МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ ЕЛЕКТРОНІКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК
СЕКЦІЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ПРОЕКТУВАННЯ

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему: «Веб-ресурс для перегляду 3d моделей з використанням технології WebGL»

за напрямом підготовки 6.050101 «Комп'ютерні науки»

Виконавець роботи: студент групи ІТ-52 Ясінська Тетяна Андріївна

Кваліфікаційна робота бакалавра
захищена на засіданні ЕК
з оцінкою _____ «__» _____2019 р.

Науковий керівник _____ к.т.н., доц., Федотова Н.А.
(підпис) (науковий ступінь, вчене звання, прізвище та ініціали)

Голова комісії _____ Шифрін Д. М.
(підпис) (науковий ступінь, вчене звання, прізвище та ініціали)

Засвідчую, що у цій дипломній роботі немає
запозичень з праць інших авторів
без відповідних посилань.

Студент _____
(підпис)

Суми-2019

Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра комп'ютерних наук
Секція інформаційних технологій проектування
Напрям підготовки – 6.050101 «Комп'ютерні науки»

ЗАТВЕРДЖУЮ
Зав. секцією ІТП

_____ В. В. Шендрик
«__» _____ 2019 р.

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ БАКАЛАВРА СТУДЕНТУ

Ясінська Тетяна Анддріївна

1 Тема роботи *Веб-ресурс для перегляду 3d моделей з використанням технології WebGL*

керівник роботи *Федотова Наталья Анатольєвна, к.т.н., доцент*,

затверджені наказом по університету від « 17 » травня 2019 р. № 0834-III

2 Строк подання студентом роботи «20» травня 2019 р.

3 Вхідні дані до роботи технічне завдання на розробку веб-ресурсу, 3d модель у форматі json

4 Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) аналіз предметної області, постановка задачі та методи дослідження, проектування веб-ресурсу для перегляду 3d моделей, розробка веб-ресурсу для перегляду 3d моделей.

5 Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) актуальність проблеми, аналіз аналогів, мета дипломної роботи, задачі проекту, аналіз технологій, етапи розробки веб-ресурсу «ITP Kaleidoscope», висновки.

6. Консультанти розділів роботи:

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7.Дата видачі завдання_____

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Ознайомлення з предметною областю	11.09.18 – 11.09.18	
2	Визначення в потребі веб-ресурсу	12.09.18 – 18.09.18	
3	Ідентифікація ідеї проекту	19.09.18 – 20.09.18	
4	Аналіз документації WebGL	21.09.18 – 27.09.18	
5	Аналіз документації JavaScript	28.09.18 – 04.10.18	
6	Визначення інструментарію	05.10.18 – 08.10.18	
7	Планування WBS	09.10.18 – 10.10.18	
8	Планування OBS	11.10.18 – 12.10.18	
9	Складання календарного плану	15.10.18 – 15.10.18	
10	Визначення ризиків	16.10.18 – 19.10.18	
11	Розробка структури веб-ресурсу	22.10.18 – 29.10.18	
12	Реалізація структури у вигляді веб-сторінки	30.10.18 – 05.11.18	
13	Розробка функціоналу завантаження моделі	06.11.18 – 19.11.18	
14	Розробка функціоналу базових операцій	20.11.18 – 30.11.18	
15	Тестування розробником	03.12.18 – 04.12.18	
16	Тестування незалежною особою	05.12.18 – 06.12.18	
17	Оформлення документації	07.12.18 – 25.04.19	
18	Введення в експлуатацію	26.04.19 – 26.04.19	
19	Архівація проекту	29.04.19 – 29.04.19	

Студент _____ Ясінська Т.А.

(підпис)

Керівник роботи _____ к.т.н., доц. Федотова Н.А.

(підпис)

РЕФЕРАТ

Тема дипломної роботи «Веб-ресурс для перегляду 3d моделей з використанням технології WebGL».

Дипломна робота складається зі вступу, чотирьох розділів, висновка, списку використаної літератури та додатків.

Пояснювальна записка містить 112с., 67 рис., 11 табл., 3 додатків, 22 джерел.

У першому розділі досліджується актуальність проблеми, проводиться аналіз існуючих аналогів.

Другий розділ присвячений формування мети дипломної роботи та задач проекту, вибору засобів реалізації та плануванню робіт.

У третьому розділі виконується проектування веб-ресурсу, де наведені діаграми у нотації IDF0 та Use Case.

Останній розділ присвячений детальному опису практичної реалізації проекту: виконання прототипування веб-сторінки, розмітка та форматування веб-ресурсу, налаштування та перевірка працездатності веб-браузера з WebGL, опис реакції на дії користувача, розробка функцій маніпуляцій над моделлю.

Результатом проведеної роботи є розроблений веб-ресурс, який дозволяє користувачу обирати одну із чотирьох моделей для візуалізації на веб-сторінці та виконувати базові маніпуляції з нею.

Ключові слова: WEBGL, ВЕБ-РЕСУРС, ТЕХНОЛОГІЇ, ВІЗУАЛІЗАЦІЯ, 3D МОДЕЛІ.

ЗМІСТ

Вступ 6

1. Аналіз предметної області 8

1.1 Актуальність проблеми 8

1.2 Аналіз існуючих аналогів 11

2. Постановка задачі та методи дослідження 18

2.1 Мета та задачі 18

2.2 Вибір засобів реалізації 20

2.3 Планування робіт 24

3. Проектування веб-ресурсу для перегляду 3d моделей 25

3.1 Діаграми нотації IDEF0 25

3.2 Use Case Diagram 31

4. Розробка веб-ресурсу для перегляду 3d моделей 33

4.1 Розробка інтерфейсу користувача 33

4.2 Розробка логіки роботи веб-ресурсу 42

Висновки 64

Список літератури 66

Додаток А 68

Додаток Б 71

Додаток С 88

ВСТУП

В сучасному світі неможливо уявити комфортну діяльність користувача без використання веб-технологій: веб-додатків, веб-ресурсів, веб-сайтів тощо. Створення даних технологій являється актуальним питанням, адже такі ресурси дають можливість сформувати інформацію у доступному форматі для користувача в глобальному масштабі та являються апаратно-незалежними та крос-платформними.

Також поряд з розвитком веб-технологій популярність набирає 3d моделювання. Розробники веб-браузерів майже щодня оновлюють версії своїх продуктів, тому сучасні веб-браузери отримують можливість використання тривимірної графіки в Інтернеті. З використанням даних технологій користувач може власноруч керувати тривимірними об'єктами(обертати, масштабувати, переміщувати модель та вивчати її структуру) [11].

Даний вид технологій застосовується на веб-сторінках Інтернет-магазинів, різних фірм, в системі дистанційного навчання. Причиною даного явища є те, що за рахунок додавання тривимірних елементів в структуру веб-сторінки розробники сайтів отримують особливий ефект у дизайні. В наслідок чого веб-сторінка стає більш цікавою для користувача та конкурентоспроможною у порівнянні з іншими веб-сайтами на ринку попиту. Тому розробка веб-технологій для відображення та використання тривимірних моделей в Інтернеті є актуальною задачею.

На сьогоднішній день технологій для завантаження тривимірних елементів на веб-сторінку існує не велика кількість, найбільш популярні: Adobe Flash, Silverlight та WebGL. Більшість з них знаходяться в статусі розробки, адже, наприклад, технологія WebGL з’явилась на ринку сайтобудування наприкінці 2015 року і на даний час можливості даної технології до кінця не вивчені.

В Інтернет просторі для користувачів, які прагнуть завантажити модель для перегляду та провести певні операції з нею, існує певний перелік сайтів, але більшість з них мають такі недоліки:

- без авторизації або сплати за профіль користувач не має доступ до ресурсу;
- відсутність базового набору операцій для роботи з моделями, лише перегляд;

На основі аналізу ринку аналогів було сформовано мету роботи - створити веб-ресурс, який дозволить користувачу якісно та ефективно працювати з 3d моделями. Адже всі ті недоліки, що були описані раніше, мають бути виправлені. Також розробником будуть додані додаткові функції/операції для створення адаптивного веб-сервера для маніпулювання з 3d моделями.

Для досягнення мети необхідно вирішити певний перелік задач:

- проаналізувати предметну область;
- обрати технологію для відображення 3d моделі на веб-ресурсі;
- розробити структуру веб-ресурсу;
- реалізувати дану структуру у вигляді веб-сторінки;
- розробити функціонал завантаження моделі та базових маніпуляцій з нею;
- провести тестування веб-ресурсу.

1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Актуальність проблеми

Веб-технології міцно увійшли в повсякденне життя людства. Кожного дня звичайні користувачі проводять у всесвітній мережі досить великий проміжок свого часу – переглядають свіжі новини, здійснюють покупки в Інтернет-магазинах, спілкуються через чати або месенджери з колегами та рідними, використовують онлайн-додатки під час трудової діяльності, наприклад Google Docs.[17]

На сьогоднішній день під час стрімкого прогресу привабливо оформлений текст та графічні зображення на веб-сторінці вже нікого не здивують. У зв’язку з чим, вимоги до оформлення сайтів змінились. Тепер для успішної конкуренції на ринку попиту розробники веб-сторінок повинні додавати в дизайн якомога більше елементів, які зможуть зацікавити користувача під час перегляду сайту, наприклад : анімоване меню, що має певні ефекти при наведені курсора; різні варіанти підсвічування для елементів сайту(кнопки, посилання) та інші[12].

Індустрія послуг та розваг в мережі Інтернет бурхливо розвивається, головні розробники програмного забезпечення покращують генерацію тривимірної графіки в своїх продуктах. Традиційно підтримка відображення 3d моделей була обмежена високою продуктивністю персональних комп’ютерів або спеціалізованими ігровими консолями, а програмування вимагало застосування складніших алгоритмів для реалізації даного процесу. Проте завдяки стрімкого розвитку потужності персональних комп’ютерів та оновлення версій веб-браузерів, їх розширень та можливостей, створення та відображення тривимірної графіки з використанням веб-технологій стали можливими[17].

Використання тривимірної графіки у такому процесі як сайтобування набуло швидкої популярності. Для тримання оригінального ефекту розробники сайту додають тривимірні моделі замість звичних графічних елементів, інколи навіть анімовані[1]. На даний час існує перелік випадків, в яких релевантна генерація 3d моделей на веб-сторінках:

- якщо сайт націлений на продаж об’єктів нерухомості;
- якщо компанія займається ландшафтним дизайном, створює унікальні екстер’єри для житлових приміщень;
- якщо сайт створений для яскравого та оригінального представлення об’єктів в сфері туризму(3d екскурсії з супровідною анімацією);
- якщо компанія виконує продаж авто, катерів, квадроциклів, мототехніки, літаків та інші;
- якщо компанія займається створенням оригінального оформлення квартир, виготовленням меблі;
- якщо підприємство виготовляє деталі, обладнання, які необхідно дистанційно представити замовнику в усіх деталях;
- якщо Ви – 3d дизайнер та прагнете створити унікальну веб-сторінку, у якості вашого портфоліо;
- якщо на сайті діє Інтернет-магазин та 3d моделі товарів допоможуть підвищити ефективність продаж[2];

3d моделювання не стоїть на місці, кожного дня додають нові плагіни для комфортної роботи зі складними моделями, роблячи їх максимально приближеної до вигляду у реальному світі. Але інколи виникають форс-мажорні ситуації, коли фахівець у даній області не має доступу до програмного забезпечення, тому виникає необхідність у використанні допоміжних засобів. 3d веб-ресурс(редактор) має досить великий перелік переваг у використанні у порівнянні з десктопним додатком:

- маніпуляції з 3d-моделями без допоміжних додатків;
- одночасний доступ до роботи з моделями з різних приладів через Інтернет;
- легка інтеграція з іншими веб-серверами та соціальними мережами[16].

На даний час існує велика кількість веб-ресурсів для завантаження 3d-моделей, але більшість з них мають певний перелік недолік:

- не можливість використання веб-ресурсу без авторизації користувача;
- лімітована кількість маніпуляцій с моделями, якщо користувач використовує безкоштовний профіль;
- відсутність певних операції у роботі з моделями:
- налаштування світла з будь-якої проекції;
- зміна кольору/текстури моделі;
- відсутність вбудованого конвертора для відкриття моделі будь-якого формату.

Тому розроблений веб-ресурс дасть можливість генерації 3d моделі на сторінці без авторизації та сплати за використання. Також у структурі веб-ресурсу буде додано панель керування, за допомогою якої користувач зможе виконати базові маніпуляції з 3d моделями.

Даний проект буде нести соціально-економічну цінність, оскільки він буде мати попит перш за все у вищих навчальних закладах, які у своєму складі мають кафедри/спеціальності з нахилом на 3d моделювання, інженерію. Адже використовуючи даний продукт, викладачі зможуть зробити навчально-виховний процес більш цікавим, інформативним та ефективним у засвоєнні знань з боку слухачів/студентів.

1.2 Аналіз існуючих аналогів

Перед початком роботи над проектом був проведений аналіз існуючих аналогів. Адже таке явище як сумісність веб та тривимірного моделювання набирає популярності як і у процесі сайтобудування, так і у створенні веб-ресурсів для покращення роботи інженерів та дизайнерів.

Перший аналог – Sketchfab. Це платформа для публікації, поширення, пошуку, купівлі та продажу 3d, VR та AR контенту. Даний ресурс надає можливість користувачу відображати 3d моделі в Інтернеті для перегляду в будь-якому браузері, як на персональному комп'ютері, так і на мобільному пристрою, наприклад, смартфон, планшет. Для роботи з Sketchfab можна використовувати гарнітуру віртуальної реальності. Дана платформа була запущена на просторі Інтернету в березні 2011 року. Цей ресурс надає можливість використовувати свій функціонал як для користувачів, що мають звичайний профіль(безкоштовний), так і для тих, що мають профіль-преміум. Проте при наявності сплаченого профілю користувачу надається більше можливостей у роботі з моделями та її поширенням у приватному порядку. Юзери Sketchfab можуть створювати свої власні 3d моделі, доступні для завантаження по ліцензії Creative Commons та продавати їх в магазині Sketchfab. 3d моделі можуть бути завантаженими в Sketchfab з його веб-сервера, з різних 3d програм, використовуючи плагіни (наприклад, для 3ds Max) або безпосередньо з програм(Blender, Adobe Photoshop)[9].

Початкова сторінка Sketchfab представлена на рис. 1.1. Якщо користувач використовує даний ресурс вперше, то при натисканні на будь-який з пунктів з'явиться вікно для авторизації\реєстрації, що представлене на рис. 1.2.

