Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет комп'ютерних наук
(повна назва)

Кафедра програмної інженерії
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Дослідження методів побудови 3D моделей у веб-додатках за допомогою React

Виконав:
студент (ка) 2 курсу, групи ІПЗм-22-4

Гладченко О. О.

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

Тип програми освітньо-наукова

Керівник доц. каф. ПІ Каук В. І.
(посада, прізвище, ініціали)

Допускається до захисту
Зав. кафедри _____ З.В.Дудар
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет комп'ютерних наук
Кафедра програмної інженерії
Рівень вищої освіти другий (магістерський)
Спеціальність 121 – Інженерія програмного забезпечення
Тип програми освітньо-наукова програма
Освітня програма Інженерія програмного забезпечення
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«___» _____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Гладченку Олександру Олександровичу

1. Тема роботи «Дослідження методів побудови 3D моделей у веб-додатках за допомогою React»

Затверджена наказом по університету від 20 03 2024 р. № 250Ст

2. Термін подання студентом роботи до екзаменаційної комісії 12.06.2024

3. Вихідні дані до роботи Вихідні дані до роботи дослідити методи побудови 3D моделей за допомогою React, середовище розробки VS Code 2024, мова програмування JavaScript, бібліотека React.js, бібліотеки для побудови 3D моделей: Three.js, Babylon.js, Whitestorm.js, A-Frame.

4. Перелік питань, що потрібно опрацювати в роботі

мета роботи, аналіз предметної галузі і постановка задачі, огляд та аналіз літературних джерел з дослідження, дослідження теоретичне, дослідження практичне.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Аналіз предметної галузі	01.05.2024 – 03.05.2024	*Виконано*
2	Розробка постановки задачі	04.05.2024 – 06.05.2024	*Виконано*
3	Розробка ПЗ	07.05.2024 – 10.05.2024	*Виконано*
4	Проведення експерименту	13.05.2024 – 14.05.2024	*Виконано*
5	Оформлення пояснювальної записки	15.05.2024 – 27.05.2024	*Виконано*
6	Підготовка презентації та доповіді	28.05.2024 – 29.05.2024	*Виконано*
7	Попередній захист	05.06.2024	*Виконано*
8	Нормоконтроль, рецензування	06.06.2024	*Виконано*
9	Здача роботи у електрониий архів	10.06.2024	*Виконано*
10	Допуск до захисту у зав. кафедри	11.06.2024	*Виконано*

Дата видачі завдання 27.04.2024 р.

Студент _____ Гладченко О. О.
(підпис)

Керівник роботи _____ доц. каф. ПІ, Каук В. І.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ / ABSTRACT

Робота містить: 59 с., 12 рис., 3 табл., 21 джер, 4 додатки.

ВЕБ-ДОДАТКИ, ГРАФІЧНИЙ ДВИГУН, КОРИСТУВАЦЬКИЙ ДОСВІД, ПРОДУКТИВНІСТЬ, ТРИВИМІРНІ МОДЕЛІ, REACT.

Робота присвячена дослідженню методів побудови тривимірних (3D) моделей у веб-додатках за використання React. Зростання популярності веб-технологій та відмінності у сприйнятті контенту споживачами визначають актуальність впровадження тривимірних візуальних елементів у веб-інтерфейсах. В цьому контексті досліджуються технології та методи, що дозволяють побудову 3D моделей у веб-додатках із використанням бібліотеки React.

Об'єктом дослідження є різноманітні підходи до побудови 3D моделей, які базуються на можливостях React. Детально розглядаються традиційні методи побудови 3D графіки, такі як використання WebGL, а також новітні техніки, що використовуються у сполученні з React, такі як React Three Fiber. Аналізуються принципи функціонування цих методів та їх відмінності.

У роботі проводиться порівняльний аналіз ефективності та можливостей різних методів побудови 3D моделей у веб-додатках за допомогою React. Особлива увага приділяється аспектам продуктивності, масштабованості та зручності використання цих методів в реальних веб-проектах.

Отримані результати та висновки роботи визначають перспективи та проблеми у використанні різних методів побудови 3D моделей у веб-додатках на базі React. Робота має практичне значення для розробників веб-додатків та науково-дослідницької галузі, сприяючи розумінню та вдосконаленню технологічних аспектів використання тривимірної графіки у веб-програмуванні.

WEB APPLICATIONS, GRAPHICS ENGINE, USER EXPERIENCE, PRODUCTIVITY, 3D MODELS, REACT.

The work is devoted to researching methods of building three-dimensional (3D) models in web applications using React. The growing popularity of web technologies and differences in the perception of content by consumers determine the relevance of the implementation of three-dimensional visual elements in web interfaces. In this context, the technologies and methods that allow the construction of 3D models in web applications using the React library are investigated.

The object of research is various approaches to building 3D models, which are based on the capabilities of React. Traditional methods of building 3D graphics, such as using WebGL, as well as newer techniques used in conjunction with React, such as React Three Fiber, are covered in detail. The principles of functioning of these methods and their differences are analyzed.

The paper provides a comparative analysis of the effectiveness and capabilities of various methods of building 3D models in web applications using React. Particular attention is paid to aspects of performance, scalability and usability of these methods in real web projects.

The obtained results and conclusions of the work determine prospects and problems in the use of various methods of building 3D models in React-based web applications. The work has practical implications for web application developers and the research industry, contributing to the understanding and improvement of technological aspects of the use of 3D graphics in web programming.

Умови публікації звіту: заява щодо самостійного виконання кваліфікаційної роботи та можливості її публікації в електронному архіві відкритого доступу ElArKhNURE.

Я,

Гладченко Олександр Олександрович

(прізвище, ім’я, по-батькові)

студент групи ІПЗм-22-4, здобувач освіти на другому (магістерському) рівні кафедра програмної інженерії ,

(повна назва кафедри)

заявляю: моя кваліфікаційна робота на тему

Дослідження методів побудови 3D моделей у веб-додатках за допомогою React ,

(назва роботи)

що буде представлена в екзаменаційну комісію для публічного захисту, виконана самостійно, в ній не містяться елементи плагіату і вона може бути опублікована в електронному архіві відкритого доступу ElArKhNURE. Всі запозичення з друкованих та електронних джерел мають відповідні посилання.

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

Дата Підпис

ЗМІСТ

ВСТУП

У сучасному світі веб-розробки, де конкуренція в галузі інтернет-технологій стає все більш високою, розуміння та впровадження передових методів побудови тривимірних (3D) моделей у веб-додатках стає ключовим завданням для розробників та компаній. Інтеграція 3D елементів у веб-інтерфейси стає необхідністю в контексті зростання очікувань користувачів та відмінностей у сприйнятті контенту.

Досліджуються різноманітні аспекти методів побудови 3D моделей у веб-додатках, включаючи класичні техніки та новаторські підходи, які дозволяють вбудовувати тривимірні об'єкти у веб-середовище. Зокрема, акцент зроблений на технологіях, що сприяють створенню та відображенню 3D моделей, а також їх вплив на користувацький досвід та продуктивність веб-додатків.

Особлива увага приділяється аспектам оптимізації відображення 3D графіки в реальному часі, можливостям анімації та взаємодії з тривимірними об'єктами на веб-сторінці. Надається огляд та порівняння різних підходів до побудови 3D моделей у веб-додатках, зокрема з використанням інструментів React.

Важливим аспектом є визначення впливу використання 3D графіки з React на продуктивність веб-додатків та вирішення можливих труднощів, що можуть виникнути в процесі розробки. Дослідження фокусується не лише на теоретичних аспектах, але й на практичних викликах та можливостях використання цих методів у реальних веб-проектах.

Обрана тема є актуальною як для виробництва, так і для розвитку науки в сучасному веб-середовищі. Результати даної роботи можуть відігравати ключову роль в прийнятті більш якісних рішень у виборі методів побудови 3D моделей та стати важливим кроком у розвитку цієї технологічної галузі.

Метою даного дослідження є визначення найбільш оптимізованих методів для побудови 3D моделей у веб-додатках, зокрема використовуючи бібліотеку React.

Робота спрямована на аналіз та порівняння існуючих методів, визначення їхнього впливу на продуктивність та розробку веб-додатків.

Для досягнення поставленої мети необхідно вирішити ряд завдань, включаючи аналіз існуючих методів побудови 3D моделей, визначення метрик для оцінювання їх продуктивності, розробку веб-застосунку для проведення експерименту, вимірювання та підрахунок значень метрик для кожного методу, а також надання рекомендацій щодо їхнього використання.

Об'єктом дослідження є веб-додатки, які використовують 3D моделі та розроблені за допомогою бібліотеки React. Предметом дослідження є методи побудови 3D моделей для цих веб-додатків.

Методи дослідження включають аналіз різних підходів до побудови 3D моделей, вимірювання розмірів вихідного коду, підрахунок метрик продуктивності та використання розробницької панелі приладів у браузері для вимірювання швидкодії.

