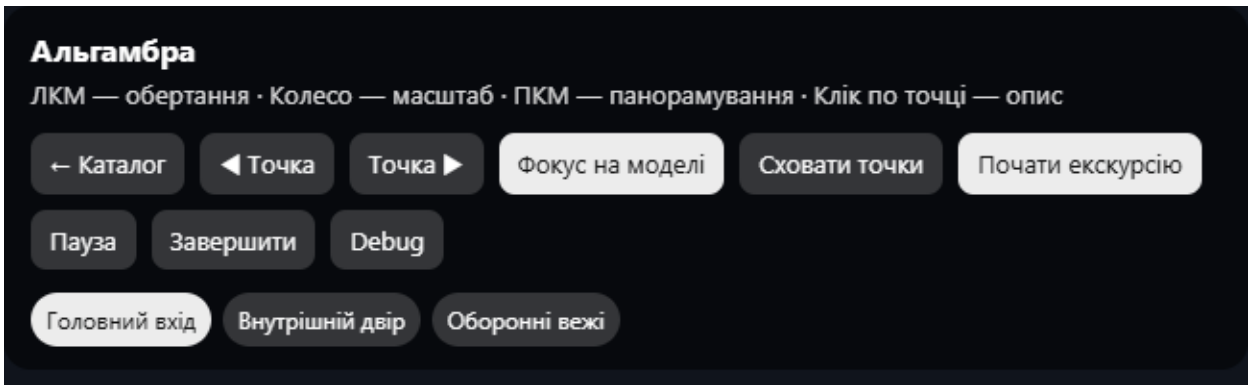


Рисунок 3.15 – Елементи керування режимом віртуальної екскурсії

Якщо для будівлі відсутні точки екскурсії, система виводить відповідне повідомлення.

Якщо точки є, користувач бачить поточний етап екскурсії та інформацію про обрану частину будівлі.

3.7 Реалізація користувацького інтерфейсу

Користувацький інтерфейс вебсистеми реалізовано як послідовність трьох основних сторінок: головної сторінки каталогу, сторінки детального опису будівлі та сторінки 3D-візуалізації. Така структура дозволяє користувачу поступово перейти від вибору об’єкта до його інтерактивного перегляду.

На рисунку 3.16 наведено приклад переходу користувача від каталогу до сторінки опису історичної будівлію.

Головна сторінка створена у файлі index.html, каталог будівель формується за допомогою js/catalog.js, а дані про об’єкти зберігаються у файлі data/buildings.js [8, 9].

Сторінка опису реалізована у building.html і використовує js/building.js для завантаження інформації про конкретну будівлю.

Сторінка 3D-переглядача реалізована у viewer.html, а її логіка знаходиться у js/viewer.js.

```
function openBuildingPage(building) {
  location.href = `building.html?id=${encodeURIComponent(building.id)}`;
}

const card = document.createElement("div");
card.className = "card";
card.onclick = () => openBuildingPage(b);

card.innerHTML = `
  <div class="cardTop">
    ${imgHtml}
    <div>
      <div class="tag">${countryName}</div>
      <div class="title">${b.title}</div>
      <div class="meta">${b.city}</div>
    </div>
  </div>
  <div class="desc">${b.description}</div>
`;
```

Рисунок 3.16 – Реалізація переходу з каталогу до сторінки опису будівлі

3.7.1 Головна сторінка каталогу

Головна сторінка вебсистеми реалізована у файлі index.html. На ній відображається каталог історичних будівель. Картки будівель формуються динамічно на основі масиву BUILDINGS з файлу data/buildings.js. За виведення карток відповідає файл js/catalog.js. На рисунку 3.17 наведено головну сторінку вебсистеми з каталогом історичних будівель.

Кожна картка містить зображення будівлі, її назву, місто, країну та короткий опис. Після натискання на картку користувач переходить до сторінки building.html з параметром id у адресному рядку. Завдяки цьому система розуміє, яку саме будівлю потрібно відкрити.