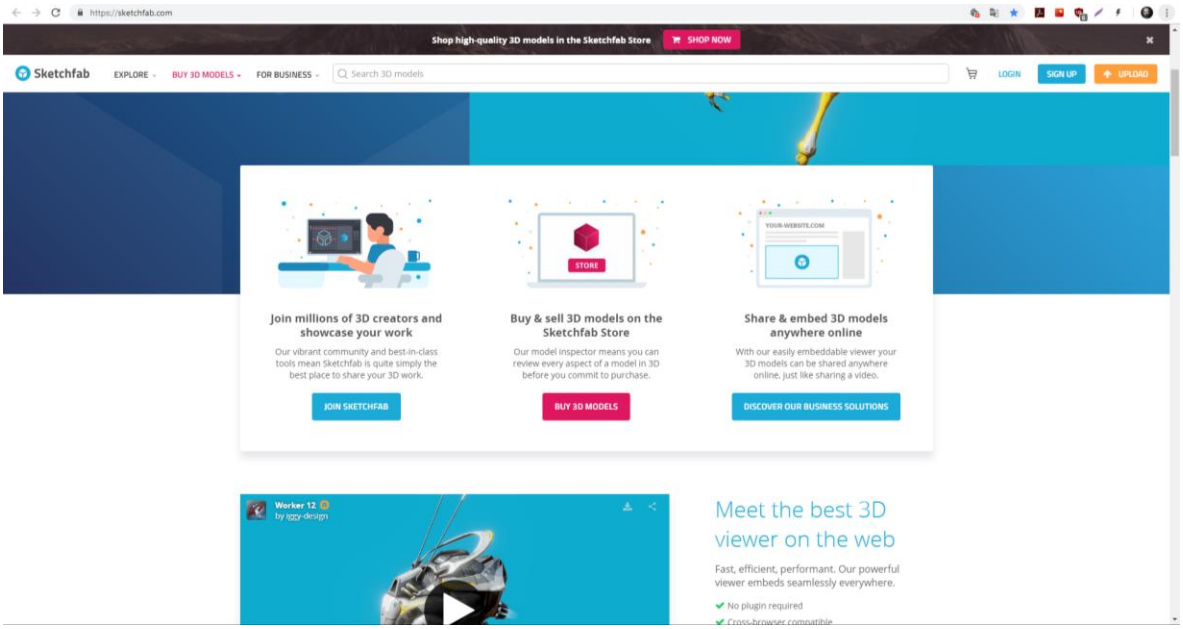

Рисунок 1.1 – Початкова сторінка Sketchfab

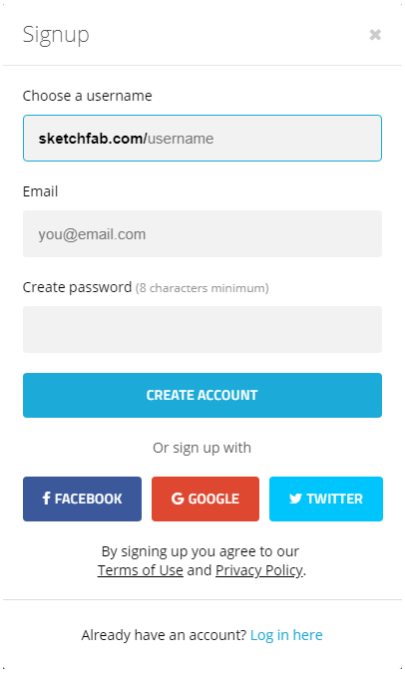

Рисунок 1.2 – Вікно для авторизації та реєстрації.

Після того, як користувач виконає авторизацію\реєстрацію, він отримає безкоштовний профіль. При використанні такого типу акаунту користувач зможе лише працювати з редактором 3d моделей без можливості її поширення. Також функції купівлі та продажу стають доступними лише при наявності преміум-профілю.

Наступний аналог - Blend4Web. Даний ресурс позиціонує себе як безкоштовний та відкритий у доступі фреймворк, який призначений для створення та відображення інтерактивної тривимірної графіки в браузерах. Blend4Web використовує відкритий пакет 3d моделей, що були створені у програмного продукті Blender[4]. Для того, щоб використовувати даний ресурс, необхідно перш за все завантажити інсталяційний пакет Blender з сайту розробників. Наступним кроком є завантаження безпосередньо Blend4Web з офіційного сайту https://www.blend4web.com. Після користувач повинен виконати налаштування для Blender: встановити завантаження скриптів безпосередньо з каталогу Blend4Web та виконати активацію експорту/імпорту з Blend4Web. Лише після цих операцій юзер має можливість працювати з моделями, що завантажені в Blend4Web в середовищі Blender та поширювати свої роботи на сайт Blend4Web. Початкова сторінка даного ресурсу представлена на рис. 1.3.

Рисунок 1.3 – Початкова сторінка Blend4Web

У вкладці Support & Services користувач може знайти ряд англомовних відео-роликів, які слугують гідами для встановлення та коректного використання Blend4Web.

Наступний аналог – Modelo.io. Це веб-платформа для презентації та сумісної роботи архітекторів та інженерів, які працюють з інструментами САПР такими, як Rhino, Revit, SketchUp. Даний ресурс був розроблений для користувачів, які працюють у сфері архітектури та промислового дизайну. Компанія Modelo розробила веб-платформу для підвищення ефективності в функції сумісної роботи, презентації та управління проектами творчих дизайнерських команд[8].

Початкова сторінка Modelo.io, що представлена на рис. 1.4, супроводжується відео-роликами, де в прискореному темпі відображається процес роботи з редактором.

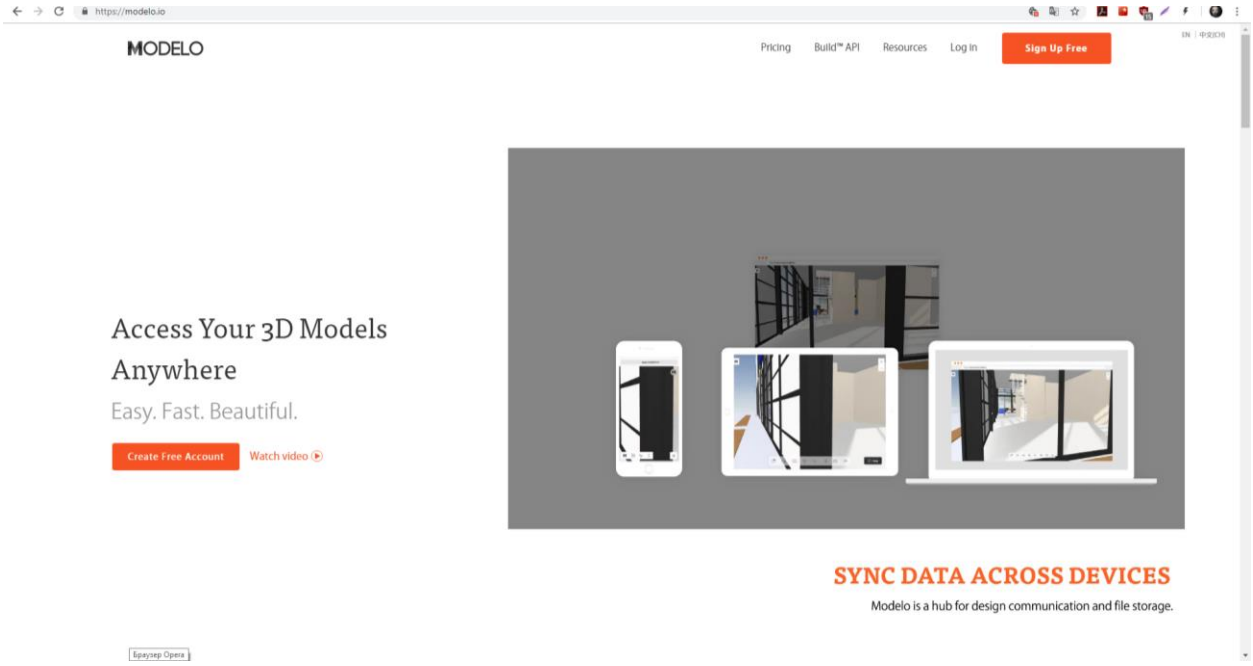

Рисунок 1.5 – Початкова сторінка Modelo.io

Перш ніж працювати з веб-платформою необхідно виконати авторизацію/реєстрацію. Якщо порівнювати реєстрацію на веб-ресурс Sketchfab, то даний процес більш важкий на Modelo.io. Адже користувач не має можливості автоматичного заповнення даних, використовуючи профіль Google або Facebook. Також при реєстрації на Modelo.io користувач має вказати свій мобільний номер, дане поле є обов'язковим для заповнення, хоча після реєстрацію користувач не отримує ніякого сповіщення чи код для активації профіля на мобільний телефон. Після того, як користувач виконав авторизацію, він має доступ до свої попередніх проектів або може створити новий. Структура проекту:

- вкладка Project Overview, де міститься інформація про строки та короткий опис проекту, про замовника та всіх членів проекту;
- вкладка Models, де користувач має можливість переглянути раніше створені моделі та завантажити нові з носія(ПК) у форматах .zip, .ifc, .3dm, .skp;
- вкладка Team Members, де можна додати/видалити учасників проекту;

Даний ресурс є корисним, у випадку його використання при реалізації масштабного проекту, де учасники можуть контролювати один одного та відслідковувати кожний етап проектування одночасно. Але при індивідуальній роботі використання Modelo.io є нераціональним.

Та останній аналог - VERGE3D. Це движок нового покоління, призначений для створення онлайн конфігураторів, презентацій товарів на просторах Інтернет-магазинів, а також вивчення ігор та програм. VERGE3D працює у будь-яких веб-браузерах без встановлення допоміжних плагінів. Для створення контенту використовуються такі програмні продукти, як 3ds Max або Blender, а також візуальний редактор логіки Puzzles[10]. Проте движок VERGE3D надає користувачу повний функціонал для ефективної роботи при купівлі ліцензії на даний продукт. Щоб уникнути конфлікту у сумісності движка та програмного забезпечення, розробники наполягають на тому, щоб версія 3ds Max була не нижче 2017 року. На офіційному сайті https://www.soft8soft.com/ користувачі за

необхідності можуть знайти безкоштовну версію VERGE3D, яка має строк дії 30 днів. Початкова сторінка офіційного сайту VERGE3D представлена на рис. 1.6.

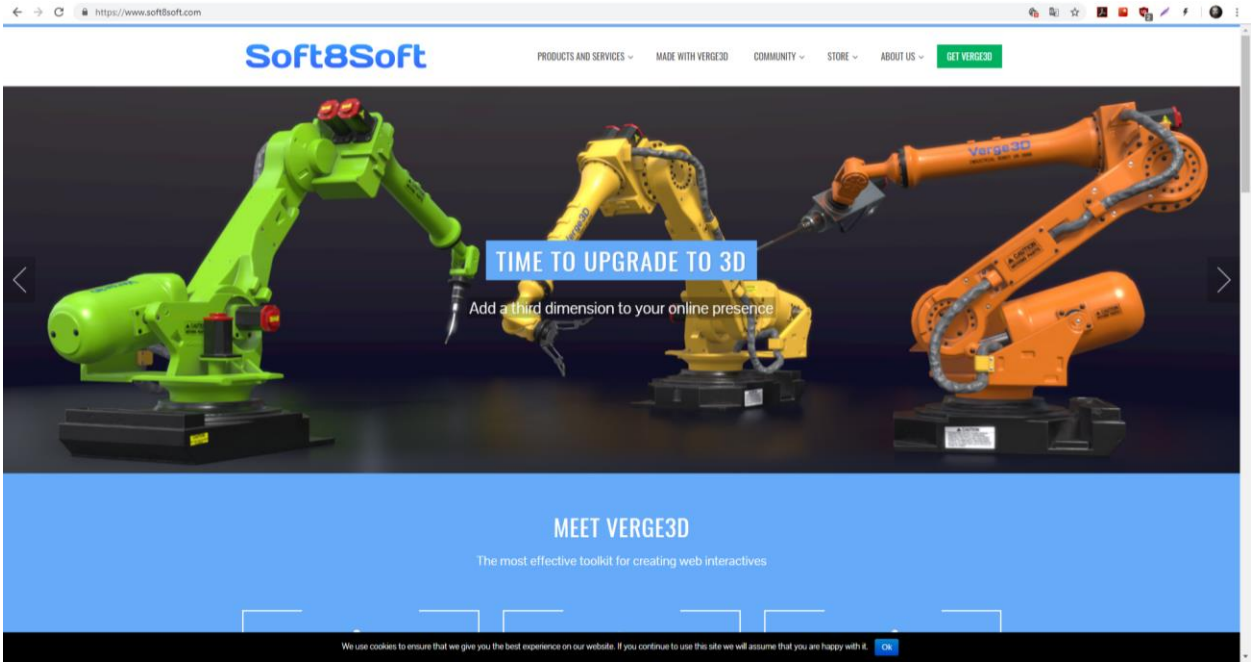

Рисунок 1.6 – Початкова сторінка VERGE3D

На основі проведеного аналізу була створена порівняльна характеристика аналогів та розробленого веб-ресурсу по кожному критерію, що представлена у табл 1.1.

Таблиця 1.1 – Порівняльна характеристика аналогів

Назва критерію	Назва ресурсу				
	Sketchfab	Blend4Web	Modelo.io	VERGE3D	ITP Kaleidoscope
1. Використання ресурсу без реєстрації	-	-	-	-	+

Продовження таблиці 1.1

Назва критерію	Назва ресурсу				
	Sketchfab	Blend4Web	Modelo.io	VERGE3D	ITP Kaleidoscope
2. Можливість завантаження моделі з розширенням .obj	-	-	-	-	-
3. Можливість роботи без інсталяції ресурсу	+	-	+	-	+
4. Використання повного функціоналу при наявності безкоштовного профілю	-	+	+	-	+
5. Можливість поширення моделі на власний сайт	-	+	-	+	+

2. ПОСТАНОВКА ЗАДАЧІ ТА МЕТОДИ ДОСЛІДЖЕННЯ

2.1 Мета та задачі

Метою проекту є створення веб-ресурсу для перегляду 3d моделей. Розроблений продукт буде мати попит перш за все у вищих навчальних закладах, які у своєму складі мають кафедри та спеціальності з нахилом на 3d моделювання та інженерію. При використанні веб-ресурсу викладачі мають змогу підвищити якість навчально-виховного процесу. В свою чергу студенти покращують свої навички у 3d моделюванні, переглядаючи моделі з різних проекційних видів.