Отримані результати можуть служити підставою для вибору оптимального методу побудови 3D моделей у веб-додатках, а також стануть важливим внеском у подальші дослідження у цій галузі. Практичне застосування виявиться в розробці веб-додатків з використанням 3D графіки для поліпшення користувацького досвіду та розширення можливостей веб-розробників.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Загальний аналіз області

До появи анімації веб-програми були здебільшого статичними і складалися з фіксованих елементів, тексту та зображень. Основний акцент був зроблений на поданні інформації та наданні користувачеві необхідного контенту. Веб-сайти складалися з коду HTML і, можливо, використовували CSS для базового дизайну [1].

Однією з основних характеристик веб-додатків того часу була їх статичність. Сторінки були фіксовані, інтерактивність була обмежена переходами між статичними сторінками (див. рис. 1.1).

Рисунок 1.1 – Веб сторінки з використанням HTML та CSS

Більшість можливостей анімації обмежувалися зміною стану елементів (таких як текст або колір фону) під час взаємодії користувача [2].

З появою CSS і JavaScript анімація відкрила нові можливості для розробки більш динамічних і привабливих веб-додатків. Анімація дозволила створювати плавні переходи, підвищуючи зручність використання та створюючи краще візуальне враження. Однак ранні веб-програми зазвичай не використовували

анімацію так широко, як сучасні програми, і обмежувалися кількома базовими ефектами.

З появою та розвитком потужніших технологій, таких як HTML5, CSS3 та вдосконалених механізмів JavaScript, анімація веб-додатків стала вдосконаленішою, динамічнішою та креативнішою [3]. Сучасні веб-додатки використовують різні анімаційні ефекти для створення привабливого та інтерактивного інтерфейсу користувача (див. рис. 1.2).

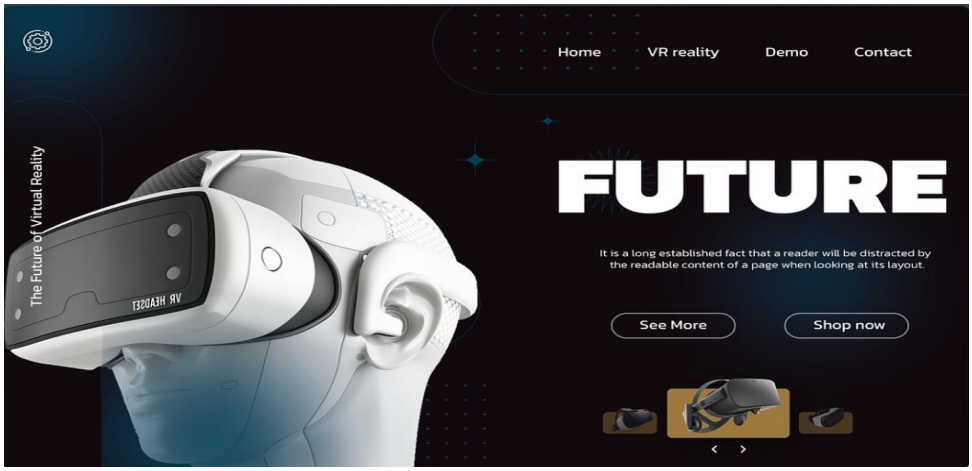

Рисунок 1.2 – Веб сторінки з використанням HTML5, CSS3 та JS

У сучасному інтернет-просторі, де веб-технології постійно розвиваються, виникає реальна необхідність побудови 3D моделей у веб-додатках для покращення користувацького досвіду.

До недавнього часу використання 3D-зображень було здебільшого обмежене сферою ігрової індустрії та графічного дизайну. Однак із стрімким розвитком веб-технологій та постійно зростаючими потребами користувачів, все частіше постає питання щодо ефективності впровадження та можливостей інтеграції тривимірної графіки у веб-додатки. Ці технологічні вдосконалення відкривають нові перспективи для використання 3D-зображень, виходячи далеко за межі традиційних застосувань, і дозволяють використовувати їх у найрізноманітніших веб-середовищах, задовольняючи попит на більш інтерактивні та візуально привабливі інтерфейси.

React, як потужний і популярний інструмент для розробки інтерфейсів користувача, також має свої можливості для роботи з 3D графікою. Один з можливих підходів полягає у використанні спеціалізованих бібліотек та компонентів, які дозволяють інтегрувати 3D моделі безпосередньо у React-додатки [4]. Це забезпечує розробникам можливість створювати інтерактивні та динамічні користувацькі інтерфейси, що включають тривимірні елементи. Використовуючи такі інструменти, як React Three Fiber, розробники можуть легко вбудовувати 3D-сцени у свої додатки, зберігаючи при цьому всі переваги роботи з React, такі як компонентний підхід та ефективне управління станом додатку.

Застосування 3D моделей у веб-додатках відкриває широкий спектр можливостей і забезпечує вражаючі візуальні ефекти, підвищуючи користувацький досвід. Інтеграція тривимірної графіки дозволяє створювати більш інтерактивні, динамічні та привабливі інтерфейси, що сприяє залученню користувачів і покращує їх взаємодію з веб-додатками. Від віртуальних турів і онлайн-конфігураторів до інтерактивного навчання і розваг, 3D-технології надають розробникам потужні інструменти для реалізації інноваційних ідей і проектів (див. рис. 1.3).

Рисунок 1.3 – Веб сторінки з використанням 3D моделей

Основні галузі, де використовують 3D моделі у веб-додатках [5]:

- ігрова індустрія (однією з основних галузей застосування 3D моделей у веб-додатках є ігрова індустрія. Відтворення складних ігрових сценаріїв та персонажів у 3D дозволяє створювати захоплюючі та реалістичні ігрові враження, а додавання цього функціоналу у веб-додатки розширює доступність гравцям);
- онлайн магазини (використання 3D моделей у таких магазинах дозволяє користувачам докладно розглядати продукти перед покупкою. Наприклад, роздивитися товар з різних боків перед покупкою);
- освіта (у веб-додатках для навчання 3D моделі можуть використовуватись для візуалізації складних концепцій. Наприклад, вивчення анатомії, хімічних молекул чи історичних подій стає більш ефективним завдяки можливості обертати та роздивлятися моделі з усіх кутів);
- архітектура та дизайн (для веб-додатків, які пропонують послуги з дизайну або архітектурні концепції, 3D моделі дозволяють клієнтам краще уявити та оцінити проекти перед їх реалізацією);
- медицина (у веб-додатках для медичних цілей 3D моделі можуть використовуватись для детального вивчення структур органів чи для навчання майбутніх медиків);
- маркетинг та реклама (використання 3D графіки у веб-додатках для створення рекламних матеріалів чи вражаючих візуальних ефектів допомагає залучити увагу користувачів та відокремити продукт чи бренд від конкурентів);
- технічна візуалізація (у веб-додатках для технічних галузей, таких як інженерія чи будівництво, 3D моделі можуть використовуватись для візуалізації складних технічних об'єктів та процесів).

Тому для того, щоб покращити користувацький досвід у всіх вище зазначених галузях, необхідно дослідити та розібратися, яким чином та за допомогою якої бібліотеки чи фреймворку найкраще це зробити нам, як розробникам програмного забезпечення. Це включає в себе розуміння потреб кожної конкретної галузі та вибір відповідних інструментів, які допоможуть реалізувати ці потреби найефективнішим чином.

1.2 Виявлення проблем

Виявлення проблем в процесі дослідження цього питання стає ключовим етапом. Наприклад, можливі труднощі в ефективному взаємодії між React та 3D графікою, що вимагає уважного аналізу та вирішення. Ось деякі можливі проблеми:

- зіткнення контекстів (використання різних контекстів для React та 3D графіки може викликати конфлікти та ускладнити їх коректну інтеграцію);
- способи маніпуляції DOM (React використовує віртуальний DOM, а 3D графіка може працювати безпосередньо з реальним DOM. Це може створювати проблеми при зміні стану або елементів, які взаємодіють з 3D об'єктами);
- великі обсяги даних (3D моделі можуть бути великими за розміром, що впливає на завантаження та продуктивність. Ефективна передача та оптимізація цих даних є ключовим аспектом);
- оптимізація відображення (рендеринг великої кількості деталей 3D моделей може впливати на продуктивність, особливо на пристроях із обмеженими ресурсами, тому потрібно шукати ефективні підходи до оптимізації відображення);
- синхронізація стану React і 3D моделі (забезпечення взаємодії між станом React та станом 3D моделі може бути викликаною задачею. Синхронне оновлення стану між цими двома середовищами важливо для коректного відображення даних та змін);

– обробка подій (організація взаємодії користувача з 3D об'єктами та правильна обробка подій, таких як кліки чи перетягування, може вимагати специфічних підходів для забезпечення комфортного користувацького досвіду).

Процес розробки 3D моделей може стати важливим етапом впровадження нових функцій у веб-додатки, і важливо визначити ефективні шляхи вирішення зазначених проблем.

Дослідження також може включати аналіз різних підходів до побудови 3D об'єктів, їх відображення та взаємодії з користувачем. Реалізація 3D графіки вимагає вивчення потужних бібліотек та фреймворків, які можуть допомогти оптимізувати процес створення та взаємодії з цифровим простором.