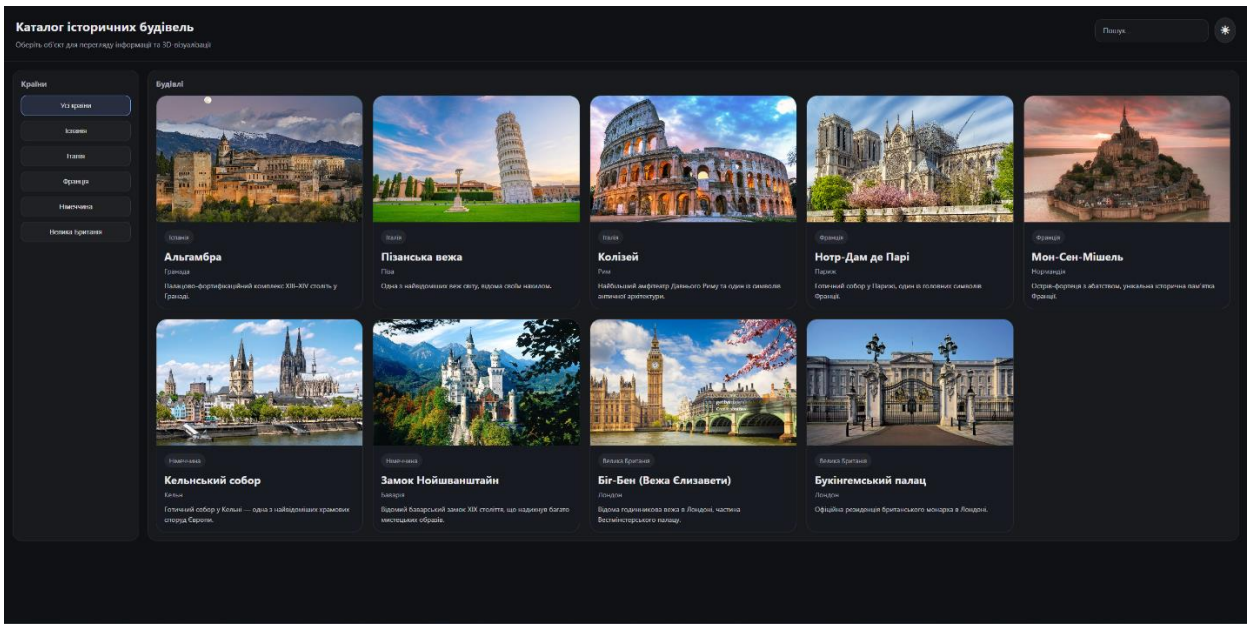


Рисунок 3.17 – Головна сторінка вебсистеми з каталогом

Головна сторінка повинна бути візуально зрозумілою і адаптивною, щоб коректно відображатися на різних пристроях.

3.7.2 Сторінка детального опису будівлі

Сторінка детального опису будівлі реалізована у файлі building.html. Вона відкривається після вибору об’єкта у каталозі.

Файл js/building.js отримує ідентифікатор будівлі з адресного рядка, знаходить відповідний об’єкт у data/buildings.js і виводить на сторінку його назву, місто, країну, зображення та опис.

На рисунку 3.18 наведено сторінку детального опису історичної будівлі.

Текстовий опис будівлі завантажується з окремого файлу у каталозі texts/buildings. Це дозволяє зберігати великі описи окремо від основного JavaScript-коду і спрощує редагування інформації про об’єкти.

На сторінці також розміщено кнопку переходу до 3D-візуалізації. Після її натискання користувач переходить до viewer.html з тим самим ідентифікатором будівлі.