Розроблений веб-ресурс має виконувати такі функції:

- зміна кольору моделі;
- налаштування чотирьох джерел світла за моделлю освітлення Фонга;
- вибір бажаного проекційного виду для перегляду моделі;
- масштабування сцени.

Створений веб-ресурс має містити інтуїтивно зрозумілий та зручний інтерфейс для користувача. При наявності англомовного інтерфейсу розроблений продукт може біти використаний не залежно від регіону чи країни.

Для досягнення мети необхідно вирішити певний перелік задач:

- проаналізувати предметну область;
- обрати технологію для відображення 3d моделі на веб-ресурсі;
- розробити структуру веб-ресурсу;
- реалізувати дану структуру у вигляді веб-сторінки;
- розробити функціонал завантаження моделі та базових маніпуляцій з нею;
- провести тестування веб-ресурсу.

Для коректного процесу реалізації необхідна наявність затвердженого технічного завдання, яке наведене у додатку А.

Серед технологій для відображення 3d моделей на веб-сторінці найбільш відомі це - Adobe Flash, Silverlight и WebGL.

Adobe Flash – мультимедійна платформа компанії Adobe Systems для розроблення веб-додатків. Технологія Adobe Flash дає змогу користувачу працювати з векторною, растровою та тривимірною графікою[13]. Перевагою Adobe Flash є те, що технологія підтримується у 99% браузерів за рахунок вбудованого flash – плеєра[3].

Прямим конкурентом Adobe Flash є Silverlight. Microsoft Silverlight – крос-платформна та крос-браузерна технологія, за допомогою якої користувачі можуть створювати інтерактивні Інтернет-додатки, десктопні додатки[7]. Microsoft Silverlight це майже та сама технологія, що і Adobe Flash. Оскільки технологія Silverlight підтримується у браузерах користувача також за рахунок спеціального плагіна(плеєра). Проте є вагома різниця між Adobe Flash та Microsoft Silverlight: перша технологія – це дизайн з додаванням коду мовою ActionScript 3.0, друга – це чисте програмування мовою C#.

На відміну від Adobe Flash та Microsoft Silverlight WebGL – це технологія, що призначена для використання безпосередньо в оболонці веб-браузерів та не вимагає інсталювання додаткових плагінів. Одна із переваг WebGL – додатки розробляються у вигляді веб-сторінки, тобто одна і та сама програма можу бути запущеною як на комп’ютері, так і на мобільному пристрої. Розробка технології WebGL дозволила відображати та виконувати операції з тривимірною графікою одразу на веб-сторінках з використанням мови програмування JavaScript. На даний час розробники веб-сторінок можуть з легкістю створювати зовсім нові, унікальні інтерфейси користувачів, тривимірні онлайн-ігри та використовувати тривимірну графіку для візуалізації різнопланової інформації [17].

Провівши порівняльну характеристику технологій за певними критеріями, була сформована табл.2.1.

Таблиця 2.1 – Порівняльна характеристика технологій

Критерій	Adobe Flash	Microsoft Silverlight	WebGL
Крос-платформність	+	-	+
Захист коду від копіювання	+	+	-
Підтримка мобільних пристроїв	+android -ios	-	+
Інсталювання додаткового плагіну	+	+	-

На основі проведеного аналізу було вирішено, що для розроблення веб-ресурсу з можливістю перегляду 3d моделей, буде використана технологія WebGL. Оскільки за рахунок неї можна створити універсальний додаток, з яким користувач зможе працювати на будь-якому пристрої.

2.2 Вибір засобів реалізації

Для реалізації веб-ресурсу у вигляді одинарної веб-сторінки було обрано такі засоби реалізації: мова гіпертекстової розмітки HTML, каскадні таблиці стилів CSS та мультипарадигменна мова програмування JavaScript.

Мова гіпертекстової розмітки HTML – це базис для веб-сайту, за допомогою якого створюються каркас сторінок, які бачать користувачі у свої браузерах[18]. Існують різні мови гіпертекстової розмітки, проте більшість сайтів створені на базі HTML. Такі сторінки з успіхом зчитаються браузерами, які відображають їх

на екранах будь-яких електронних пристроїв(персональний комп’ютер, смартфон, планшет)[5]. Мова розмітки HTML зазвичай використовується для того, щоб відправити повідомлення браузеру як саме користувач прагне відобразити веб-сторінку. HTML являється легкою мовою у розумінні та вивченні. Існує велика кількість електронних посібників та сайтів, де користувач може переглянути мануал зі створення легкої веб-сторінки або зі синтаксису певного тегу. Однією із важливою задач «всесвітньої павутини» є надання інформації для користувачів за мінімальні строки. Тому розробка веб-сторінок та їх розміщення на просторі Інтернету не лише наддасть користувачу необхідно йому інформацію, та стане рекламою для діяльності компанії розробників. В кінцевому рахунку це принесе прибуток компанії та збільшить заробіток розробників. Максимально у короткі строки для реалізації веб-сторінки підходить лише HTML. Ця мова дає змогу оформляти контент веб-сайту та контролювати її зовнішній вигляд у вікні будь-якого браузера[15]. До того ж сайт, що був реалізований на базі мови HTML, майже на 100% захищені від хакерської атаки. Оскільки HTML файли до моменту завантаження на сервер створюються та зберігаються безпосередньо на пристрої користувача, то власник сайту завжди має доступ до всіх файлів та контроль над ними. Тому не виникає необхідності у періодичному створенні резервних копій[6].

Каскадні таблиці CSS – мова, яка призначена для опису зовнішнього вигляду веб-сторінки. Якщо користувач має каркас сторінки, створений на базі мови HTML, то наступним етапом є дизайнерське оформлення. Звичайно користувач може додати бажаний дизайн безпосередньо у файл HTML за допомогою спеціальних тегів. Проте за допомогою CSS ця задача легко вирішується, особливо якщо розробник має справу з декількома сторінками одночасно. При підключенні CSS стилів до файлу HTML користувач має можливість застосовувати дизайн як до всіх сторінок одночасно, так і визначити унікальне оформлення як для тієї чи іншої сторінки. Тому при наявності CSS

таблиць зміна дизайну сторінки або її окремих елементів займає набагато менше часу, ніж коли користувач повинен перегляну всі файли HTML та віднайти необхідний йому тег. При підключені до файлів HTML каскадних таблиці стилів CSS розробники сайтів значно зменшують фізичний розмір сайту. Також після того, як створений сайт буде завантажений, веб-браузер одразу виконує кешування CSS файлів. Ця операція дозволяє застосувати даний стиль до інших веб-сторінок, що будуть завантажені у майбутньому[20]. З використанням CSS стилів розробники мають змогу створити адаптивний сайт, який буде мати однакове представлення на будь-яких носіях з різним розширенням екрану. Каскадні таблиці CSS мають широкий асортимент стильових атрибутів у порівнянні з HTML. За рахунок чого розробники можуть створювати яскраві та більш привабливі інтерфейси до сайтів[19].

JavaScript – це мультипарадигменна мова програмування. За допомогою JavaScript розробники веб-сайтів мають можливість додати до контенту більш складні та цікаві елементи: відображення інтерактивних мап, анімації 2d/3d графіки, запуск відео в програвачі та інші[21]. Мова програмування JavaScript має велику кількість переваг, які надають їй змогу бути найкращій серед собі подібних, особливо в певний варіантах використання. Короткий перелік переваг використання JavaScript:

- користувач не повинен інсталювати компілятор, оскільки веб-браузер інтерпретує його за допомогою HTML;
- можливість роботи з JavaScript без підключення до мережі Інтернет;
- JavaScript вважається легкою мовою програмування у вивченні;
- помилки набагато швидко виявляються, в наслідок чого швидко виправляються;
- JavaScript легко з’єднується до спеціальних елементів сторінок або подій, таких як: click або mousemove;

- JavaScript працює в різних браузерах та на різних платформах;
- користувач може використовувати JavaScript для перевірки коректності вхідних даних, що зменшує необхідність у ручній верифікації даних;
- засоби JavaScript дозволяють зробити сайт більш привабливим та інтерактивним для користувачів[22];

Одним із яскравих та видовищних способів застосування JavaScript є робота з графікою. Використання JavaScript дозволяє отримати цікаві ефекти, які розробники можуть застосувати до векторної або растрової графіки. Також існує велика кількість додаткових бібліотек, за допомогою яких користувачі можуть додати повноцінні та унікальні у дизайні графіки, створити персонажі та інші елементи до майбутніх онлайн-ігор.

Говорячи про візуалізацію 3d моделей на веб-сторінці, гармонійне та безконфліктне поєднання – це WebGL та JavaScript. WebGL дає змогу веб-контенту використовувати API для відображення тривимірної графіки без використання додаткових плагінів. Елемент мови HTML canvas виконує роль полотна, на якому користувач може працювати з тривимірною графікою. JavaScript безпосередньо відповідає за виклик функції, яка буде виконувати процес відображення графіки на полотні. Більшість WebGL програм складаються із коду керування, що написаний на мові програмування JavaScript, та коду спеціальних ефектів(шейдерів), що виконується на графічному процесорі[14]. Оскільки сучасні веб-браузери підтримують JavaScript та більшість з них вже мають змогу використовувати WebGL у контенті, то поєднання JavaScript з WebGL вважається на даний час найбільш актуальним та раціональним.

2.3 Планування робіт

Після того, як було сформовано мету розробки проекту, визначено перелік задач для її реалізації та відібрано необхідний інструментарій, наступним етапом є планування робіт. На цьому етапі перш за все було проведено деталізацію мети створення веб-ресурсу за методологією SMART. Далі були розроблені діаграми OBS та WBS, що відображають ієрархічну структуру плану робіт по створенню продукту та зазначають учасників, що несуть відповідність за той чи інший етап. Наступним етапом є формування діаграми Ганта та мережі PDM. Ці діаграми є невід'ємним елементом у контролі строків для реалізації проекту. За допомогою мережі PDM менеджер проекту може з легкістю визначити мінімальний строк завершення проекту та окремих робіт/процесів, за рахунок чого зручно збільшити або зменшити тривалість критичних процесів, що приведе до оптимальної тривалості всього проекту. За допомогою діаграми Ганта легко відслідковується процентне завершення кожного із етапу робіт по проекту. Далі були визначені всі можливі ризики, що можуть виникнути при реалізації, та були сформовані план запобігання та план реакції для кожного з них. В кінці було розраховано бюджет проекту, а саме заробітна плата для кожного із учасників проекту. Вся детальна інформація, діаграми, мережі та супровідні таблиці представлені у Додатку Б.

3. ПРОЕКТУВАННЯ ВЕБ-РЕСУРСУ ДЛЯ ПЕРЕГЛЯДУ 3D МОДЕЛЕЙ

Після того як було проведено аналіз предметної області, визначено мету проекту та задачі, які необхідні для його реалізації, сформовано список вимог до функціоналу майбутнього продукту та обрано засоби реалізації, наступним етапом є проектування веб-ресурсу. Під терміном проектування продукту зазвичай розуміють процес реалізації зовсім нового продукту на ринку IT, що базується саме на необхідності в ньому як з боку користувача, так і підприємств.

3.1 Діаграми нотації IDEF0

Процес проектування веб-ресурсу необхідно розпочинати з розробки контекстної діаграми A-0. Даний крок обумовлений тим, що контекстна діаграма містить короткі, чіткі та лаконічні твердження, що дають загальний опис до системи та її взаємодії з навколишнім середовищем. Головними елементами даної діаграми є:

- Вхідні дані: інформаційні та матеріальні дані, що існують на стадії ініціалізації продукту.
- Вихідні дані: результат, який буде отриманий після реалізації продукту.
- Управління: інформаційні та матеріальні дані, які необхідні під час реалізації.
- Механізми: команда проекту, програмне та апаратне забезпечення.

Провівши аналіз відносно головних елементів для контекстної діаграми «Розробка веб-ресурсу для перегляду 3d моделей», було сформовано перелік даних:

- Вхідні дані: 3d модель у форматі json, технічне завдання на розробку веб-ресурсу.
- Вихідні дані: веб-ресурс.
- Управління: методологія створення веб-ресурсів, загальноприйняті принципи макетування.
- Механізми: команда проекту, технічне забезпечення, HTML, CSS, Photoshop, JavaScript, WebGL.

На основі цих даних була розроблена контекстна діаграма, що представлена на рис.3.1. Діаграма A-0 була розроблена у програмному продукті AllFusion Process Modeler.

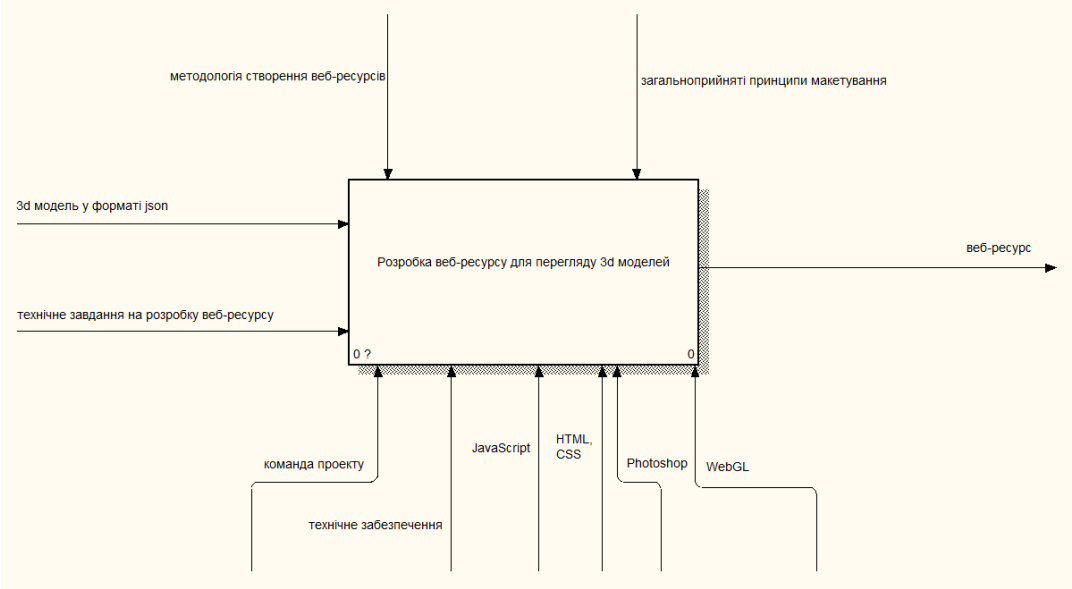

Рисунок 3.1 – Контекстна діаграма

Оскільки контекстна діаграма дає лише загальний опис системи, то виникає необхідність у виконанні декомпозиції. Даний процес дозволить детально

ознайомитись з логікою послідовності виконання робіт для реалізації фінального продукту.