1.3 Постановка задачі

З розвитком веб-технологій виникає потреба в удосконаленні користувацького досвіду за допомогою тривимірних моделей. Використання бібліотеки React для побудови інтерфейсу забезпечує зручний шлях створення веб-додатків, тому дослідження методів побудови 3D моделей залишається актуальним напрямком.

В рамках дослідження необхідно вирішити наступні завдання:

– розглянути і проаналізувати існуючі методів побудови 3D моделей у веб-додатках;
– визначення їх основні переваги та обмеження;
– дослідження можливості інтеграції 3D графіки з бібліотекою React;
– визначити найкращі практики та оптимальні шляхи взаємодії React з 3D моделями;
– виміряти продуктивність кожного методу в умовах веб-додатка з використанням React;

- порівняти швидкодії та використання ресурсів під час відображення та маніпуляцій з 3D моделлю;
- провести аналіз взаємодії користувача з різними типами 3D моделей в контексті веб-додатків на React;
- визначити оптимальні методи для користувацької взаємодії з 3D об'єктами;
- сформувати рекомендації для розробників щодо вибору найбільш підходящих методів побудови 3D моделей у веб-додатках на основі отриманих результатів.

Дослідження має надати вичерпну інформацію щодо ефективності та можливостей різних методів побудови 3D моделей у веб-додатках з використанням React, що дозволить розробникам та командам проектів здійснити обміркований вибір та покращити взаємодію користувачів з веб-додатками.

2 АНАЛІЗ МЕТОДІВ ПОБУДОВИ 3D МОДЕЛЕЙ

2.1 Опис існуючих методів побудови 3D моделей

При розробці веб-додатків, які потребують використання 3D-графіки, існує кілька методів побудови моделей, які можна успішно застосовувати. Перш ніж вибрати конкретний метод, важливо зрозуміти їхні особливості та відмінності. Один з найпоширеніших підходів – це використання бібліотек, таких як Three.js або Babylon.js, які надають програмістам інструменти для інтеграції 3D-графіки безпосередньо у веб-додатки. Це дозволяє створювати вражаючі візуальні ефекти та інтерактивність, що поліпшує користувацький досвід.

Деякі розробники використовують спеціалізовані фреймворки, такі як A-Frame, які дозволяють створювати віртуальну реальність у веб-додатках, використовуючи простий HTML-подібний синтаксис. Крім того, для складних 3D-моделей можна використовувати програми для моделювання, такі як Blender або Maya, що надають розширені можливості для створення та редагування моделей.

Розуміння різних методів побудови 3D-моделей допомагає розробникам обирати найбільш підходящий під їхні потреби та характер проекту. Нижче наведено огляд найпопулярніших методів: Three.js, Babylon.js, Whitestorm.js та A-Frame.

Всі ці бібліотеки підтримують використання основних компонентів, необхідних для рендерингу 3D сцени:

- сцена;
- камера;
- рендерер;
- об’єкти (meshes);
- світло;
- анімація;
- фізика.

Кожен з цих методів має свої унікальні переваги та недоліки, що робить їх придатними для різних завдань і проектів. Вибір конкретного методу залежить від специфіки проекту, технічних вимог і досвіду розробників.

2.2 Three.js

Це бібліотека для JavaScript, яка дозволяє легко створювати та взаємодіяти з тривимірними графічними сценами у веб-додатках (див. рис. 2.1).

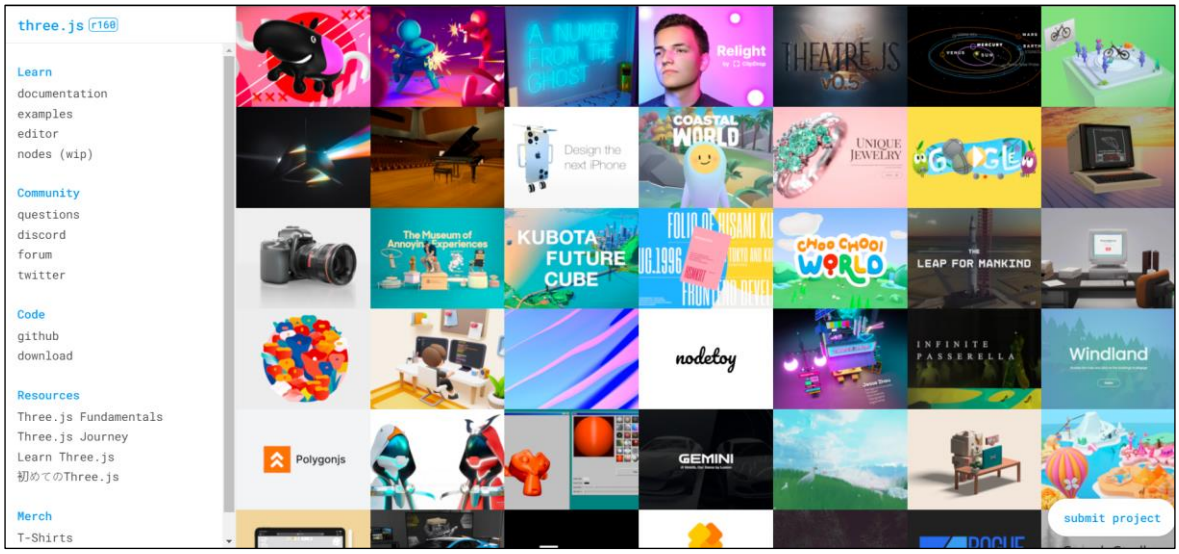

Рисунок 2.1 – Офіційна сторінка Three.js

Заснована на мові програмування JavaScript. Three.js використовує можливості WebGL для відтворення тривимірних графічних об'єктів у браузері [6].

Однією з головних завдань Three.js є полегшення створення тривимірних сцен, об'єктів та анімацій у веб-додатках, надаючи розробникам простий та ефективний інтерфейс.

Three.js має добре структуровану документацію, яка надає повну і структуровану інформацію про всі аспекти використання Three.js, що допомагає розробникам ефективно використовувати всі можливості бібліотеки.

У документації зазвичай розглядаються такі питання, як встановлення бібліотеки, базові концепції 3D-графіки, створення та маніпулювання об'єктами,

робота з матеріалами та текстурами, анімація, освітлення, взаємодія з мишкою та клавіатурою, а також розширені теми, такі як використання фізики, шейдерів і VR.

Документація Three.js зазвичай включає в себе приклади коду, які демонструють різні функціональності та техніки роботи з бібліотекою. Це допомагає розробникам краще зрозуміти, як використовувати той чи інший фрагмент коду у своїх проектах.

Основні переваги Three.js:

- простота використання (Three.js забезпечує високорівневий API, який дозволяє розробникам створювати тривимірні сцени без глибокого розуміння внутрішньої роботи WebGL);
- кросбраузерність (Three.js автоматично вирішує проблеми, пов'язані з різними реалізаціями WebGL у різних браузерах, забезпечуючи стабільну роботу на різних платформах);
- широкі можливості (бібліотека пропонує різноманітні можливості для створення ефектних тривимірних об'єктів, світла, тіней, текстур, анімації тощо);
- активна спільнота (Three.js користується великою та активною спільнотою розробників, яка постійно розвиває та підтримує цю бібліотеку);
- відкритий код (Three.js є відкритим проектом, що дозволяє розробникам вносити свій внесок, а також адаптувати бібліотеку під свої потреби).

Основні недоліки Three.js:

- великий розмір файлів (Three.js може додати певний об'єм до розміру веб-додатка, особливо якщо використовується повний набір його можливостей);
- високий поріг входження (для новачків у тривимірній графіці може виникнути певна крутизна вивчення Three.js та основ WebGL);

– важкість оптимізації (при роботі з великою кількістю тривимірних об'єктів можуть виникати проблеми з продуктивністю, і оптимізація коду може стати нетривіальною задачею).

Незважаючи на всі недоліки, Three.js залишається однією з найпопулярніших та потужних бібліотек для створення тривимірних сцен у веб-додатках і використовується для створення ігор, віртуальної реальності, візуалізації даних тощо. Вона забезпечує широкий спектр можливостей для рендерингу складних сцен, підтримує різні формати моделей і текстур, а також має активну спільноту розробників, яка постійно оновлює та вдосконалює бібліотеку. Завдяки своїй гнучкості та багатофункціональності, Three.js залишається вибором багатьох професіоналів та ентузіастів у сфері веб-розробки [7].

2.3 Babylon.js

Це потужна веб-бібліотека для створення 3D-графіки у веб-додатках. Забезпечує можливості реалістичного відтворення графіки та анімацій у веб-середовищі (див. рис. 2.2).

Рисунок 2.2 – Офіційна сторінка Babylon.js

Заснована на JavaScript, вона надає розробникам інструменти для створення захоплюючих візуальних ефектів, ігор та віртуальних середовищ.

Висока продуктивність, велика спільнота та підтримка широкого спектру платформ роблять Babylon.js популярним вибором для веб-розробників, які працюють з 3D-графікою та відмінно впишуться в проекти, де важлива динамічна та іммерсивна візуалізація [8].