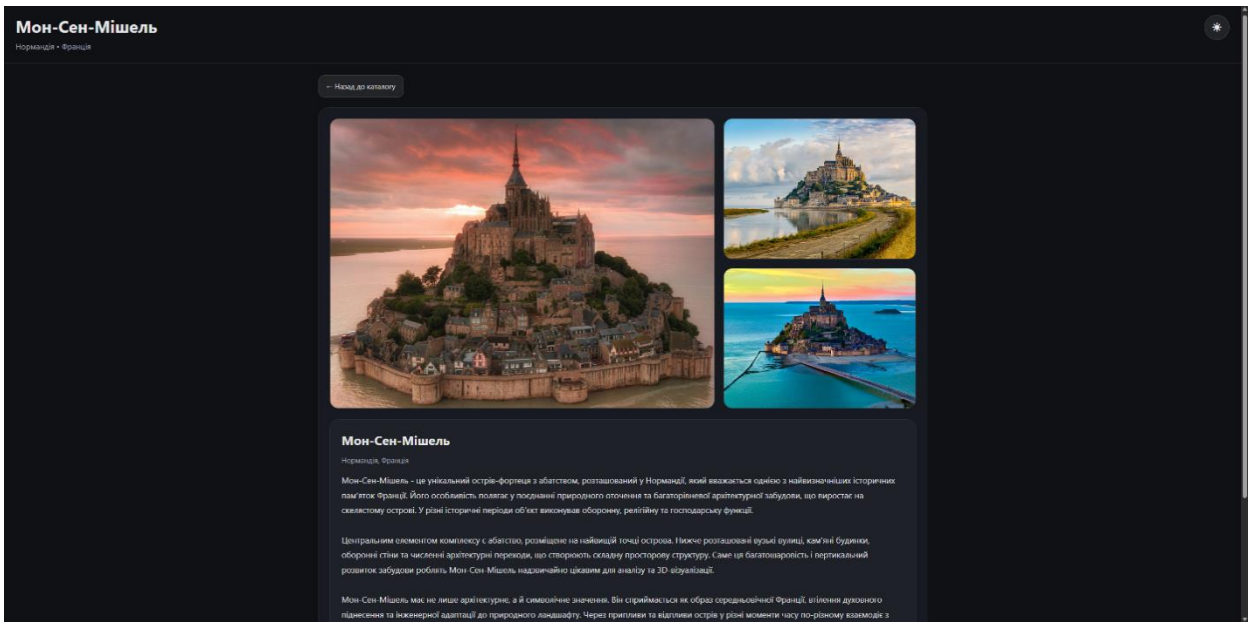


Рисунок 3.18 – Сторінка детального опису історичної будівлі

3.7.3 Сторінка 3D-візуалізації

Сторінка 3D-візуалізації реалізована у файлі viewer.html. Вона є основною практичною частиною вебсистеми, оскільки саме тут користувач взаємодіє з 3D-моделлю історичної будівлі. На рисунку 3.19 наведено сторінку 3D-візуалізації історичної будівлі з інтерактивною сценою.



Рисунок 3.19 – Сторінка інтерактивної 3D-візуалізації історичної будівлі

Логіка цієї сторінки знаходиться у файлі js/viewer.js. Після відкриття viewer.html система отримує id будівлі з адресного рядка, знаходить відповідну модель у data/buildings.js і завантажує GLB-файл з каталогу models. Одночасно підключаються дані екскурсії з каталогу texts/tours, після чого створюються точки інтересу.

На сторінці 3D-переглядача користувач може обертати модель, змінювати ракурс, натискати на точки інтересу, читати інформаційну панель і запускати режим віртуальної екскурсії.

3.7.4 Реалізація перемикання теми інтерфейсу

У вебсистемі реалізовано перемикання світлої та темної теми оформлення. Ця функція винесена в окремий файл js/theme.js, що дозволяє використовувати її на різних сторінках проєкту. Кнопка перемикання теми розміщена в інтерфейсі вебдодатку.