Діаграма А-0 була розбита на два підрівні, а саме: розробка інтерфейсу користувача та розробка логіки роботи ресурсу. Це обумовлено тим, що процес створення функціонального веб-ресурсу поділяється на два етапи:

- Перший етап: формування каркасу веб-сторінки та подальше її оформлення.
- Другий етап: додавання логіки для можливості реакції на дії користувача.

При описі першого етапу були сформовані такі дані:

- Вхідні дані: технічне завдання на розробку веб-ресурсу.
- Вихідні дані: сторінки у форматі HTML.
- Управління: методологія створення веб-ресурсів, загальноприйняті принципи макетування.
- Механізми: команда проекту, технічне забезпечення, HTML, CSS, Photoshop.

За такою самою аналогією були сформовані дані і для другого етапу:

- Вхідні дані: 3d модель у форматі json, технічне завдання на розробку веб-ресурсу, сторінки у форматі HTML.
- Вихідні дані: веб-ресурс.
- Управління: методологія створення веб-ресурсів.
- Механізми: команда проекту, технічне забезпечення, JavaScript, WebGL.

Діаграма другого рівня представлена на рис.3.2.

Після декомпозиції контекстної діаграми на два процеси було виконано розбиття кожного з них на певний перелік робіт, результат кожної з яких є вхідними даними для наступної.

Процес розробки інтерфейсу користувача містить в собі такі етапи, а саме: розробка прототипу веб-ресурсу та розмітка та форматування веб-сторінок. Це пов’язано з тим, що перш ніж виконувати формування каркасу сторінки, необхідно визначитись з її дизайном на основі головних принципів макетування та вимог замовника.

Для процесу розробки прототипу веб-ресурсу були визначені такі дані:

- Вхідні дані: технічне завдання на розробку веб-ресурсу.
- Вихідні дані: готовий дизайн веб-сторінок.
- Управління: загальноприйняті принципи макетування.
- Механізми: команда проекту, технічне забезпечення, Photoshop.

Оскільки процес розмітка та форматування веб-сторінок виконується після прототипування, то перелік даних дещо відрізняється:

- Вхідні дані: технічне завдання на розробку веб-ресурсу, готовий дизайн веб-сторінок.
- Вихідні дані: сторінки у форматі HTML.
- Управління: методологія створення веб-ресурсів.
- Механізми: команда проекту, технічне забезпечення, HTML, CSS.

Діаграма декомпозиції процесу розробки інтерфейсу користувача представлена на рис.3.3.

Лише після формування сторінок у форматі HTML розробник може виконувати додавання логіки для можливості реакції на дії користувача. Даний етап також було розбито на два процеси, оскільки перш за все необхідно додати логіку до панелі керування, а лише потім додати функціонал для візуалізації моделі.

Для етапу опису реакції на дії користувача були визначені такі дані:

- Вхідні дані: технічне завдання на розробку веб-ресурсу, сторінки у форматі HTML.

- Вихідні дані: команда зміни моделі від дії користувача.
- Управління: методологія створення веб-ресурсів.
- Механізми: команда проекту, технічне забезпечення, JavaScript.

Лише після того, як було додано логіку до панелі керування, розробник може переходити до завершального етапу – візуалізація 3d моделі. Для даного процесу визначені такі дані:

- Вхідні дані: технічне завдання на розробку веб-ресурсу, 3d модель у форматі json, команда зміни моделі від дії користувача.
- Вихідні дані: веб-ресурс.
- Управління: методологія створення веб-ресурсів.
- Механізми: команда проекту, технічне забезпечення, WebGL.

Діаграма декомпозиції процесу розробки логіки роботи ресурсу представлена на рис.3.4.

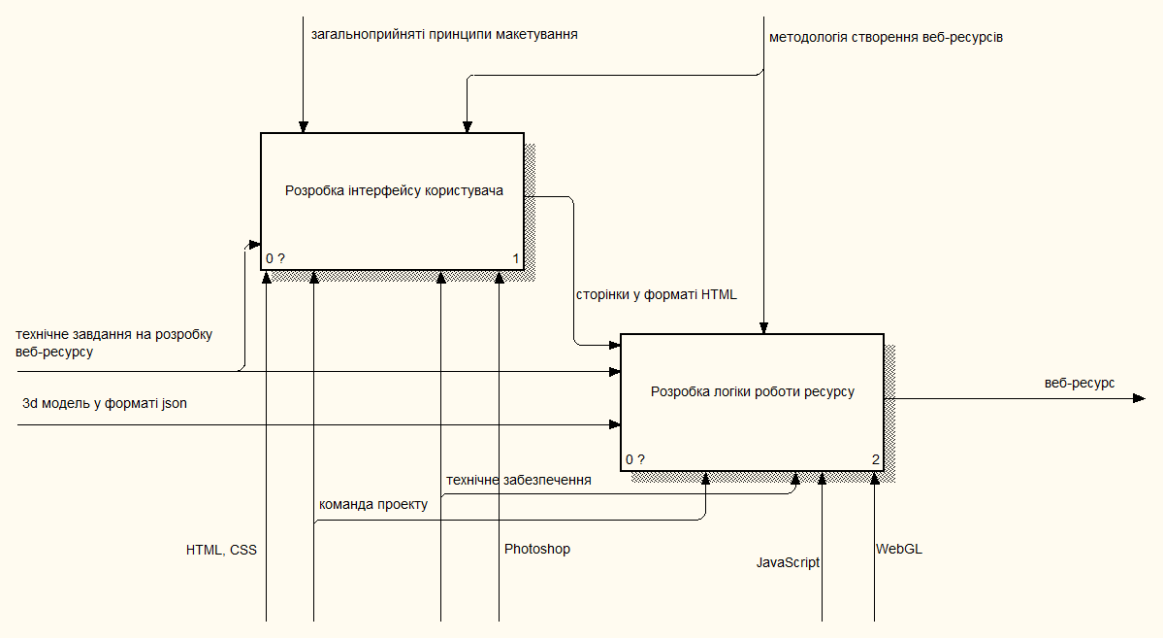

Рисунок 3.2 – Декомпозиція діаграми A-0

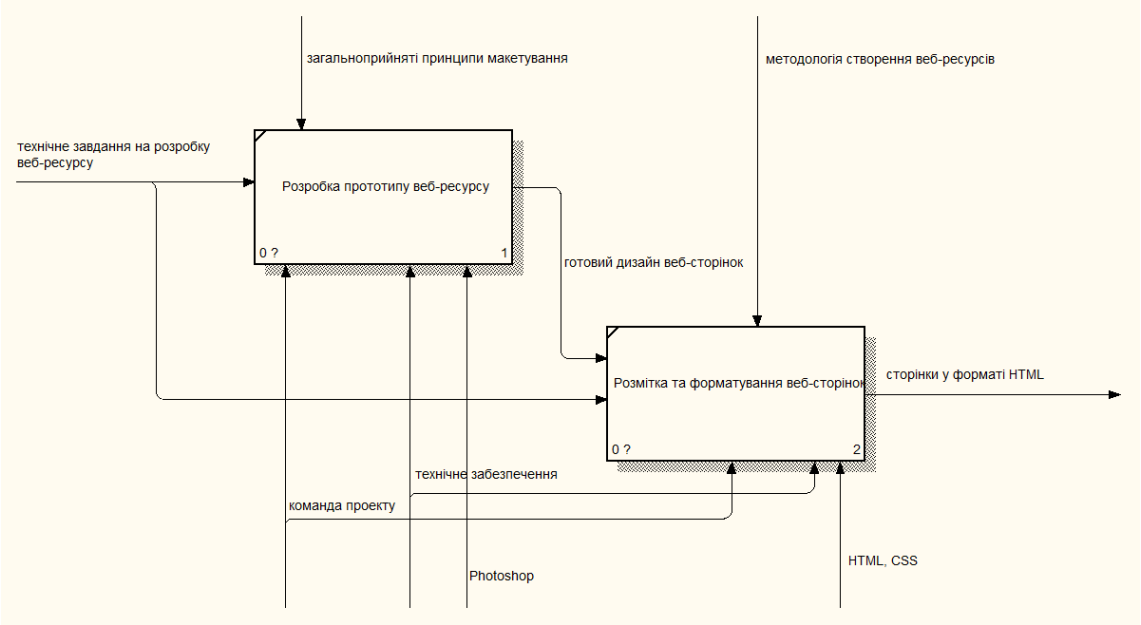

Рисунок 3.3 – Декомпозиція першого етапу реалізації

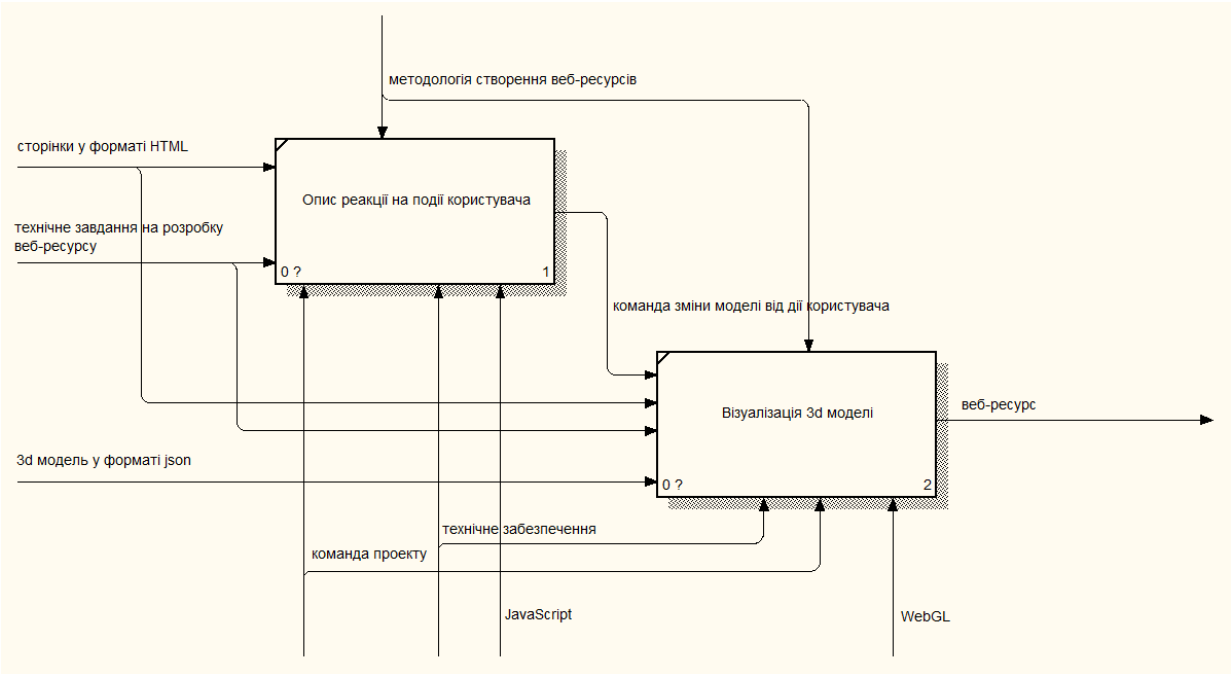

Рисунок 3.4 – Декомпозиція другого етапу реалізації

3.2 Use Case Diagram

Одним із невід’ємних етапів при проектуванні продукту є розробка Use Case Diagram (Діаграма варіантів використання). Даний тип діаграм відображає взаємну залежність між групою варіантів використання та групою акторів, що беруть участь у процесі. Розробка діаграми варіантів використання необхідна для відображення інформації про те, які саме функції має виконувати система та яким чином певний актор має можливість взаємодіяти з нею.

Для розробки діаграми Use Case були визначені такі актори:

- Customer – користувач, який прагне візуалізувати обрану модель та провести певні маніпуляції з нею.
- WebGL – технологія, методи якої дозволяють реалізовувати функції маніпуляції над моделлю.

Після того, як були визначені всі актори, які будуть взаємодіяти з системою, необхідно сформувати перелік варіантів використання. Вони відповідають вимогам, яким повинен відповідати веб-ресурс, що зазначені у Додатку А.

Варіанти використання для веб-ресурсу:

- візуалізація моделі;
- зміна кольору моделі;
- налаштування освітлення;
- налаштування блиску моделі;
- вибір проекційного виду;
- масштабування сцени;
- поворот сцени.

На основі сформованих даних про акторів та всі можливі варіанти використання веб-ресурсу, була розроблена Use Case діаграма. Вона представлена на рис.3.5.