Основні переваги Babylon.js [9]:

- багатофункціональність (Babylon.js надає широкий набір інструментів для створення різноманітних тривимірних сцен та об'єктів. Вона охоплює всі аспекти розробки графіки, включаючи світло, тіні, матеріали та анімації);
- WebGL та WebXR (Babylon.js використовує WebGL для виконання графічних обчислень на рівні апаратного забезпечення. Також підтримує WebXR для взаємодії з віртуальною та доповненою реальністю);
- простота використання (хоча вона є потужною, бібліотека намагається зробити розробку тривимірних додатків якомога найпростішою для розробників. Інтуїтивний API полегшує створення складних ефектів);
- анімації та взаємодія (Babylon.js дозволяє створювати динамічні анімації та взаємодію з об'єктами, що допомагає реалізовувати різноманітні ігрові та інтерактивні сценарії);
- інтеграція з іншими інструментами (підтримка інших інструментів, таких як Blender, 3ds Max, та інших, спрощує імпорт та використання зовнішніх моделей);
- документація та спільнота (Babylon.js має добре документовану API та активну спільноту розробників, яка надає підтримку та різноманітні приклади).

Серед основних недоліків можна виділити розмір файлів для відображення, зазвичай має великий розмір файлів, що може вплинути на час завантаження веб-сторінок.

2.4 Whitestorm.js

Це бібліотека для розробки 3D-графіки у веб-додатках, зосереджена на спрощенні створення інтерактивних та анімованих 3D-сцен. Забезпечує розробникам потужні інструменти для створення віртуальних об'єктів, анімацій та взаємодії. Whitestorm.js базується на JavaScript і використовує WebGL для оптимальної відтворення 3D-графіки у веб-браузерах [10].

Завдяки своїй легкості використання та розширюваності вона стає важливим інструментом для веб-розробників, які прагнуть впровадити інноваційні та ефектні 3D-елементи у свої проекти (див. рис. 2.3).

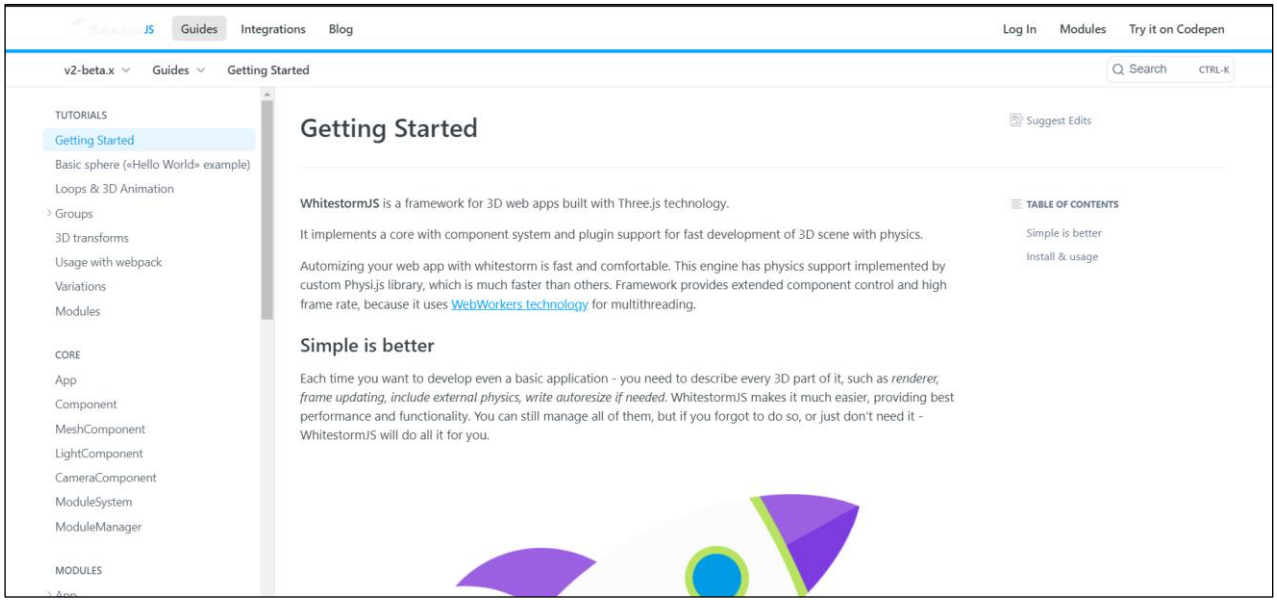

Рисунок 2.3 – Офіційна сторінка Whitestorm.js

Основні переваги Whitestorm.js:

– легкість використання (Whitestorm.js має простий та легкий у використанні API, що полегшує створення 3D сцен та об'єктів);

- зручна робота з веб-технологіями (бібліотека добре інтегрується з іншими веб-технологіями, такими як WebGL, що дозволяє використовувати її у веб-проектах);
- сучасний функціонал (Whitestorm.js надає потужні можливості для створення сучасних 3D-додатків, таких як фізичне моделювання, анімація та обробка подій);
- підтримка VR та AR (це робить Whitestorm.js відмінним вибором для розробки 3D додатків у цих областях);
- велика спільнота (Whitestorm.js має активну спільноту користувачів та розробників, що сприяє обміну досвідом та розвитку проєкту).

Основні недоліки Whitestorm.js:

- обмежена документація (деякі користувачі вказують на обмеженість документації, що може зробити важчим вивчення та використання деяких аспектів Whitestorm.js);
- менша розширюваність (у порівнянні з іншими бібліотеками, такими як Three.js, Whitestorm.js може мати менше розширюваності та менше готових рішень для деяких завдань);
- менша популярність (оскільки Whitestorm.js не так відомий, як деякі інші бібліотеки, може бути важче знайти велику кількість додаткових матеріалів, прикладів чи рішень спільноти);
- можливі проблеми з сумісністю (з огляду на розробку та оновлення, можливі проблеми зі сумісністю з новими версіями браузерів або інших бібліотек).

2.5 A-Frame

A-Frame – це відкритий фреймворк для створення віртуальної реальності (VR) та 3D-сцен у веб-браузерах (див. рис. 2.4).

Розроблений командою Mozilla, A-Frame базується на веб-технологіях, таких як HTML, CSS та JavaScript, і надає простий та декларативний спосіб створення 3D сцен без необхідності глибоких знань графіки чи програмування віртуальної реальності. За допомогою пакету aframe-react його можна легко інтегрувати в React-застосунок [11].

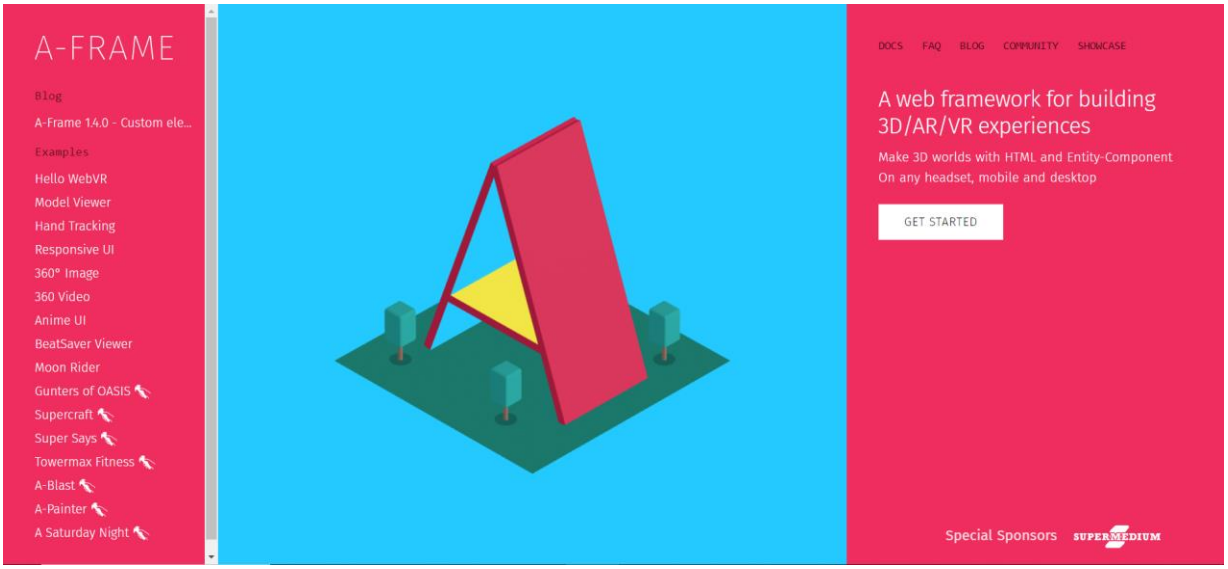

Рисунок 2.4 – Офіційна сторінка A-Frame

Основні переваги A-Frame [12]:

- декларативний синтаксис (A-Frame використовує простий HTML-подібний синтаксис для опису сцен та об'єктів у 3D просторі. Це полегшує створення та редагування сцен без глибокого розуміння коду);
- кросс-платформений (фреймворк підтримує багато платформ, включаючи веб-браузери для комп'ютерів, мобільні пристрої та різні VR-пристрої, такі як Oculus Rift, HTC Vive та Samsung Gear VR);
- розширені можливості VR (A-Frame дозволяє інтегрувати різноманітні компоненти VR, такі як контролери, анімації рухів та інші, що робить його потужним інструментом для розробки VR-застосунків);
- можливості розширення (фреймворк має систему компонентів, яка дозволяє легко розширювати можливості сцен. Компоненти можуть бути використані

для додавання функціональності, такої як анімації, освітлення, фізика та іншого);

– спільнота та екосистема (A-Frame має активну спільноту розробників, яка допомагає вирішувати проблеми, надає підтримку та спільно працює над розвитком фреймворку. Також існує ряд розширень та готових компонентів, які можна використовувати для розширення можливостей).