Після натискання на кнопку система змінює значення атрибута теми для документа. Обрана тема зберігається у локальному сховищі браузера, тому після повторного відкриття вебсистеми користувач бачить раніше вибране оформлення. На рисунку 3.20 наведено приклад перемикання світлої та темної теми інтерфейсу вебдодатку.

```
function toggleTheme() {
  const currentTheme = document.documentElement.getAttribute("data-theme") || "dark";
  const nextTheme = currentTheme === "dark" ? "light" : "dark";
  localStorage.setItem(THEME_STORAGE_KEY, nextTheme);
  applyTheme(nextTheme);
}

function setupThemeButtons() {
  const buttons = document.querySelectorAll("[data-theme-toggle]");
  buttons.forEach((button) => {
    button.addEventListener("click", toggleTheme);
  });
}
```

Рисунок 3.20 – Перемикання теми оформлення вебдодатку

Така функція покращує зручність користування і демонструє додаткову інтерактивність вебдодатку.

3.8 Оптимізація продуктивності вебсцени

У розробленій вебсистемі оптимізація продуктивності пов’язана насамперед з роботою 3D-моделей у форматі GLB. Оскільки моделі історичних будівель можуть мати великий розмір, важливо забезпечити стабільне завантаження сцени, коректне відображення моделі та зручну взаємодію користувача з камерою.

У проєкті 3D-моделі зберігаються окремо у каталозі models, а дані про шляхи до них винесено у файл data/buildings.js. Такий підхід дозволяє не змішувати моделі з логікою роботи системи та полегшує додавання нових об’єктів у майбутньому.

Також у проєкті використовується розділення логіки на окремі JavaScript-файли: catalog.js відповідає за каталог, building.js – за сторінку опису, viewer.js – за 3D-переглядач, а theme.js – за перемикання теми. Це робить структуру проєкту зрозумілішою та спрощує підтримку коду.

3.8.1 Оптимізація розміру 3D-моделі

У межах проєкту моделі історичних будівель використовуються у форматі GLB. Цей формат дозволяє зберігати геометрію, матеріали та текстури в одному файлі, що спрощує підключення моделі до вебдодатку. Водночас деякі GLB-файли мають значний розмір, тому їх завантаження може залежати від продуктивності комп’ютера та швидкості роботи браузера.

Під час перевірки роботи системи було протестовано відкриття моделей у 3D-переглядачі. Після завантаження модель центрується у сцені, а камера

фокусується на об’єкті. Це дозволяє користувачу одразу почати перегляд без ручного пошуку моделі у просторі.

3.8.2 Оптимізація рендерингу сцени

Для рендерингу сцени використовується WebGL-рендерер Three.js. У проєкті не застосовуються надмірно складні графічні ефекти, що дозволяє зменшити навантаження на браузер.

Основний акцент зроблено на стабільному відображенні моделі, роботі керування камерою та точок інтересу.

Параметри рендерера оновлюються відповідно до розміру вікна браузера. Це потрібно для того, щоб 3D-сцена не спотворювалася під час зміни розміру сторінки та залишалася зручною для перегляду.

Оптимізований рендеринг забезпечує плавну навігацію і стабільну роботу вебдодатку

3.8.3 Зменшення навантаження на браузер

Зменшення навантаження на браузер у проєкті досягається за рахунок впорядкованої структури файлів і поділу функцій між окремими скриптами. Сторінка каталогу не завантажує 3D-моделі, а працює лише з даними будівель та зображеннями.

GLB-модель завантажується тільки після переходу користувача на сторінку viewer.html.

Такий підхід дозволяє не перевантажувати головну сторінку зайвими ресурсами. Користувач спочатку переглядає каталог і опис будівлі, а важкі 3D-ресурси завантажуються лише тоді, коли він відкриває 3D-переглядач.

3.9 Адаптація вебдодатку під різні пристрої

Адаптація вебдодатку реалізована засобами CSS [8]. У проєкті передбачено коректне відображення сторінок каталогу, опису будівлі та 3D-переглядача при різних розмірах вікна браузера. Це важливо, оскільки користувач може відкривати вебсистему не лише на великому моніторі, а й на ноутбуці або екрані меншого розміру.