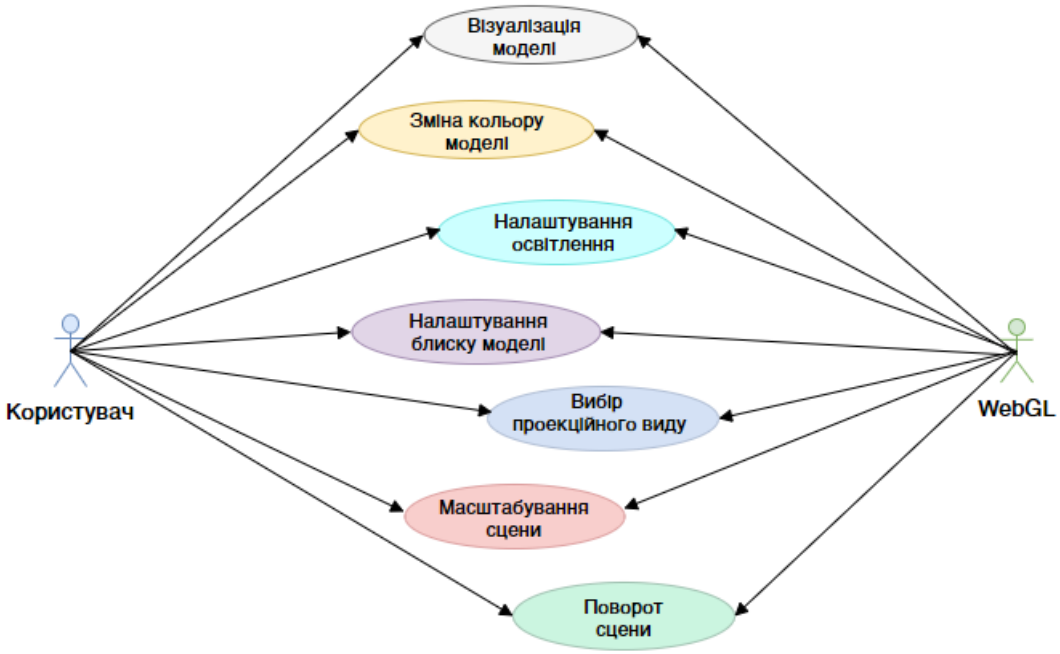

Рисунок 3.5 – Діаграма варіантів використання

4. РОЗРОБКА ВЕБ-РЕСУРСУ ДЛЯ ПЕРЕГЛЯДУ 3D МОДЕЛЕЙ

4.1 Розробка інтерфейсу користувача

4.1.1 Розробка прототипу веб-ресурсу

Перш ніж виконувати формування веб-сторінки за допомогою засобів мови HTML та додавати до неї унікальне оформлення, використовуючи CSS, необхідно виконати прототипування майбутньої сторінки. Під час виконання даного процесу необхідно визначитись з розміщенням та кількістю основних блоків, які відповідають за отримання функціонального продукту. Також необхідно узгодити загальний дизайн з оформлення разом з замовником. При створення макету сторінки необхідно дотримуватись загально прийнятих принципів макетування, а саме:

- обрати універсальний шрифт;
- кольорова гама сторінки має складатися з тих відтінків, зо сприймаються оком людини без подразнень та не відволікають від подальшої роботи;
- розміри всіх елементів повині мати середнє значення масштабу, як для людей з поганим зором, так і для тих, що не мають вад.

Саме після того, як були вирішені ці питання, команда проекту безпосередньо може переходити до створення макету веб-ресурсу, використовуючи програмний продукт Adobe Photoshop. Вибір даного програмного забезпечення є очевидним, оскільки:

- в індустрії сайтобудування даний продукт надає великі можливості при проектуванні макету до сторінки;

- даний програмний продукт дуже зручний та легкий у вивченні;
- останні версії Adobe Photoshop дозволяють створювати макети сайту зі смарт об’єктами, що полегшує у майбутньому процес створення функціонального продукту.

Спираючись на результати під час розробки Use Case діаграми командою проекту було визначено, що майбутній користувач може взаємодіяти як з 3d моделлю, так і зі сценою. Проте розбиття пульту керування на два блоку не є раціональним, оскільки маніпуляції зі сценою необхідно також розбити на 2 блок:

- блок, що відповідає за налаштування чотирьох джерел світла;
- блок, що відповідає за вибір бажаного проекційного виду.

Тому було прийнято рішення про створення пульту керування із трьох блоків, що логічно розділені по функціоналу. Схематичне представлення вмісту кожного із блоків веб-сторінці представлено на рис.4.1.

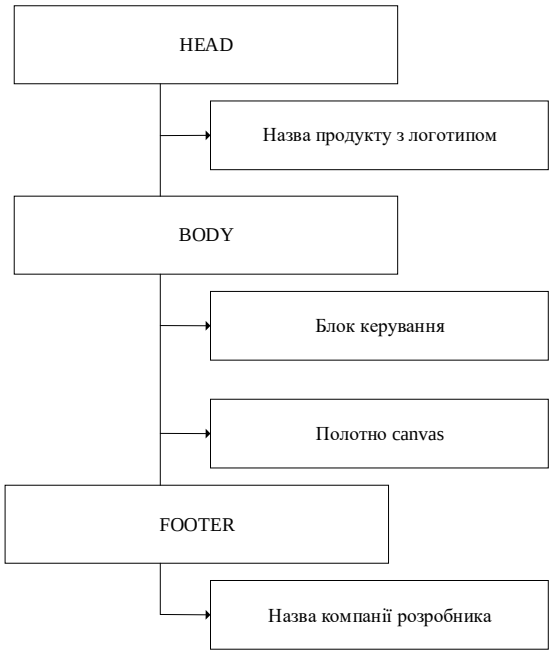

Рисунок 4.1 – Схематичний вміст сторінки

На рис.4.2 представлений макет, що дає загальне уявлення про розміщенні основних блоків на веб-сторінці.

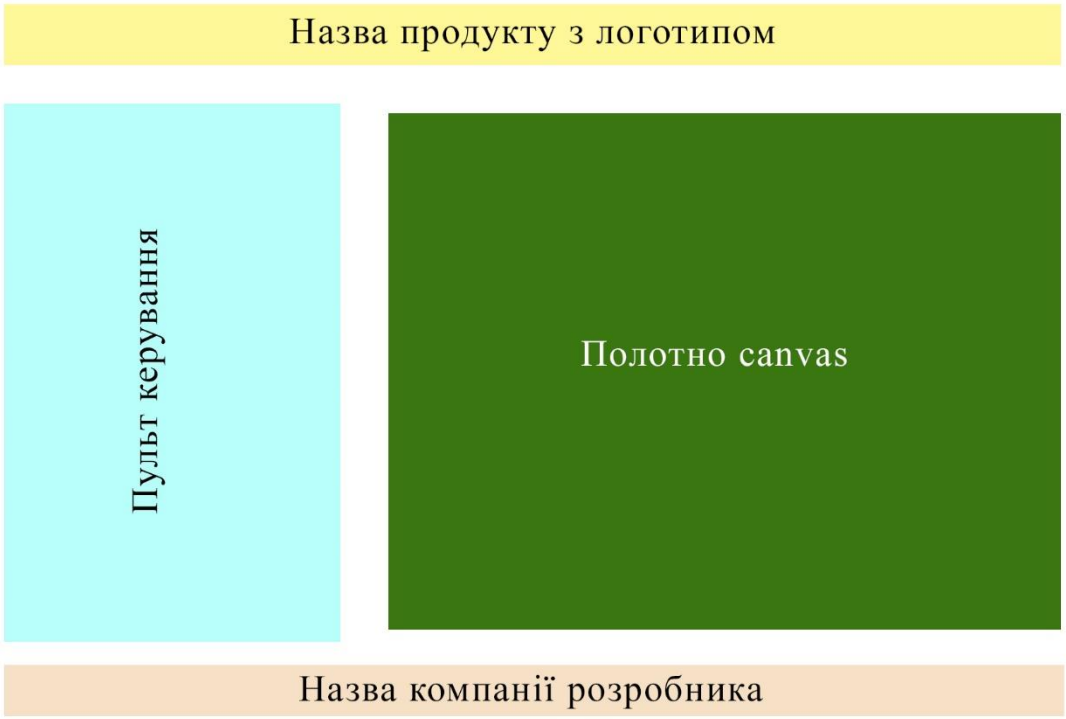

Рисунок 4.2 – Розміщення основних блоків

Після того, як командою проекту та замовником було узгоджено розміщення основних блоків, що будуть формувати веб-сторінку, наступним етапом роботи є розробка прототипу безпосередньо для пульту керування, а саме вибір оптимального виду елементу керування для кожного варіанту використання веб-ресурсу.

Перший блок з пульту керування це блок, що відповідає за вибір моделі, за зміну її кольору та за налаштування її блиску.

Загальний вид макету першого блоку пульту керування представлений на рис.4.3.

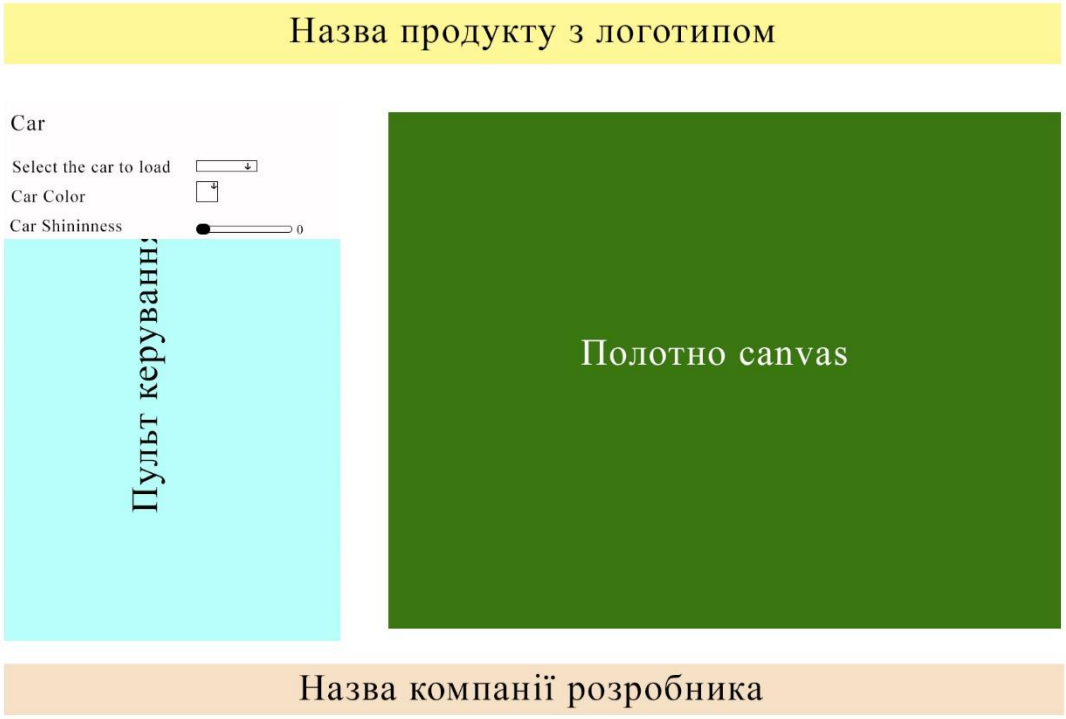

Рисунок 4.3 – Загальний вид першого блоку керування

Макет вмісту першого блоку доповнюється двома супровідними зображення, що дають загальне уявлення про сам механізм вибору моделі та зміну її забарвлення. При виборі моделі користувач може обрати одну із чотирьох, використовуючи при цьому випадаючий список. Для зміни кольору 3d моделі користувачу надається кольорова палітра моделі RGB. Супровідні зображення до першого блоку представлені на рис. 4.4 – 4.5.

Після того, як було виконано макетування першого блоку, то команда проекту переходить до прототипування другого блоку, що відповідає за налаштування чотирьох джерел освітлення за моделлю Фонга.

Макет розміщення елементів керування для другого блоку представлений на рис.4.6.

Останнім блоком в панелі керування є блок, що відповідає за вибір бажаного проекційного виду за рахунок вбудованої камери у майбутній проект. Даний блок буде містити такі елементи керування (кнопки) як:

- вид за замовчуванням(перспектива);
- вид зверху;
- вид спереду;
- вид позаду;
- вид зліва;
- вид справа.

Макет повноцінного блоку керування зображений на рис.4.7.

Після того, як було розроблено макет пульту керування, команда проекту може переходити до загального оформлення сторінки, а саме додати назву продукту, його майбутній логотип та назву компа. Кольорова гама має бути узгоджена з замовником.

Фінальний прототип веб-ресурсу представлений на рис.4.8.