Основні недоліки A-Frame:

– обмежена функціональність (у порівнянні з іншими, більш потужними графічними двигунами, A-Frame може мати обмежені можливості для великих та складних проектів);
– швидкість та оптимізація (деякі проекти можуть потребувати додаткової оптимізації для досягнення високої швидкодії, особливо на старіших пристроях);
– не ідеально для складних графічних ефектів (для проектів, що вимагають складних графічних ефектів, може бути виправданіше використовувати більш потужні графічні двигуни).

3 ВИЗНАЧЕННЯ МЕТОДИКИ ПРОВЕДЕННЯ ДОСЛІДЖЕННЯ

Визначення методики проведення дослідження є критичним етапом, оскільки воно визначає рамки, способи та підходи до вирішення конкретної наукової чи технічної проблеми. Цей процес є важливим з кількох причин, таких як чіткі цілі та завдання, ефективне планування, вибір методології, обґрунтування доцільності, контроль якості, визначення об'єктів дослідження, забезпечення об'єктивності, та надання рамок для аналізу.

Методика дослідження може бути визначена наступним чином:

- визначення об'єктивів (сформулювати конкретні цілі та завдання дослідження та розкрити основні аспекти, які потрібно вивчити, такі як ефективність, продуктивність, можливості інтеграції з React тощо);
- вибір методології (визначити методологію, яка буде використана для збору та аналізу даних. Можливі методології включають емпіричне дослідження, порівняльний аналіз, експерименти та польові випробування);
- вибір об'єктів дослідження (визначити конкретні бібліотеки, фреймворки чи підходи для створення 3D моделей у веб-додатках, які будуть об'єктами дослідження);
- створення програмного забезпечення для експериментів (розробити веб-додаток або використовувати існуючий для реалізації 3D моделі за допомогою React та вибраних технологій);
- вимірювання та збір даних (визначити метрики для вимірювання ефективності та продуктивності, такі як швидкість завантаження, використання ресурсів, реакція на взаємодію користувача тощо та здійснювати вимірювання та збір даних під час роботи веб-додатка);
- проведення експериментів (запускати експерименти з різними методами побудови 3D моделей та реалізацією у веб-додатку та фіксувати результати експериментів та забезпечити їх об'єктивність);

– аналіз та висновки (провести аналіз отриманих даних та порівняти різні методи побудови 3D моделей та зробити висновки щодо ефективності, можливостей та обмежень кожного методу).

Основна ціль експерименту – порівняння методів побудови 3D моделей за допомогою React та визначити найкращий як для користувача, так і для розробника.

3.1 Критерії оцінювання методів

Важливим етапом проведення експерименту є визначення критеріїв, за якими будуть оцінюватись представлені бібліотеки та фреймворки [13]. Це дозволить чітко визначити переваги та недоліки кожного методу.

Для цього скористаємося лінійною адаптивною згорткою з ваговими коефіцієнтами.

Перевагами лінійної адаптивної згортки є можливість пристосовуватися до різних масштабів у даних. Це дозволяє ефективно виявляти шаблони як на великих, так і на малих відрізках даних, що може бути критичним для повного аналізу наших вхідних даних. А також використання нормуючих показників допомагає уніфікувати шкалу вхідних даних, що є важливим аспектом у випадку, коли у нас є дані з різних джерел чи з різних частин системи [14].

У якості критеріїв для порівняння технологій, було обрано наступні параметри:

– продуктивність (важливим критерієм є швидкість та продуктивність в реальному часі, оскільки 3D графіка може вимагати значних обчислювальних ресурсів [15]. Оцінка продуктивності дозволить визначити, наскільки ефективно кожен метод працює при створенні та відображенні 3D моделей);
– швидкість відгуку (важливим для користувачів є час, необхідний для відгуку системи на їхні дії. Швидка реакція особливо важлива у веб-додатках для забезпечення позитивного враження користувачів);

- легкість використання (простота використання та інтеграції є ключовими аспектами для розробників. Менше складнощів у використанні дозволяють швидше розгортати та підтримувати веб-додатки);
- підтримка різних форматів 3D моделей (різноманітність форматів об'єктів, які можна створювати та відображати, важлива для використання в різних сценаріях. Визначення, як добре кожен метод підтримує різноманітні формати об'єктів, є ключовим для універсальності);
- споживання ресурсів (забезпечення ефективного використання ресурсів, таких як пам'ять та процесор, є важливим аспектом для оптимізації продуктивності та забезпечення плавного виконання);
- обсяг коду та інтеграція (мінімізація обсягу коду та легка інтеграція з іншими технологіями роблять розробку та підтримку більш простою та зручною);
- можливості для розширення (важливо враховувати можливості для розширення функціоналу в майбутньому без необхідності повного переписування коду);
- спільнота та підтримка (наявність активної спільноти та підтримки з боку розробників важлива для вирішення проблем та отримання нових можливостей).

Проведемо аналіз шкал:

- продуктивність (абсолютна шкала, що показує швидкість виконання коду у секундах). Чим менше час виконання, тим краще. Використання мілісекунд дозволяє точно виміряти продуктивність;
- швидкість відгуку (абсолютна шкала, що показує швидкість відтворення 3D моделі у мілісекундах). Чим менше час відгуку, тим швидше система реагує на дії користувача;

– легкість використання (порядкова шкала від 1 до 10, де 1 – складно використовувати, а 10 – легко). Використання числової шкали відображає суб'єктивну оцінку експерта з точки зору легкості використання;
– підтримка різних форматів 3D моделей (абсолютна шкала, що вказує на кількість підтримуємих форматів 3D об’єктів). Чим більше підтримуваних форматів об'єктів, тим більше можливостей для реалізації різноманітних сценаріїв;
– споживання ресурсів (абсолютна шкала, що вказує на кількість ресурсів у МБ, які потрібні для відображення об'єкта). Чим менше пам'яті використовує система, тим менше вплив на ресурсоємність;
– обсяг коду та інтеграція (абсолютна шкала, що вказує на кількість рядків коду для відображення об'єкту). Чим менше рядків коду та легше інтеграція, тим простіше підтримувати та розвивати систему;
– можливості для розширення (шкала найменувань, Так або Ні). Більше можливостей для розширення дозволяє системі адаптуватися до зростаючих потреб;
– спільнота та підтримка (абсолютна шкала, що відображає кількість людей, які підтримують та використовують дану технологію). Більша кількість активних користувачів свідчить про активну спільноту та підтримку.

Для вимірювання швидкодії сторінки буде заміряно та порівняно показники Largest Contentful Paint (див. рис. 3.1).

Рисунок 3.1 – Оцінка Largest Contentful Paint

Largest Contentful Paint (LCP) – це стабільний показник Core Web Vital для вимірювання передбачуваної швидкості завантаження [16]. Він позначає момент на часовій шкалі завантаження сторінки, коли ймовірно завантажено основний вміст сторінки. Швидкий LCP допомагає переконати користувача, що сторінка корисна.

Щоб забезпечити хорошу взаємодію з користувачем, сайти повинні намагатися мати LCP 2,5 секунди або менше.

Також для вимірювання швидкодії сторінки буде заміряно та порівняно показники First Input Delay (див. рис. 3.2).