Під час адаптації враховувалось розташування карток каталогу, читабельність тексту на сторінці опису та масштабування області 3D-сцени. Також важливо, щоб кнопки керування залишалися доступними для користувача.

3.9.1 Адаптація сторінок каталогу та опису

На сторінці каталогу картки історичних будівель розміщуються у вигляді сітки. При зміні ширини екрана їх розташування може змінюватися, щоб зберегти зручність перегляду. Це дозволяє користувачу комфортно обирати будівлю навіть у разі зменшення розміру вікна браузера.

Сторінка опису також адаптується до ширини екрана. Зображення та текстові блоки мають залишатися читабельними, а кнопка переходу до 3D-візуалізації повинна бути помітною для користувача.

3.9.2 Адаптація 3D-переглядача

Сторінка 3D-переглядача також адаптується до розміру вікна браузера [6]. Для цього у viewer.js передбачено оновлення параметрів камери та рендерера при зміні розміру вікна. Завдяки цьому 3D-модель не спотворюється, а область сцени залишається коректною.

Це особливо важливо для режиму віртуальної екскурсії, оскільки під час переходу між точками камера повинна зберігати правильний ракурс незалежно від розміру вікна.

3.9.3 Перевірка роботи на різних розмірах екрана

Перевірка адаптивності виконувалась шляхом зміни розміру вікна браузера. Було перевірено головну сторінку каталогу, сторінку опису будівлі та сторінку 3D-візуалізації.

Основна увага приділялась тому, щоб елементи інтерфейсу не виходили за межі екрана, текст залишався читабельним, а 3D-сцена відображалася без спотворень.

На рисунку 3.21 наведено приклад перевірки адаптивності 3D-переглядача на різних розмірах екрана.

```
window.addEventListener("resize", () => {
  camera.aspect = window.innerWidth / window.innerHeight;
  camera.updateProjectionMatrix();
  renderer.setSize(window.innerWidth, window.innerHeight);
});
```

Рисунок 3.21 – Реалізація адаптації 3D-переглядача до розміру екрана

Перевірка може виконуватися за допомогою інструментів розробника у браузері, які дозволяють імітувати різні пристрої.

Також доцільно перевірити систему на реальних екранах, якщо така можливість є.

Результати перевірки використовуються для виправлення помилок і покращення зручності вебдодатку.

3.10 Тестування роботи системи

Після реалізації вебсистеми було виконано тестування основних функцій. Тестування проводилося шляхом локального запуску проєкту через вебсервер і перевірки основних сценаріїв роботи користувача. Для запуску використовувалась команда python -m http.server 8000, після чого вебсистема відкривалась у браузері за адресою http://localhost:8000 [17].

Під час тестування перевірялося відкриття головної сторінки, формування каталогу будівель, перехід до сторінки опису, відкриття 3D-переглядача, завантаження GLB-моделі, робота точок інтересу, інформаційної панелі, режиму віртуальної екскурсії та перемикання теми оформлення.

3.10.1 Функціональне тестування

Функціональне тестування виконувалося для перевірки того, чи працюють основні можливості вебсистеми відповідно до поставлених вимог. Спочатку було перевірено головну сторінку каталогу: вона відкривається у браузері, виводить картки історичних будівель і дозволяє перейти до сторінки опису. Далі було перевірено сторінку building.html. Після вибору будівлі система коректно завантажує її назву, місто, країну, зображення та текстовий опис. Кнопка переходу до 3D-візуалізації відкриває сторінку viewer.html для відповідного об’єкта. Також було перевірено роботу 3D-переглядача: завантаження моделі, керування камерою, відображення точок інтересу, відкриття інформаційної панелі та запуск віртуальної екскурсії.

3.10.2 Тестування інтерфейсу користувача

Під час тестування інтерфейсу перевірялась зручність взаємодії користувача з вебсистемою. Було перевірено, чи зрозуміло користувачу, де

розташований каталог, як обрати будівлю, як перейти до сторінки опису та як запустити 3D-візуалізацію.