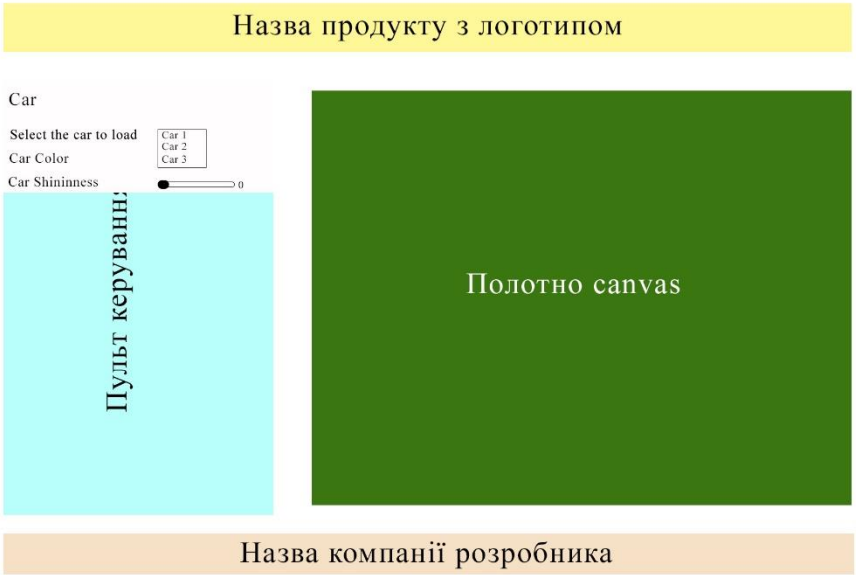

Рисунок 4.4 – Вигляд елементу керування для вибору моделі

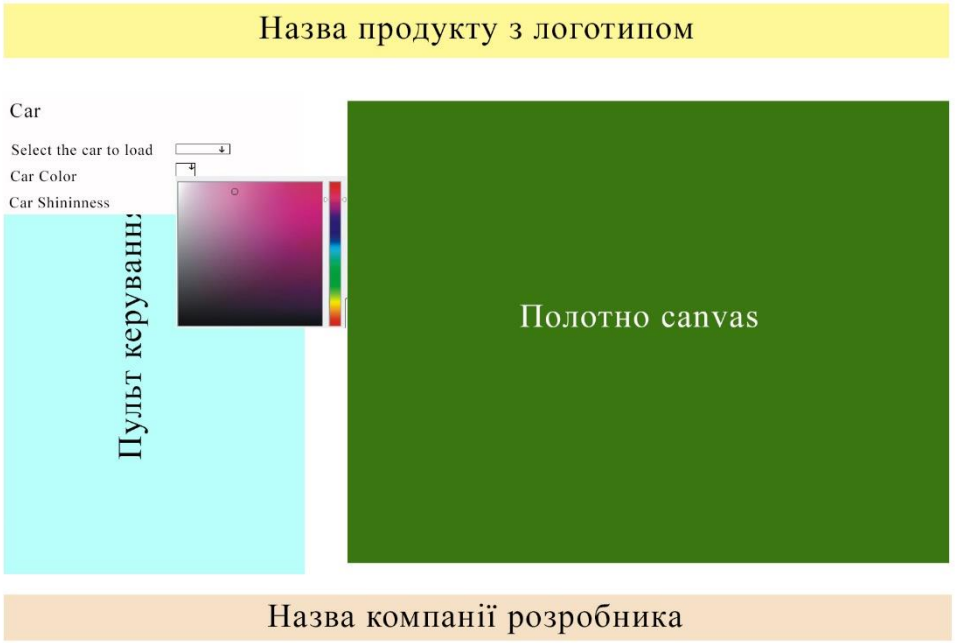

Рисунок 4.5 – Вигляд елементу керування для зміну кольору моделі

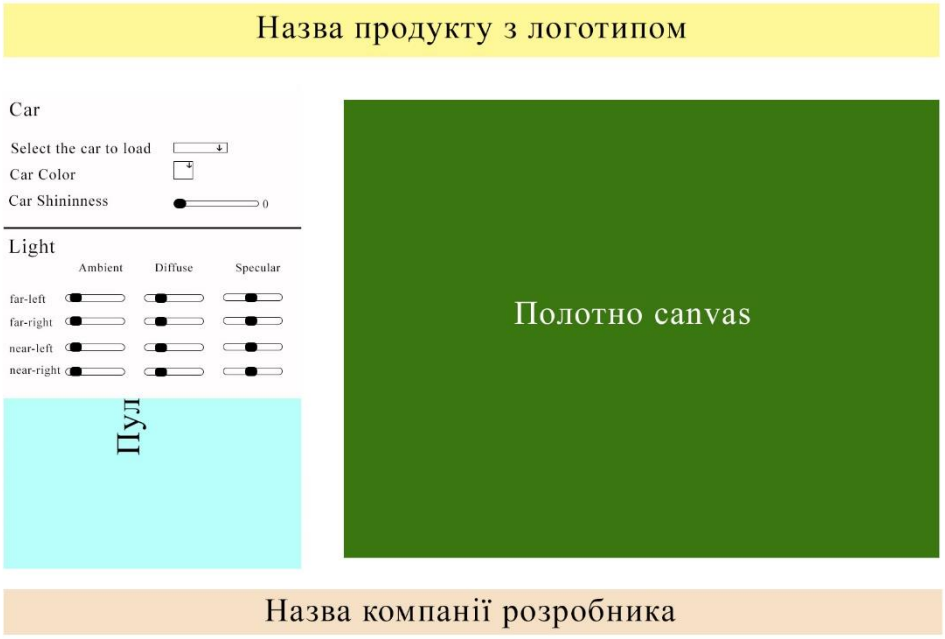

Рисунок 4.6 – Прототип другого блоку пульту керування

Рисунок 4.7 – Прототип третього блоку пульту керування

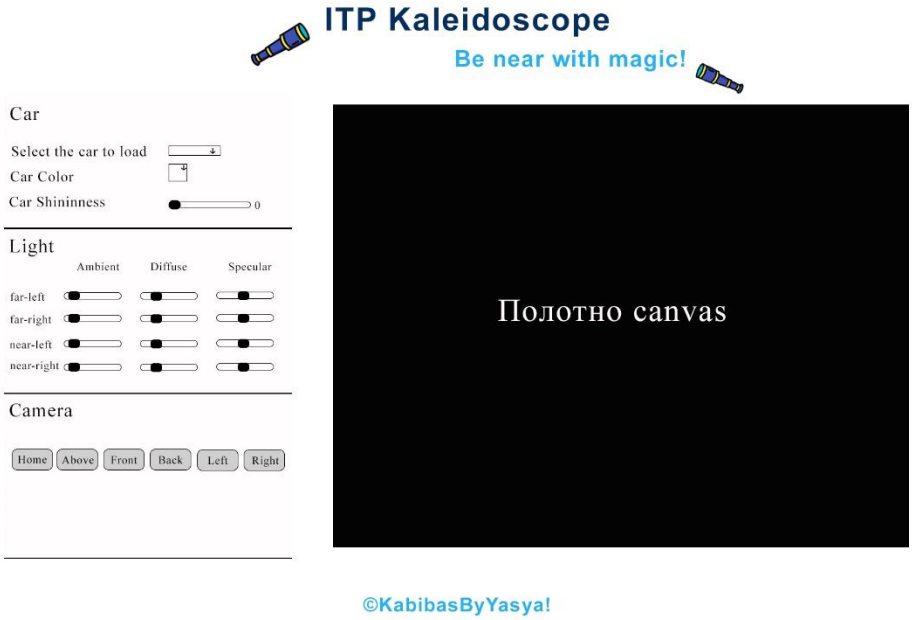

Рисунок 4.8 – Фінальний вигляд прототипу веб-ресурсу

4.1.2 Розмітка та форматування веб-сторінок

Після того, як було проведено прототипування веб-ресурсу команда проекту може переходити до формування сторінки та її дизайнерського оформлення засобами мов HTML та CSS. Формування сторінки виконується у відповідності до схеми змісту сторінки, що приведена у попередньому розділі.

Основний контент сторінки міститься у середній частині сторінки html, у тегу <body>. Детальний опис кожного блочного елементу зазначено у табл.4.1.

Таблиця 4.1 – Опис блочних елементів тегу <body>

Назва	Пояснення
Header	Блок, що містить в собі назву продукту
telescope1 telescope2	Блоки, що містять в собі растрові зображення, які фігурують у назві продукту
Nav	Блок, що містить в собі елементи пульту керування
Content	Блок, що містить в собі полотно canvas
Footer	Блок, що містить інформацію про компанію-розробника

До файла веб-сторінки також були додані файли коду формату css. Детальний опис для кожного файлу CSS приведений у табл.4.2.

Таблиця 4.2 – Опис підключених файлів css

Назва	Пояснення
Styles	Містить форматування для заголовків, абзаців, таблиць
Div	Містить форматування для блочних елементів, а саме: header, nav, content

Продовження таблиці 4.2

Назва	Пояснення
jquery-ui-1.8.13.custom	Додатковий файл для оформлення пульту керування, а саме кнопок та повзунків
colorpicker	Додатковий файл для формування кольорової палітри моделі RGB

Файли коду HTML та CSS, що відповідають за розмітку та форматування веб-ресурсу, представлені у Додатку С.

Загальний вигляд веб-ресурсу представлений на рис.4.9.

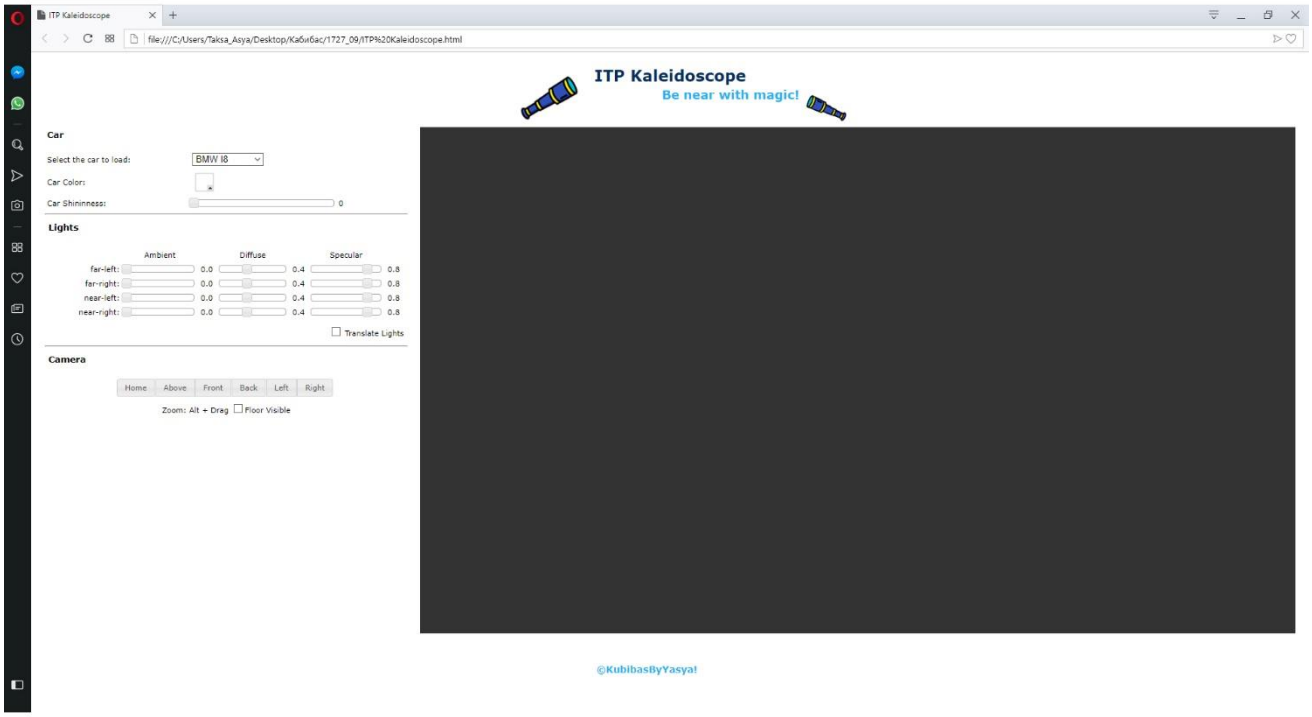

Рисунок 4.9 – Фінальний вигляд веб-ресурсу

4.2 Розробка логіки роботи веб-ресурсу

4.2.1 Налаштування та перевірка працездатності веб-браузера з WebGL

Як і в будь-якій 3d графічній бібліотеці, так і в WebGL, необхідно мати певні компоненти для створення повноцінної 3d сцени, а саме:

- полотно canvas: місце, де будуть відображатися сцена. Це стандартний елемент HTML;
- об’єкти: 3d моделі, що будуть відображатися на сцені;
- джерела світла: WebGL містить в собі шейдери, що дозволяють моделювати ліхтарі на сцені;
- камери: засобами WebGL існує можливість виконання різних матричних операцій, що дозволяє обертати сцену та обирати бажаний проекційний вид.

Перш етапом у роботі з WebGL є знайомство з полотном canvas та його можливостями.

Фрагмент коду для створення полотна на веб-сторінці представлений на рис.4.10.

```
<html>
<head>
    <title> WebGL Beginner's Guide - Setting up the canvas </title>
    <style type="text/css">
    canvas {border: 2px dotted blue;}
    </style>
</head>
<body>
<canvas id="canvas-element" width="800" height="600">
Your browser does not support the HTML5 canvas element.
</canvas>
</body>
</html>
```

Рисунок 4.10 – Створення полотна canvas

Результат даного коду представлений на рисунку 4.11.

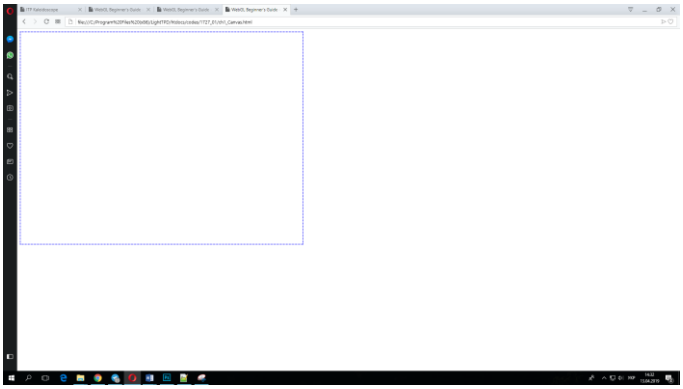

Рисунок 4.11 – Відображення canvas на веб-сторінці

Якщо користувач після запуску коду не побачить полотно canvas, то він отримає повідомлення про те, що браузер не підтримує елемент canvas HTML. Проте в даному випадку є можливість налаштування браузера у вкладці «Конфігурація», де необхідно увімкнути властивість webgl.disabled. Якщо навіть після цих дій користувач все ще не бачить полотно, то це говорить про те, що браузер доданий до чорного списку у драйверах графічних картах. Тому єдиний варіант це спробувати запустити код на іншому пристрою.

Контекст WebGL – дескриптор (об’єкт JavaScript), за допомогою якого існує можливість отримувати доступ до атрибутів та функцій WebGL. Для перевірки того, чи підтримує браузер користувача зв’язок полотна та контексту, було розроблено програмний код, що представлений на рис.4.12.

```
<html>
<head>
    <title> WebGL Beginner's Guide - Checking the WebGL Context </title>
    <style type="text/css">
    canvas {border: 2px dotted blue;}
    </style>

    <script>
       var gl = null;

       function getGLContext(){
           var canvas = document.getElementById("canvas-element-id");
           if (canvas == null){
               alert("there is no canvas on this page");
               return;
           }

           var names = ["webgl", "experimental-webgl", "webkit-3d", "moz-webgl"];

           for (var i = 0; i < names.length; ++i) {
               try {
                   gl = canvas.getContext(names[i]);
               }
               catch(e) {}
               if (gl) break;
           }

           if (gl == null){
               alert("WebGL is not available");
           }
           else{
               alert("Hooray! You got a WebGL context");
           }
       }
    </script>
</head>
<body onLoad="getGLContext()">
<canvas id="canvas-element-id" width="800" height="600">
Your browser does not support the HTML5 canvas element.
</canvas>
</body>
</html>
```

Рисунок 4.12 – Перевірка зв’язку полотна та контексту

Якщо програмний код виконанався без помилок, то першим, що побачить користувач – це повідомлення. Воно предсталене на рис.4.13.

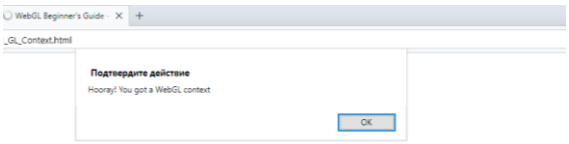

Рисунок 4.13 – Повідомлення про наявність WebGL контексту

Після закриття даного повідомлення, для користувача знову відображається полотно canvas. В результаті чого можна отримати відповідь на запитання: чи підтримується зв’язок між полотном та контекстом? Відповідь – так.

Якщо під час роботи з полотном змінюється значення його атрибутів, то змінюється і контекст WebGL. В будь-який момент користувач має можливість формування запиту до атрибутів.

Фрагмент коду, що дає можливість звертатися до атрибутів контексту, представлений на рис.4.14 – 4.15.