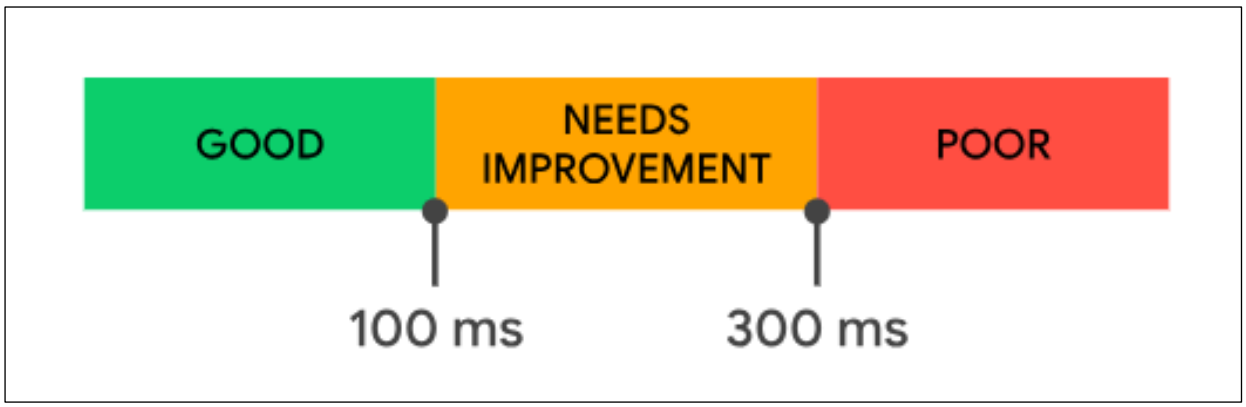

Рисунок 3.2 – Оцінка First Input Delay

Щоб забезпечити хорошу взаємодію з користувачем, сайти повинні прагнути мати FID (First Input Delay) 100 мілісекунд або менше. Якщо значення більше за 300 мілісекунд – це означає, що веб-додаток потребує негайного покращення, для того, щоб не примушувати користоувача чекати занадто довго, і він не покинув сторінку.

FID виникає через те, що головний потік браузера зайнятий чимось іншим, тому він не може відповісти користувачеві. Однією з поширених причин цього може статися те, що браузер зайнятий аналізом і виконанням великого файлу JavaScript (в нашому випадку – 3D модель), завантаженого вашою програмою. Поки він це робить, він не може запустити жодних прослуховувачів подій, оскільки JavaScript, який він завантажує, може наказати йому зробити щось інше [17].

Для того, щоб отримати дані LCP та FID, в програмному коді для кожного методу побудови 3D моделей буде запущено методи getLCP() та getFID(), які будуть імпортовані з модуля web-vitals.

Приклад отриманих даних для LCP наведено на рисунку 3.3.

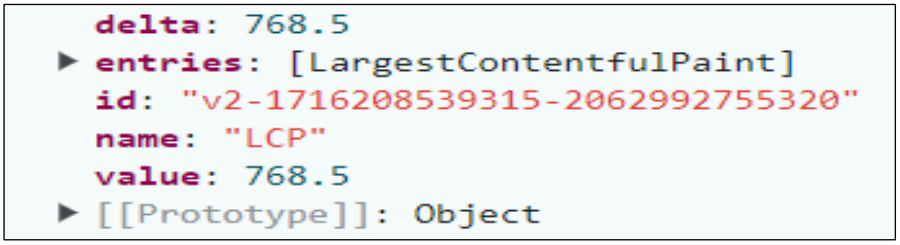

Рисунок 3.3 – Приклад отриманих даних для LCP

Приклад отриманих даних для FID наведено на рисунку 3.4.

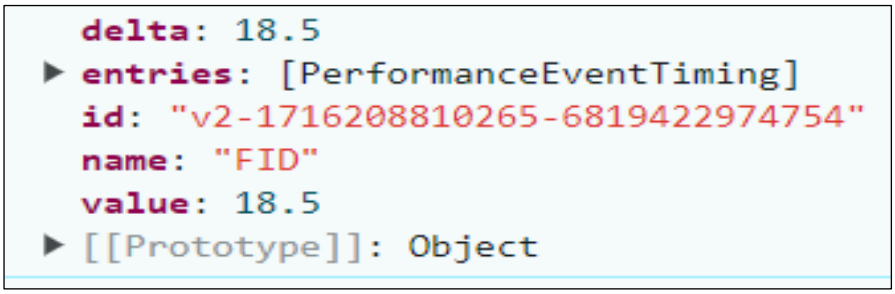

Рисунок 3.4 – Приклад отриманих даних для FID

Для експерименту нам буде потрібно значення value – це час у мілісекундах.

Після проведення ряду експериментів для кожного методу буде складено порівняльну таблицю та визначено, який метод, або їх сукупність, найкраще підходять для побудови 3D моделей у веб-додатках.

Найбільш чіткими та легко вимірюваними показниками ефективності роботи веб-додатків, зокрема тих, що містять 3D графіку, є показники швидкодії. Ці показники включають завантаженість процесору, зайнятий обсяг оперативної пам'яті, час, необхідний на обробку операцій, та інші. Виміряти їх можна за допомогою спеціального програмного забезпечення, такого як Chrome Devtools, а зокрема інструменту Lighthouse [18].

Chrome Devtools забезпечує можливість аналізувати різні аспекти роботи веб-додатків та отримувати детальну інформацію про їх продуктивність та ефективність. Інструмент Lighthouse, який доступний у складі Devtools, надає змогу проводити аудит сторінки, включаючи аналіз швидкодії завантаження, продуктивності JavaScript, оптимізації для мобільних пристроїв, доступності та інших аспектів [19].

Аналіз цих показників дозволяє виявити можливі проблеми та недоліки у роботі веб-додатка, включаючи ті, які пов'язані з використанням 3D графіки. Наприклад, високий обсяг оперативної пам'яті або завантаженість процесору можуть свідчити про неефективне використання ресурсів під час рендерингу 3D сцен або обробки великої кількості даних. Час, необхідний на обробку операцій, може бути показником продуктивності конкретних операцій, таких як завантаження та рендеринг тривимірних об'єктів.

3.2 Підготовка до проведення експерименту

Щоб дослідити можливості кожного методу, буде розроблено веб-додаток на базі React з використанням кожної з представлених бібліотек та фреймворків [20].

До розробленого веб-застосунку було визначено наступні функціональні вимоги:

- програма повинна відображати 3D об’єкт;
- 3D об’єкт можна обертати для того, щоб мати змогу подивитись з різних сторін;
- 3D об’єкт можна приблизити або віддалити для покращення користувацького досвіду.

3D моделі для експериментальної частини повинні бути однаковими (в нашому випадку – це модель автомобіля), а реалізація для кожного методу та фреймворку буде відрізнятися.

4 ПРОВЕДЕННЯ ДОСЛІДЖЕННЯ

Для проведення дослідження було розроблено веб-сторінку, на якій відображається 3D модель та базова інформація про неї (див. рис. 4.1).

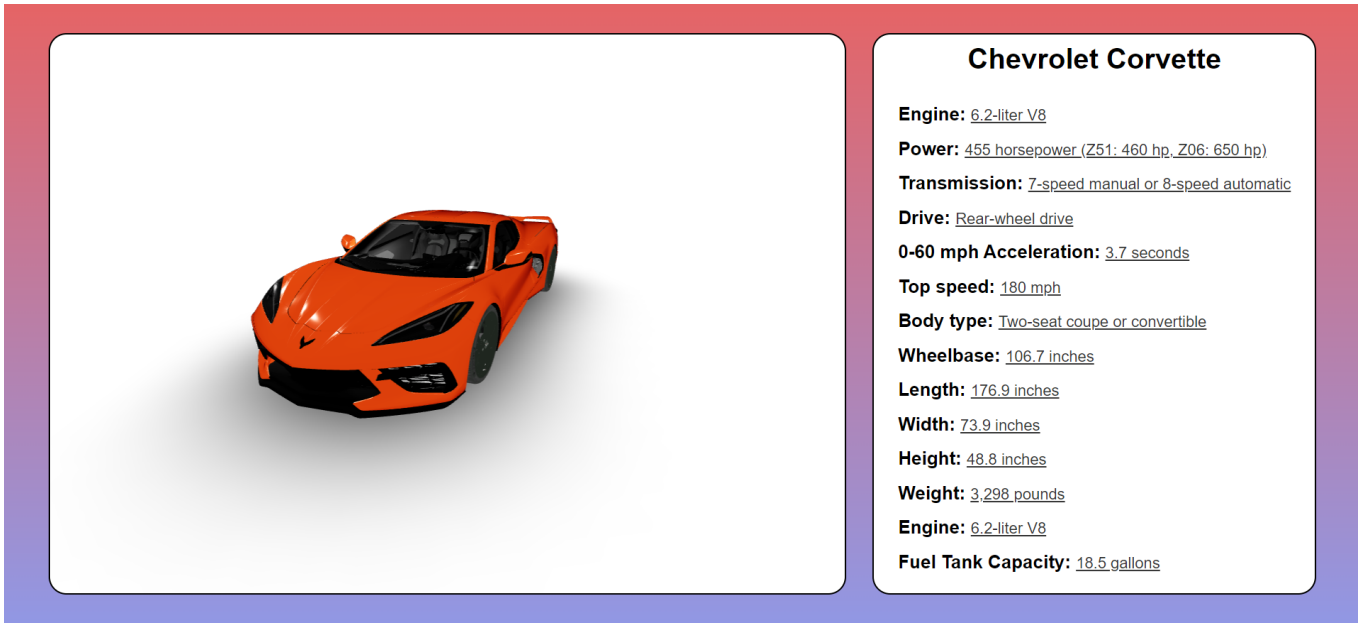

Рисунок 4.1 – Веб-сторінка для проведення дослідження

Під час проведення дослідження використовувалась однакова 3D модель формату .glTF, але вона відображалась за допомогою різних матодів та бібліотек, таких як:

– Three.js;
– Babylon.js;
– Whitestorm.js;
– A-Frame.