На сторінці 3D-переглядача перевірялись елементи керування екскурсією, інформаційна панель і кнопка перемикання теми. Основна увага приділялась тому, щоб ці елементи не заважали перегляду моделі та були доступними під час роботи з 3D-сценою.

3.10.3 Тестування продуктивності

Тестування продуктивності виконувалось під час відкриття 3D-моделей у браузері. Було перевірено, чи завантажується модель, чи відображаються її матеріали, чи працює обертання камери та чи відкриваються точки інтересу без критичних затримок.

Оскільки частина моделей має великий розмір, швидкість завантаження може залежати від характеристик комп’ютера. Проте після завантаження модель доступна для перегляду, а основні функції 3D-переглядача працюють коректно.

3.10.4 Тестування адаптивності

Тестування адаптивності проводилося шляхом зміни розміру вікна браузера.

Було перевірено, як змінюється розташування карток у каталозі, як відображається сторінка опису та чи коректно масштабується 3D-переглядач.

Під час перевірки було встановлено, що основні елементи інтерфейсу залишаються доступними, текст не виходить за межі блоків, а область 3D-сцени адаптується до розміру вікна.

3.11 Аналіз отриманих результатів

У результаті реалізації було створено вебсистему, яка дозволяє користувачу переглядати каталог історичних будівель, відкривати сторінки з описом об’єктів і переходити до інтерактивної 3D-візуалізації. Основні сторінки системи пов’язані між собою через ідентифікатор будівлі, що дозволяє завантажувати відповідні дані, зображення, модель і сценарій екскурсії.

У 3D-переглядачі реалізовано завантаження GLB-моделі, керування камерою за допомогою OrbitControls, точки інтересу, інформаційну панель і режим віртуальної екскурсії. Також реалізовано перемикання світлої та темної теми оформлення.

Отриманий результат підтверджує, що бібліотека Three.js підходить для створення інтерактивної вебсистеми 3D-візуалізації історичних будівель.

Система може використовуватися як демонстраційний або навчальний ресурс, а також може бути розширена новими моделями і маршрутами екскурсій.

3.11.1 Оцінка відповідності системи вимогам

Розроблена вебсистема відповідає основним функціональним вимогам, визначеним у другому розділі.

У системі реалізовано каталог історичних будівель, сторінку опису об’єкта, сторінку 3D-візуалізації, завантаження GLB-моделі, точки інтересу, режим віртуальної екскурсії та перемикання теми інтерфейсу.

Також система відповідає основним нефункціональним вимогам. Вона працює у браузері, має зрозумілу файлову структуру, використовує локальні бібліотеки, може запускатися через локальний вебсервер і дозволяє додавати нові будівлі шляхом розширення даних у проєкті.

3.11.2 Переваги розробленої вебсистеми

Основною перевагою розробленої вебсистеми є поєднання інформаційного вебсайту та інтерактивного 3D-переглядача. Користувач може спочатку ознайомитися з короткою інформацією про будівлю, потім прочитати детальний опис, а після цього перейти до 3D-візуалізації..

Ще однією перевагою є реалізація точок інтересу та віртуальної екскурсії. Завдяки цьому 3D-модель використовується не лише для візуального перегляду, а й для послідовного ознайомлення з окремими частинами історичної будівлі..

Також перевагою є модульна структура проєкту. Дані будівель, тексти описів, сценарії екскурсій, скрипти та моделі зберігаються окремо, що спрощує подальше редагування і розширення системи.

3.11.3 Обмеження та напрями подальшого розвитку

До обмежень розробленої вебсистеми можна віднести залежність продуктивності від розміру 3D-моделей. Великі GLB-файли можуть завантажуватися довше, особливо на слабших пристроях. Тому одним із напрямів подальшого розвитку є додаткова оптимізація моделей і текстур.