Рисунок 4.14 – Фрагмент коду для отримання значень атрибутів контексту

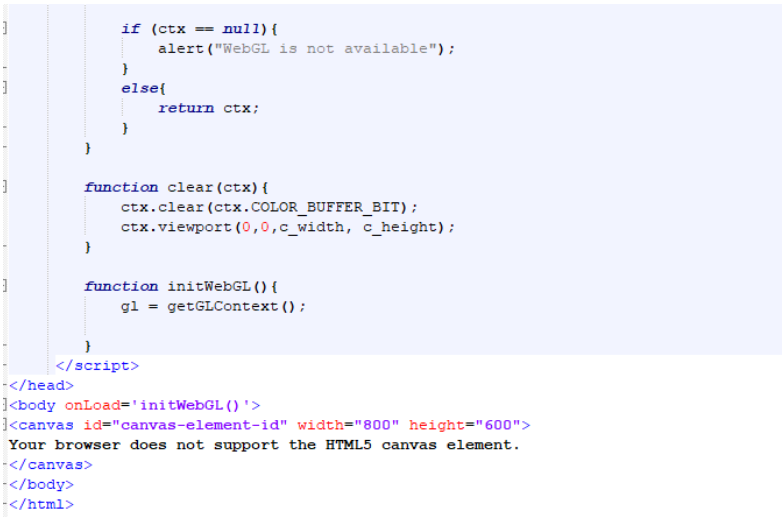

Рисунок 4.15 – Продовження коду

З даного програмного коду видно, що при натисканні клавіші 1 або 2, полотно canvas має змінити своє забарвлення на певний колір.

Результат даної операції представлений на рис.4.16-4.17.

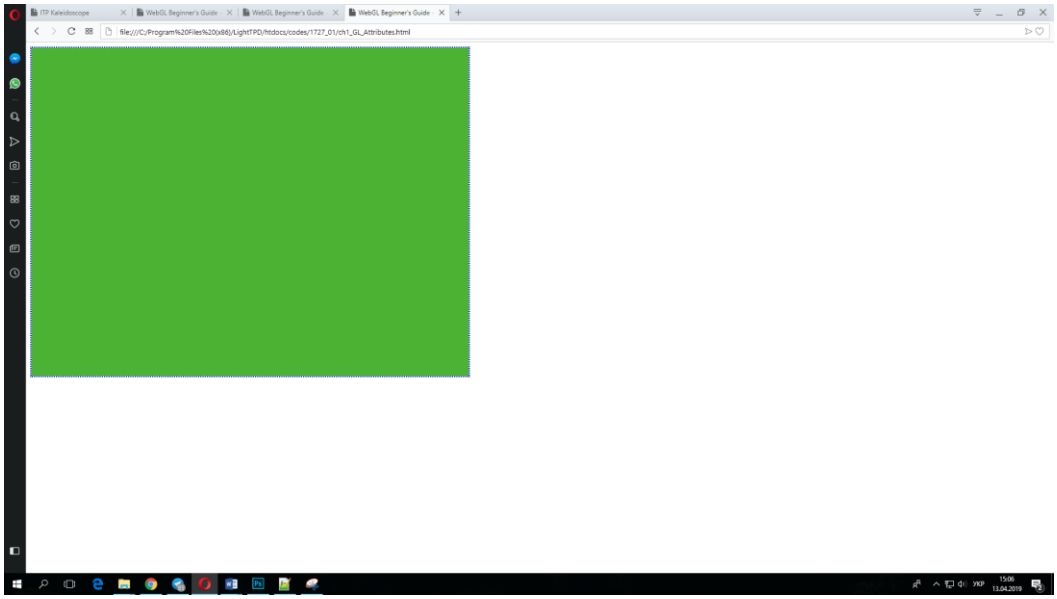

Рисунок 4.16 – Зміна кольору полотна при натисканні клавіші 1

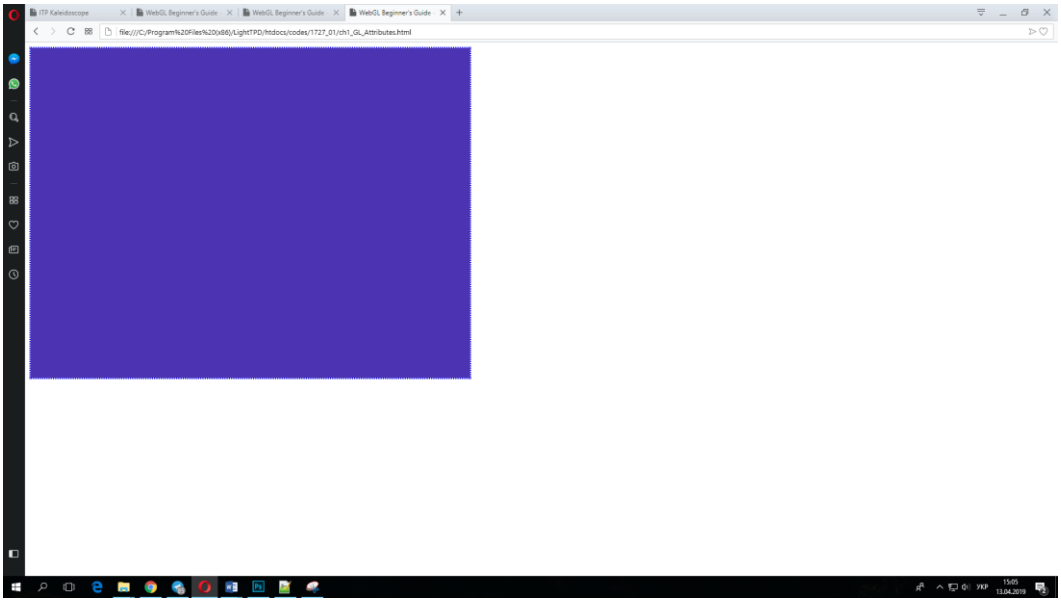

Рисунок 4.17 – Зміна кольору полотна при натисканні клавіші 1

Також з даного програмного коду видно, що при натисканні клавіші 3, користувач отримує повідомлення зі значення кольору полотна на даний момент часу. Результат даної операції представлений на рис.4.18.

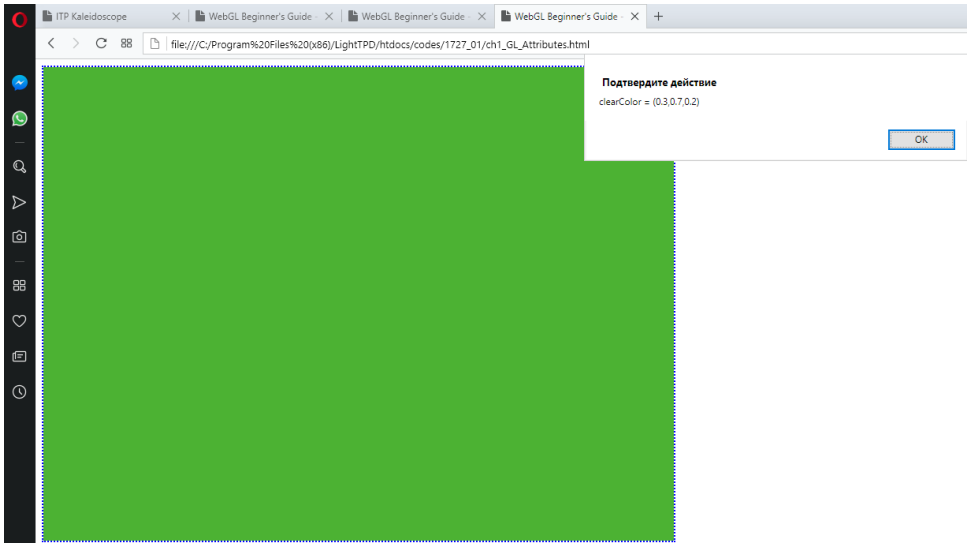

Рисунок 4.18 – Отримання значення кольору полотна на даний час

Таким чином при першому знайомстві з WebGL користувачу необхідно переконатися у можливості відображення полотна canvas та можливості підтримки WebGL при використанні стаціонарного браузера. Також необхідно переконатися у передачі даних між полотном та контекстом WebGL.

4.2.2 Опис реакції на дії користувача

Після того, було виконано макетування веб-ресурсу та створено веб-сторінку з дизайнерським оформленням, наступним етапом роботи є додавання логіки роботи веб-ресурсу. Перш ніж переходити безпосередньо до розробки функцій маніпуляцій над 3d моделлю, базуючись на методах та атрибутах технології WebGL, необхідно додати до самого пульту керування та полотна

canvas процес реакції на дії користувача. Даний етап роботи реалізується з допомогою засобів мови JavaScript.

Перш за все необхідно реалізувати функцію, яка визначить точку входу до програми, тобто початок роботи програмного коду.

Фрагмент коду, що відповідає за реалізацію функції, представлений на рис.4.19.

```
var app;
function runShowRoom(){
    app = new WebGLApp("the-canvas");
    app.configureGLHook = configure;
    app.loadSceneHook   = load;
    app.drawSceneHook   = render;
    app.run();
}
```

Рисунок 4.19 – Точка входу у програму

Фрагмент коду, що відповідає за реалізацію адаптації розміру полотна canvas в залежності від його вмісту, представлений на рис. 4.20.

```
function resizeCanvas(){
    c_width = $('#content').width();
    c_height = $('#content').height();
    $('#the-canvas').attr('width',c_width);
    $('#the-canvas').attr('height',c_height);
}

$(window).resize(function(){resizeCanvas();});
```

Рисунок 4.20 – Функція адаптації розміру полотна canvas

Після того, як були реалізовані етапи встановлення точки запуску програмного коду та адаптування розміру полотна, необхідно додати функціонал до самого пульту керування. Файли коду, що відповідають за додавання панелі керування та додавання логіки до неї, винесені у Додаток С.

4.2.3 Розробка функцій маніпуляцій над моделлю

Після того, як командою проекту було додано логіку роботи веб-ресурсу наступним етапом роботи є безпосередня розробка функціоналу веб-ресурсу, використовуючи методи технології WebGL.

Розробка кожної операції маніпуляції над моделлю реалізовувалась на прикладі роботи з примітивними моделями. Перш за все командою проекту було детально досліджено механізм відображення 3d моделі на веб-сторінці.

WebGL обробляє геометрію стандартним способом, незалежно від складності та кількості точок, які можуть мати поверхні. Є два типи даних, які є фундаментальними для подання геометрії будь-якого тривимірного об'єкту: вершини і індекси. Вершини - це точки, які визначають кути тривимірних об'єктів. Кожна вершина представлена трьома числами з плаваючою точкою, які відповідають координатам x, y і z. На відміну від OpenGL, WebGL не надає методи API для передачі незалежних даних вершин в конвеєр рендеринга, тому користувачу необхідно записати всі вершини в масив JavaScript, а потім створити буфер вершин WebGL з ним. Індекси - це числові мітки для вершин в даній тривимірній сцені. Індекси дозволяють сказати WebGL як з'єднати вершини, щоб отримати поверхню. Як і у випадку з вершинами, індекси зберігаються в масиві JavaScript, а потім вони передаються в конвеєр рендеринга WebGL використовуючи індексний буфер WebGL. В процесі відображення моделі на веб-сторінці участь приймають два шейдери:

- шейдер вершин - маніпулює даними для кожної вершини, такими як координати вершини, нормалізує обмін речовин, кольору і координати текстури;
- шейдер фрагментів – шейдер, що виконує обчислення кольору для кожного пікселя фігури(кожного трикутника).

Використовуючи шейдери даного типу дуже легко реалізувати процес візуалізації примітиву, наприклад, кубу чи піраміди на полотні canvas. Проте, якщо говорити про візуалізацію повноцінної та складної моделі, то в парадигмі

технології WebGL використовують файли формату json. Це легкий текстовий відкритий формат, що використовується для обміну даними. JSON зазвичай використовується в якості альтернативи XML. Формат JSON не залежить від мови. Це означає, що є парсери на багатьох мовах для зчитування й інтерпретації об'єктів JSON. Крім того, JSON є підмножиною об'єктної літеральної нотації JavaScript. Отже, це означає, що користувач може визначати об'єкти JavaScript, використовуючи JSON. На рис.4.21 – 4.25 представлені фрагменти коду, що відповідають за додавання шейдеру вершин, шейдеру фрагментів та за сам механізм відображення моделі, використовуючи файл формату json.

```
<script id="shader-vs" type="x-shader/x-vertex">
    attribute vec3 aVertexPosition;
    uniform mat4 uMVMatrix;
    uniform mat4 uPMatrix;

    void main(void) {
        gl_Position = uPMatrix * uMVMatrix * vec4(aVertexPosition, 1.0);
    }
</script>
```

Рисунок 4.21 – Додавання шейдеру вершин

```
<script id="shader-fs" type="x-shader/x-fragment">
    #ifdef GL_ES
    precision highp float;
    #endif

    void main(void) {
        gl_FragColor = vec4(1.0,1.0,1.0, 0.1);
    }
</script>
```

Рисунок 4.22 – Додавання шейдеру фрагментів

```
<script id="code-js" type="text/javascript">

var gl = null; // WebGL context
var prg = null; // The program (shaders)
var c_width = 0; // Variable to store the width of the canvas
var c_height = 0; // Variable to store the height of the canvas

var part = [];    //PARTS LOADED
var vbo = [];
var ibo = [];


var mvMatrix = mat4.create(); // The Model-View matrix
var pMatrix = mat4.create(); // The projection matrix


var angle = 0;


/*
* The program contains a series of instructions that tell the Graphic Processing Unit (GPU)
* what to do with every vertex and fragment that we pass it. (more about this on chapter 3)
* The vertex shader and the fragment shader together are called the program.
*/
function initProgram() {
    var fgShader = utils.getShader(gl, "shader-fs");
    var vxShader = utils.getShader(gl, "shader-vs");

    prg = gl.createProgram();
    gl.attachShader(prg, vxShader);
    gl.attachShader(prg, fgShader);
    gl.linkProgram(prg);

    if (!gl.getProgramParameter(prg, gl.LINK_STATUS)) {
        alert("Could not initialise shaders");
    }

    gl.useProgram(prg);

    prg.aVertexPosition    = gl.getAttribLocation(prg, "aVertexPosition");
    prg.uPMatrix           = gl.getUniformLocation(prg, "uPMatrix");
    prg.uMVMatrix          = gl.getUniformLocation(prg, "uMVMatrix");
}
```