Всі ці бібліотеки підтримують відображення 3D моделей формату .glTF. Це бажаний формат для використання у веб-розробці. Завдяки своїй ефективності та якості, цей формат забезпечує швидке та ефективне завантаження та відображення трьохмірних моделей у веб-додатках без необхідності використання додаткових плагінів або розширень [21].

Бібліотеки Three.js, Babylon.js, Whitestorm.js та A-Frame мають вбудовану підтримку .glTF, що дозволяє розробникам легко і безпроблемно інтегрувати 3D моделі у свої веб-додатки, забезпечуючи високу якість відображення та оптимальну продуктивність. Такий підхід дозволяє створювати захоплюючі та вражаючі візуальні ефекти у веб-додатках, підвищуючи користувацький досвід та залучення аудиторії.

4.1 Three.js

Three.js підтримує наступні формати 3D моделей: .glTF, .obj, .stl, .dae, .fbx, .ply.

Завантаживши 3D модель, її необхідно перетворити у формат .jsx для відображення у React застосунку. Для цього запускаємо команду npx gltfjsx corvette.gltf, де corvette.gltf – це наша 3D модель. Тепер в нас є React-компонент, який можна використовувати у веб-застосунку:

```
import React from 'react';
import { Canvas } from 'react-three-fiber';
import { OrbitControls, ContactShadows } from '@react-three/drei';

<Canvas className='canvas'>
  <pointLight intensity={10} position={[0, 5, 1]} />
  <pointLight intensity={10} position={[0, 0.7, -0.5]} />
  <pointLight intensity={10} position={[0, 2, -2.5]} />
  <ambientLight intensity={10} color="#ffffff" />
  <OrbitControls />
  <Corvette />
  <ContactShadows opacity={0.9} scale={60} />
</Canvas>
```

Компонент <Canvas /> створює контейнер для відображення 3D-сцени.

Компонент <pointLight /> – це джерело точкового світла, яке додає освітлення в сцену. Властивість intensity задає інтенсивність освітлення, а position – позицію в тривимірному просторі.

Компонент <ambientLight /> – це навколишнє освітлення, яке додає загальне освітлення сцени.

Компонент <OrbitControls /> – це компонент, який дозволяє користувачу обертати та наближати/віддаляти сцену.

Компонент <ContactShadows /> – це компонент, який додає контактні тіні на сцену. Властивість opacity задає прозорість тіні, а scale – встановлює розмір тіні об’єкту.

4.2 Babylon.js

Babylon.js підтримує наступні формати 3D моделей: .glTF, .obj, .stl, .dae, .fbx, .ply, .3ds.

Для створення 3D-сцени за допомогою Babylon.js, необхідно завантажити тривимірну модель формату .glTF та розмістити її у файлах проекту. Тепер в нас є 3D модель, яку можна використовувати у веб-застосунку:

```
import React from 'react';
import { Engine, Scene, ArcRotateCamera, Vector3, HemisphericLight } from
'@babylonjs/core';
import { Model } from '@babylonjs/loaders';

<Engine antialias adaptToDeviceRatio canvasId="babylonJS">
  <Scene>
    <ArcRotateCamera
      name="camera"
      target={Vector3.Zero()}
      alpha={-Math.PI / 4}
      beta={(Math.PI / 4)}
      radius={5}
    />
    <HemisphericLight
name="light"
intensity={0.7}
direction={Vector3.Up()}
         />
    <Model sceneFilename="./models/corvette.gltf" />
  </Scene>
</Engine>
```

Компонент <Engine /> створює двигун WebGL для рендерингу 3D-сцени. Властивість antialias вказує на наявність згладжування для відображення графіки, а

adaptToDeviceRatio дозволяє адаптувати різні пристрої до співвідношення сторін канвасу.

Комопнент <Scene /> створює сцену, яка буде містити всі елементи 3D-сцени, такі як камери, освітлення та моделі.

Комопнент <ArcRotateCamera /> створює камеру, яка дозволяє користувачеві обертати та переміщувати камеру навколо сцени. Властивість target вказує на точку, навколо якої камера буде обертатися, а alpha, beta та radius визначають положення та відстань камери.

Компонент <HemisphericLight /> створює полуденне освітлення для сцени. Властивість intensity вказує на яскравість світла, а direction визначає напрямок освітлення.

Компонент <Model /> відображає 3D-модель, яка завантажується з файлу corvette.gltf.

4.3 Whitestorm.js

Whitestorm.js підтримує наступні формати 3D моделей: .glTF, .obj, .stl, .dae.

Для інтеграції тривимірної моделі, використовуючи Whitestorm.js, необхідно створити компонент, налаштувати в ньому освітлення, камеру, обертання, наближення/віддалення та імпорт самої моделі:

```
import React, { useEffect, useRef } from 'react';
import { World, Model, Light, Extend } from 'whitestormjs';
Extend('cameraOrbit', {
  zoom = (delta) => this.controls.zoom(delta),
  rotateY = (delta) => this.controls.yaw(delta),
  rotateX = (delta) => this.controls.pitch(delta),
});
export const Corvette = () => {
  const whsCanvas = useRef(null);
  useEffect(() => {
    const world = new World({
      background: { color: 0xffffff },
      camera: {
        position: [0, 10, 50],
        near: 0.1,
```

```
        far: 1000,
        orbitControls: true
      }
    });
    world.setControls('cameraOrbit');
    world.add(new Light({
      color: 0xffffff,
      intensity: 1,
      position: [10, 20, 30],
    }));
    world.add(new Model({
      geometry: { path: './models/corvette.gltf' },
      position: [0, 0, 0],
    }));

    return () => world.dispose();
  }, []);

  return <div ref={whsCanvas}></div>;
};
```

Функція Extend використовується для розширення можливостей компоненту, а саме наближення/віддалення та обертання камери навколо моделі.

new World() створює новий об'єкт сцени, який представляє собою віртуальний простір, де ви можете відображати ваші 3D-моделі.

world.setControls('cameraOrbit') використовується для встановлення контролерів камери у Whitestorm.js.

world.add(new Light()) додає нове світло до сцени.

world.add(new Model()) додає нову 3D модель до сцени, імпортуючи її з папки models.

4.4 A-Frame

A-Frame підтримує наступні формати 3D моделей: .glTF, .obj, .stl, .dae, .ply, .fbx.

Для інтеграції тривимірної моделі на веб-сторінку з використанням бібліотеки A-Frame було використано наступний код:

```
import React from 'react';
```

```
import 'aframe';
import { Entity, Scene } from 'react-aframe';

export const Corvette = () => {
  return (
    <Scene>
      <Entity
        gltf-model="./models/corvette.gltf"
        position="0 0 -4"
        scale="0.5 0.5 0.5"
        rotation="0 45 0"
      />
      <Entity
             light={{
               type: 'point',
               intensity: 0.7,
               color: '#ffffff',
               distance: 20
             }}
             position="2 4 4"
           />
      <Entity camera look-controls />
    </Scene>
  );
};
```

Компонент <Entity camera look-controls /> додає камеру з можливістю керування та наближення/віддалення.

Для додавання моделі використовується компонент <Entity /> з параметром gltf-model, в якому вказується шлях до 3D моделі.

Для додавання світла використовується компонент <Entity /> з параметром light, в якому вказується тип освітлення, інтенсивність, колір та відстань.

4.5 Порівняння методів

Після реалізації кожного методу побудови 3D моделей було створено таблицю 4.1, яку було заповнено зібраними метриками за обраними критеріями.

Таблиця 4.1 – Зібрані метрики для кожного методу побудови 3D моделей

	Three.js	Babylon.js	Whitestorm.js	A-Frame
Продуктивність (LCP)	1.5 с	1.6 с	1.5 с	1.7 с
Швидкість відгуку (FID)	60 мс	70 мс	70 мс	80 мс
Легкість використання	8	9	7	9
Підтримка різних форматів	6	7	4	6
Споживання ресурсів	76 мб	78 мб	80 мб	79 мб
Обсяг коду	68	17	33	25
Можливості для розширення	Так	Так	Так	Так
Спільнота та підтримка	2 540 000	600 000	550 000	875 000

Нормалізуємо абсолютні шкали за формулою 4.1.

$$x' = \frac{x - (x)}{(x) - (x)} \tag{4.1}$$

Порядкові шкали нормалізуємо за формулою 4.2.

$$x' = \frac{x - (x)}{(x) - (x)} \tag{4.2}$$

де *x* – значення, яке треба нормалізувати;

max(x) – максимальне значення в наборі даних;

min(x) – мінімальне значення в наборі даних;

x′ – нормалізоване значення.

Нормалізуємо значення для кожного критерія та заповнюємо значення до таблиці 4.2.