У майбутньому систему можна розширити шляхом додавання нових історичних будівель, нових точок інтересу та розширених сценаріїв екскурсій. Також перспективним є додавання аудіосупроводу, підтримки VR-режиму та адміністративної панелі для зручного керування контентом без ручного редагування файлів.

ВИСНОВКИ

У кваліфікаційній роботі було розроблено вебсистему 3D-візуалізації історичної будівлі з функцією віртуальної екскурсії на основі бібліотеки Three.js. У процесі виконання роботи було проаналізовано предметну область, визначено актуальність використання веборієнтованих 3D-технологій для представлення історичних будівель, розглянуто сучасні засоби створення 3D-графіки у браузері та обґрунтовано вибір Three.js як основної технології реалізації.

У першому розділі було розглянуто історичні будівлі як об’єкти цифрової реконструкції, особливості віртуальних музеїв та онлайн-екскурсій, а також сучасні технології 3D-візуалізації у вебсередовищі. Було визначено, що використання 3D-моделей у браузері дозволяє зробити ознайомлення з архітектурними об’єктами більш наочним, інтерактивним і доступним для користувачів незалежно від їхнього місця перебування.

У другому розділі було виконано проєктування вебсистеми. Визначено основні функціональні та нефункціональні вимоги, розроблено загальну архітектуру вебдодатку, описано взаємодію сторінок, структуру файлів, підхід до організації даних, сценарій віртуальної екскурсії та принципи побудови інтерфейсу користувача. Окрему увагу приділено вибору формату GLB/GLTF для 3D-моделей і підготовці матеріалів для використання у вебсередовищі.

У третьому розділі було описано практичну реалізацію вебдодатку на основі Three.js. У системі реалізовано каталог історичних будівель, сторінки з описом об’єктів, інтерактивний 3D-переглядач, завантаження моделей у форматі GLB, керування камерою, точки інтересу, режим віртуальної екскурсії, інформаційні панелі та перемикання світлої і темної теми інтерфейсу. Також розглянуто питання оптимізації продуктивності, адаптації вебдодатку під різні пристрої та тестування основних функцій системи.

У результаті виконання роботи було досягнуто поставленої мети – створено вебсистему для 3D-візуалізації історичної будівлі з можливістю

інтерактивного перегляду та проходження віртуальної екскурсії. Розроблена система дозволяє користувачу ознайомитися з історичними будівлями у зручному форматі, переглянути їх опис, перейти до 3D-сцени, взаємодіяти з моделлю та отримувати додаткову інформацію через точки інтересу.

Практичне значення роботи полягає у тому, що створений вебдодаток може використовуватися в освітніх, демонстраційних і туристичних цілях. Він може бути основою для подальшого розвитку системи цифрового представлення історичних будівель, зокрема шляхом додавання нових 3D-моделей, розширення сценаріїв екскурсій, інтеграції аудіосупроводу, підтримки VR-режиму та створення адміністративної частини для керування контентом.

ПЕРЕЛІК ПОСИЛАНЬ

1. Three.js Documentation. *Three.js*. URL: https://threejs.org/docs (дата звернення: 04.02.2026).

2. Three.js Examples. *Three.js*. URL: https://threejs.org/examples (дата звернення: 04.02.2026).

3. WebGL Overview. *Khronos Group*. URL: https://www.khronos.org/webgl (дата звернення: 04.02.2026).

4. glTF Overview. *Khronos Group*. URL: https://www.khronos.org/gltf (дата звернення: 11.02.2026).

5. WebGL API. *MDN Web Docs*. URL: https://developer.mozilla.org/en-US/docs/Web/API/WebGL_API (дата звернення: 11.02.2026).

6. JavaScript. *MDN Web Docs*. URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript (дата звернення: 02.03.2026).

7. HTML: HyperText Markup Language. *MDN Web Docs*. URL: https://developer.mozilla.org/en-US/docs/Web/HTML (дата звернення: 02.03.2026).

8. CSS: Cascading Style Sheets. *MDN Web Docs*. URL: https://developer.mozilla.org/en-US/docs/Web/CSS (дата звернення: 17.03.2026).

9. GLTFLoader. *Three.js Documentation*. URL: https://threejs.org/docs/#examples/en/loaders/GLTFLoader (дата звернення: 20.03.2026).

10. OrbitControls. *Three.js Documentation*. URL: https://threejs.org/docs/#examples/en/controls/OrbitControls (дата звернення: 20.03.2026).

11. WebXR Device API. *MDN Web Docs*. URL: https://developer.mozilla.org/en-US/docs/Web/API/WebXR_Device_API (дата звернення: 09.04.2026).

12. Babylon.js Documentation. *Babylon.js*. URL: https://doc.babylonjs.com (дата звернення: 13.04.2026).

13. A-Frame Documentation. *A-Frame*. URL: https://aframe.io/docs (дата звернення: 13.04.2026).

14. Blender Manual. *Blender Documentation*. URL: https://docs.blender.org/manual/en/latest (дата звернення: 04.05.2026).

15. Google Arts & Culture. *Google*. URL: *https://artsandculture.google.com* (дата звернення: 16.04.2026).

16. Sketchfab. *Sketchfab*. URL: https://sketchfab.com (дата звернення: 16.04.2026).

17. Репозиторій вебсистеми 3D-візуалізації історичної будівлі. *GitHub*. URL: https://github.com/takashixgod/diploma-3d-viewer (дата звернення: 21.05.2026).

18. Culture and Digital Technologies. *UNESCO*. URL: https://www.unesco.org/en/culture-and-digital-technologies (дата звернення: 20.04.2026).

19. Virtual reality tours. *Google Arts & Culture*. URL: https://artsandculture.google.com/project/virtual-tours (дата звернення: 18.05.2026).

20. Google Arts & Culture. URL: https://artsandculture.google.com (дата звернення: 09.05.2026).

21. Europeana : Discover Europe’s digital cultural heritage. *Europeana*. URL: https://www.europeana.eu/en (дата звернення: 18.05.2026).

22. The Europeana platform. *European Commission*. URL: https://digital-strategy.ec.europa.eu/en/policies/europeana (дата звернення: 18.05.2026).

23. Explore 3D Models. *Sketchfab*. URL: https://sketchfab.com/3d-models (дата звернення: 18.05.2026).

ДОДАТОК А

Структура вебсистеми 3D-візуалізації історичних будівель

У додатку наведено структуру розробленої вебсистеми 3D-візуалізації історичних будівель з функцією віртуальної екскурсії. Проєкт містить HTML-сторінки, CSS-файли, JavaScript-файли, локальні бібліотеки Three.js, 3D-моделі, зображення та файли з даними для опису будівель і точок інтересу.

А.1 Основні сторінки вебсистеми

До основних сторінок вебсистеми належать:

– index.html – головна сторінка вебсистеми з каталогом історичних будівель;
– building.html – сторінка детального опису обраної історичної будівлі;
– viewer.html – сторінка 3D-візуалізації з інтерактивною сценою Three.js.

Головна сторінка забезпечує перегляд каталогу будівель. Сторінка building.html використовується для відображення опису, зображень і кнопки переходу до 3D-переглядача. Сторінка viewer.html відповідає за завантаження 3D-моделі, роботу точок інтересу та режим віртуальної екскурсії.

А.2 Каталоги ресурсів проєкту

У структурі вебсистеми використовуються такі каталоги:

– css – файли стилів для оформлення сторінок;
– js – JavaScript-файли з логікою роботи системи;
– libs – локальні бібліотеки Three.js та допоміжні модулі;

– models – 3D-моделі історичних будівель у форматі GLB;
– images – зображення для каталогу та сторінок опису;
– data – файли з описами будівель, точками інтересу та сценаріями екскурсій.

Такий поділ дозволяє зберігати ресурси впорядковано та спрощує подальше редагування вебсистеми.