Таблиця 4.2 – Нормалізовані шкали

	Three.js	Babylon.js	Whitestorm.js	A-Frame
Продуктивність (LCP)	1	0.5	1	0
Швидкість відгуку (FID)	1	0.5	0.5	0
Легкість використання	0.5	1	0	1
Підтримка різних форматів	0.3	1	0	0.3
Споживання ресурсів	1	0.5	0	0.25
Обсяг коду	0	1	0.686	0.843
Можливості для розширення	1	1	1	1
Спільнота та підтримка	1	0.0208	0	0.0417

На таблиці 4.2 видно, що:

- Three.js та Whitestorm.js мають високу оцінку в продуктивності, що свідчить про швидку відповідь на запити та зниження часу, необхідного для завантаження важливого контенту для користувачів;
- Three.js також демонструє високу ефективність в швидкості відгуку, забезпечуючи користувачам швидкий та безперервний досвід взаємодії з веб-додатком;
- Babylon.js та A-Frame мають найвищу оцінку з легкості використання, що сприяє швидкому розвитку та реалізації проєктів;
- Babylon.js перевершує Three.js у підтримці багатьох різних форматів моделей, зокрема .glTF, який є основним стандартом у веб-розробці;

- Three.js забезпечує ефективне використання ресурсів, знижуючи навантаження на сервер та підтримуючи оптимальну продуктивність веб-додатків;
- Babylon.js може виявитися меншим за обсягом коду порівняно з іншими бібліотеками, що полегшує розробку та підтримку великих проєктів;
- кожна з розглянутих бібліотек та фреймворків має широкі можливості для розширення та налаштування, що дозволяє розробникам створювати різноманітні 3D-сцени та анімації за допомогою додаткових плагінів та розширень;
- Three.js має найбільш активну спільноту користувачів та підтримку з боку розробників, що допомагає швидко знаходити рішення на форумах та відстежувати оновлення та виправлення помилок.

Після додавання кожного критерія для кожного методу Просумуємо критерії для кожної бібліотеки та отримаємо таблицю 4.3.

Таблиця 4.3 – Сума критеріїв для кожної бібліотеки

Бібліотека	Сума
Three.js	**5.8**
Babylon.js	5.5208
Whitestorm.js	3.186
A-Frame	3.4347

Розробивши просту веб-сторінку, провівши дослідження, отримавши всі результати та зробивши аналіз, можна впевнено заявити, що Three.js – найкраща бібліотека для побудови 3D моделей у веб-середовищі за допомогою React.

ВИСНОВКИ

В результаті дослідження методів побудови 3D моделей у веб-додатках за допомогою React виявлено, що для того щоб вони визначали сучасні стандарти для створення вражаючих та інноваційних користувацьких інтерфейсів, необхідно провести дослідження та виявити найкращий метод серед всіх інших, який дозволить покращити користувацький досвід при використанні 3D об’єктів у веб-додатку.

Розглядаючи різноманітні підходи та техніки у використанні 3D графіки у веб-додатках за допомогою React, було досліджено інструменти, серед яких виділяються Three.js, Babylon.js, Whitestorm.js та A-Frame. Кожен з цих інструментів має свої переваги та недоліки, а також особливості в інтеграції з React. Дослідження цих технологій спрямоване на визначення оптимального вибору для реалізації 3D моделей у веб-додатках, зокрема на платформі React, з метою досягнення високої якості візуалізації та оптимальної взаємодії з користувачем.

Важливим аспектом дослідження було виявлення та аналіз проблем, пов'язаних з використанням 3D моделей у веб-додатках. Однією з таких проблем є великий обсяг даних, який може виникнути при роботі з тривимірними моделями, особливо якщо вони містять велику кількість вершин, текстур та інших даних. Це може призвести до збільшення часу завантаження та обробки даних, що в свою чергу може негативно вплинути на продуктивність додатка та користувацький досвід.

Крім того, ще однією важливою проблемою є можливість впливу на продуктивність веб-додатка через використання 3D графіки. Завантаження та рендеринг великої кількості тривимірних об'єктів може призвести до зниження швидкості роботи додатка та спричинити перебої в його роботі, особливо на пристроях з обмеженими ресурсами.

Дослідження було проведено з метою визначення оптимального методу побудови тривимірних моделей у веб-середовищі з використанням фреймворка React. Кожен з методів був застосований для створення веб-додатку, який відображав тривимірну модель разом з базовою інформацією про неї. Під час експерименту здійснювалася оцінка продуктивності кожної бібліотеки, враховуючи швидкість та ресурсоємність процесу рендерингу, а також підтримку різноманітних форматів 3D моделей.

Застосована методика лінійної адаптивної згортки з ваговими коефіцієнтами дозволила систематизувати та порівняти результати експерименту для кожного методу побудови 3D моделей. На основі отриманих даних було визначено, що бібліотека Three.js виявилася найбільш ефективною у контексті використання разом з фреймворком React. Її висока популярність серед програмістів, швидкість рендерингу моделей, підтримка різноманітних форматів 3D моделей та легкість використання зробили її найкращим вибором для побудови тривимірних моделей у веб-додатках з використанням React.

Дослідження спрямоване на розробку стратегій та рекомендацій для подолання цих викликів та оптимізації використання 3D графіки у React-заснованих веб-додатках.

Отримані результати дослідження будуть важливими для веб-розробників та компаній, що прагнуть не лише вдосконалити візуальний ефект своїх веб-додатків, але й надати їм конкурентну перевагу на ринку електронних продуктів. Аналіз ефективних методів інтеграції 3D моделей у веб-додатки за допомогою React дозволить розробникам обирати оптимальні стратегії та інструменти для досягнення високої якості візуального враження, що сприятиме покращенню сприйняття користувачами та підвищує конкурентоспроможність їхніх продуктів в інтернет-просторі.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Kleppmann M. Designing Data-Intensive Applications. – O'Reilly Media, 2017. – 616 с.

2. Duckett J. Web Design with HTML, CSS, JavaScript and jQuery Set. – Wiley, 2014. – 1152 с.

3. Colins M. Pro HTML5 with CSS, JavaScript, and Multimedia: Complete Website Development and Best Practices. – Apress, 2017. – 592 с.

4. Angel, E., Shreiner, D. Interactive Computer Graphics: A Top-Down Approach with WebGL. – Pearson, 2014. – 752 с.

5. Getting Started with React 3D. URL: https://reactjs.org/docs/three.html (дата звернення 04.04.2024).

6. Building 3D Experiences with React and Three.js. URL: https://www.awwwards.com/building-3d-experiences-with-react-and-three-js.html (дата звернення 06.04.2024).

7. Sufyan U. Conquering JavaScript: Three.js. – Taylor & Francis, 2023. – 194 с.

8. Exploring 3D Graphics in React using Babylon.js. URL: https://doc.babylonjs.com/start/chapter1 (дата звернення 08.04.2024).

9. Moreau-Mathis J. Babylon.js Essentials. – Packt Publiching, 2016. – 202 с.

10. Whitestorm.js Documentation. URL: https://whs.io/docs (дата звернення 10.04.2024).

11. Building Virtual Reality Experiences with A-Frame and React. URL: https://aframe.io/docs/1.2.0/introduction (дата звернення 12.04.2024).

12. Straccia A. Interactive Web Development with A-Frame. – Orange Education Pvt, 2024. – 290 с.

13. React 3D Library Comparison. URL: https://dev.to/lesnitsky/react-3d-library-comparison-4a4p (дата звернення 16.04.2024).

14. Леснік С., Хижняк Т. Застосування методу лінійної згортки для вибору джерела альтернативної енергії//Електроніка і зв’язок. 2013. С. 24.

15. Yerokhin A., Semenets V., Nechyporenko A., Turuta O., Babii A. F-transform 3D point cloud filtering algorithm//IEEE Second International Conference on Data Stream Mining & Processing (DSMP). 2018. С. 524-527.

16. Largest Contentful Paint (LCP) | Articles. URL: https://web.dev/articles/lcp (дата звернення: 22.04.2024).

17. First Input Delay (FID) | Articles. URL: https://web.dev/articles/fid (дата звернення: 24.04.2024).

18. What is Google Lighthouse and how to use it? URL: https://www.elegantthemes.com/blog/wordpress/what-is-google-lighthouse-and-how-to-use-it (дата звернення: 26.04.2024).

19. Goggle Lighthouse: What it is & how to use it. URL: https://www.semrush.com/blog/google-lighthouse (дата звернення: 26.04.2024).

20. Integrating 3D Models into React Apps. URL: https://medium.com/@xavier44/integrating-3d-models-into-react-apps-28e5d1f6c58b (дата звернення 28.04.2024).

21. Dirksen J. Learn Three.js – fourth edition. – Packt Publishing, 2023. – 554 с.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ ЗА НАУКОВИМИ НАПРЯМАМИ КЕРІВНИКА ТА НАУКОВЦІВ КАФЕДРИ ПРОГРАМНОЇ ІНЖЕНЕРІЇ

15. Yerokhin A., Semenets V., Nechyporenko A., Turuta O., Babii A. F-transform 3D point cloud filtering algorithm//IEEE Second International Conference on Data Stream Mining & Processing (DSMP). 2018. С. 524-527.