Рисунок 4.23 – Ініціалізація процесу візуалізації моделі

```
function loadModel(){
  for(var i = 1; i < 179; i++){
        var filename = 'models/nissan_gts/pr'+i+'.json';
        loadPart(filename);
    }
}

function loadPart(filename){
    var request = new XMLHttpRequest();
    console.info('Requesting ' + filename);
    request.open("GET",filename);

    request.onreadystatechange = function() {
      if (request.readyState == 4) {
        if(request.status == 404) {
            console.info(filename + ' does not exist');
        }
        else {
            handleLoadedPart(filename,JSON.parse(request.responseText));
        }
      }
    }
    request.send();
}

/**
* Creates the buffers that contain the geometry of the model
*/
function handleLoadedPart(filename,payload) {

    console.info(filename + ' has been retrieved from the server');

    var vertexBufferObject = gl.createBuffer();
    gl.bindBuffer(gl.ARRAY_BUFFER, vertexBufferObject);
    gl.bufferData(gl.ARRAY_BUFFER, new Float32Array(payload.vertices), gl.STATIC_DRAW);
    gl.bindBuffer(gl.ARRAY_BUFFER,null);

    var indexBufferObject = gl.createBuffer();
    gl.bindBuffer(gl.ELEMENT_ARRAY_BUFFER, indexBufferObject);
    gl.bufferData(gl.ELEMENT_ARRAY_BUFFER, new Uint16Array(payload.indices), gl.STATIC_DRAW);
    gl.bindBuffer(gl.ELEMENT_ARRAY_BUFFER, null);

    vbo.push(vertexBufferObject);
    ibo.push(indexBufferObject);
    part.push(payload);

}
```

Рисунок 4.24 – Створення функції для зчитування кожного файлу та створення буферів

```
function drawScene(){

    gl.clearColor(0.5, 0.2, 0.2, 1.0);
    gl.enable(gl.DEPTH_TEST);

    gl.clear(gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT);
    gl.viewport(0,0,c_width, c_height);

    mat4.perspective(45, c_width / c_height, 10, 10000.0, pMatrix);
    mat4.identity(mvMatrix);
    mat4.translate(mvMatrix, [0.0, -200.0, -2300.0]); //Sets the camera to a reasonable distance to view the part
    mat4.rotate(mvMatrix, 30 * Math.PI /180, [1,0,0]);
    mat4.rotate(mvMatrix, 30 * Math.PI /180, [0,1,0]);

    gl.uniformMatrix4fv(prg.uPMatrix, false, pMatrix);
    gl.uniformMatrix4fv(prg.uMVMatrix, false, mvMatrix);

    for(var i = 0; i < part.length; i++){
        gl.bindBuffer(gl.ARRAY_BUFFER, vbo[i]);
        gl.vertexAttribPointer(prg.aVertexPosition, 3, gl.FLOAT, false, 0, 0);
        gl.enableVertexAttribArray(prg.aVertexPosition);

        gl.bindBuffer(gl.ELEMENT_ARRAY_BUFFER, ibo[i]);
        gl.drawElements(gl.LINES, part[i].indices.length, gl.UNSIGNED_SHORT,0);
    }
}


/**
 * Render Loop
 */
function renderLoop() {
    utils.requestAnimFrame(renderLoop);
    drawScene();
}
/**
 * Executes the WebGL application
 * This function is invoked on the onLoad event of the webpage.
 */
function runWebGLApp(){
    //Obtains a WebGL context
    gl = utils.getGLContext('canvas-element-id');
    //Initializes the program (shaders). More about this on chapter 3!
    initProgram();
    //Initializes the buffers that we are going to use to draw the part (vertex buffer and index buffer)
    loadModel();
    //Draws the part!
    renderLoop();
}
```

Рисунок 4.25 – Створення функції для відображення моделі на сцені та запуску програми

Результат даного програмного коду представлений на рис.4.26.

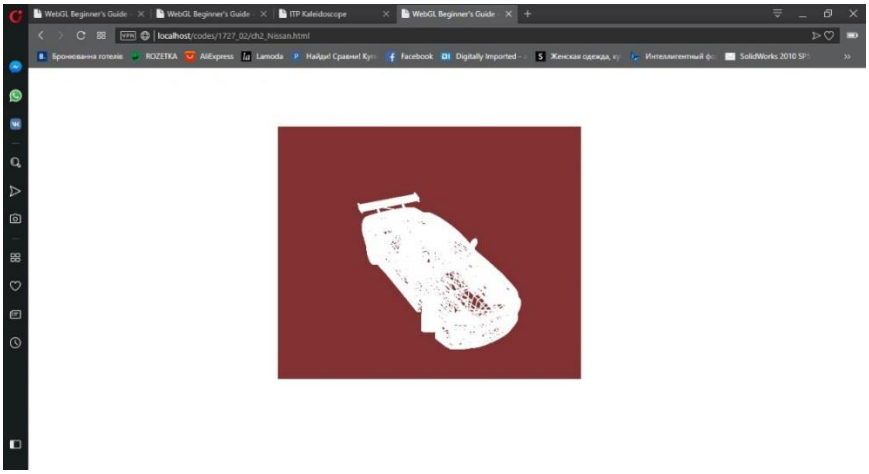

Рисунок 4.26 – Візуалізація 3d моделі на веб-сторінці

Наступним етапом є розробка алгоритму зміни кольору моделі, використовуючи кольорову палітру RGB. WebGL містить четвертий атрибут моделі RGB. Цей атрибут називається альфа канал. Розширена модель тоді відома

як модель RGBA, де A означає альфа. Альфа-канал містить значення в діапазоні від 0,0 до 1,0, як і інші три канали (червоний, зелений і синій).

Значення кольору використовується скрізь на сцені WebGL:

- об'єкти: 3D-об'єкти можуть бути пофарбовані, обираючи один колір для кожного пікселя (фрагмента) об'єкта, або шляхом вибору кольору, який буде мати об'єкт;
- джерела світла: за замовчуванням використовують білий колір для освітлення, проте користувач може мати на сцені ліхтарі, що мають інші налаштування кольору;
- сцена: фон полотна має колір, який користувач можемо змінити за допомогою виклику gl.clearColor.

Остаточний колір пікселя зазначається у шейдері фрагментів шляхом встановлення спеціальної змінної ESSL gl_FragColor. Якщо всі фрагменти в об'єкті мають однаковий колір, то можна сказати, що об'єкт має постійний колір. В іншому випадку колір об’єкту встановлюється за вершинами.

Для отримання постійного кольору потрібно зберегти налаштування кольору у формі, що передається до шейдеру фрагментів.

Механізм реалізації забарвлення моделі в колір було розроблено на прикладі, де користувач може виконати забарвлення кожної грані кубу у зв’язку з тим, що модель машини «важка» для обробки первинних даних. При цьому для кожної грані існує можливість встановлення режиму відображення кольору. На рис.4.27 – 4.30 представлені фрагменти коду для реалізації даної операції.

```
void main(void) {

  //Transformed vertex position
  vec4 vertex = uMVMatrix * vec4(aVertexPosition, 1.0);

  //we are not using lambert weighing in this exercise :-)
  float lambertTerm = 1.0;

  //reading vertex color
  vec4 Ia = uLightAmbient * uMaterialAmbient;
  vec4 Id = vec4(0.0);

  if (uUseVertexColor){
      Id = uLightDiffuse * aVertexColor * lambertTerm;
  }
  else {
      Id = uLightDiffuse * uMaterialDiffuse * lambertTerm;
  }

  vColor = Ia + Id;
  vColor.a = uAlpha;

  //Final vertex position
  gl_Position = uPMatrix * uMVMatrix * vec4(aVertexPosition, 1.0);

}
```

Рисунок 4.27 – Функція для прорахунку остаточного кольору

```
function configure(){

    gl.clearColor(0.3,0.3,0.3, 1.0);
    gl.clearDepth(1.0);

    //Enables depth testing
    gl.enable(gl.DEPTH_TEST);
    gl.depthFunc(gl.LESS);

    //Enables blending
    gl.enable(gl.BLEND);

    //Blending function for transparencies
    gl.blendFunc(gl.SRC_ALPHA, gl.ONE_MINUS_SRC_ALPHA);
    gl.blendColor(blendingColor[0], blendingColor[1], blendingColor[2], blendingAlpha);
    //Enable culling
    gl.enable(gl.CULL_FACE);

     blendingEquation    = gl.FUNC_ADD;
     blendingSource      = gl.SRC_ALPHA;
     blendingTarget      = gl.ONE_MINUS_SRC_ALPHA;

    //Creates and sets up the camera location
    camera = new Camera(CAMERA_ORBITING_TYPE);
    camera.goHome([0,0,4]);
    camera.setFocus([0.0,0.0,0.0]);
    camera.setAzimuth(50);
    camera.setElevation(-30);
    camera.hookRenderer = draw;

    //Creates and sets up the mouse and keyboard interactor
    interactor = new CameraInteractor(camera, document.getElementById('canvas-element-id'));

    //Scene Transforms
    transforms = new SceneTransforms(camera);
```

Рисунок 4.28 – Початок функції конфігурації

```
    //init transforms
    transforms.init();

    //Program
    attributeList = ["aVertexPosition",
                     "aVertexNormal",
                     "aVertexColor"];

    uniformList = [ "uPMatrix",
                    "uMVMatrix",
                    "uNMatrix",
                    "uMaterialDiffuse",
                    "uMaterialAmbient",
                    "uLightAmbient",
                    "uLightDiffuse",
                    "uLightPosition",
                    "uWireframe",
                    "uAlpha",
                    "uUseVertexColor",
                    "uUseLambert"
                    ];


    Program.load(attributeList, uniformList);

    gl.uniform3fv(Program.uLightPosition,   [0,5,20]);
    gl.uniform4fv(Program.uLightAmbient,    [1.0,1.0,1.0,1.0]);
    gl.uniform4fv(Program.uLightDiffuse,    [1.0,1.0,1.0,1.0]);
    gl.uniform1f(Program.uAlpha, 0.5);
    gl.uniform1i(Program.uUseVertexColor, useVertexColors);
    gl.uniform1i(Program.uUseLambert, false);

}
```

Рисунок 4.29 – Кінець функції конфігурації

```
$("#selSource").change(function (event) {
    blendingSource = gl[event.target.value];
    updateBlending();
});
```

Рисунок 4.30 – Реалізація вибору типу відображення кольору

Кінцевий результат даного програмного коду представлений на рис.4.31-4.32.

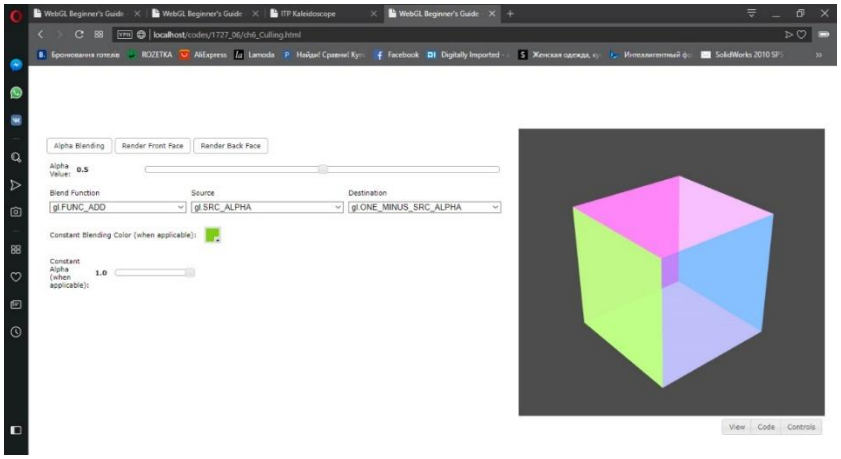

Рисунок 4.31 – Вигляд моделі за замовчуванням

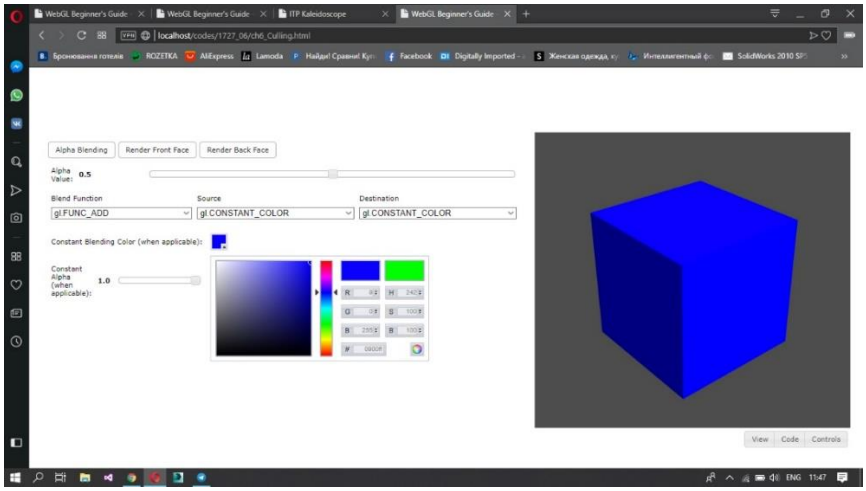

Рисунок 4.32 – Вигляд моделі після виконання маніпуляцій з кольором

Наступним етапом є реалізація механізму додавання джерел світла до сцени. В даному процесі користувачу необхідно виконувати роботу зі спеціальними уніформами, що являються доступними як для шейдеру вершин, так і для шейдеру фрагментів. Це дає велику гнучкість для розрахунків майбутнього освітлення модель, тому що користувач можемо розраховувати, як світло відходить від вершини до вершини (шейдер вершин) або на основі від фрагменту до фрагмента (шейдер фрагментів). Налаштування світла відбувається за моделлю освітлення Фонга. Метод Фонга обчислює остаточний колір в шейдері фрагментів. Для цього кожна вершина нормалі передається від шейдера вершин до шейдеру фрагментів, використовуючи змінні. Модель відображення Фонга описує, як поверхня відбиває світло у вигляді суми трьох типів відображення: навколишнє, розсіяне і дзеркальне. На рис.4.33 – 4.35 представлені фрагменти коду, що відповідають за реалізацію налаштування моделі освітлення.

```
/*-------------------------------------------------------*/
var sphereNormalsBuffer;              //VBO for Normals
/*-------------------------------------------------------*/
```

Рисунок 4.33 – Створення буферу нормалей

```
function initLights(){
    gl.uniform3f(prg.uLightDirection,   0.0, -1.0, -1.0);
    gl.uniform4fv(prg.uLightAmbient, [0.03,0.03,0.03,1.0]);
    gl.uniform4fv(prg.uLightDiffuse,  [1.0,1.0,1.0,1.0]);
    gl.uniform4fv(prg.uLightSpecular,  [1.0,1.0,1.0,1.0]);
    gl.uniform4fv(prg.uMaterialAmbient, [1.0,1.0,1.0,1.0]);
    gl.uniform4fv(prg.uMaterialDiffuse, [0.5,0.8,0.1,1.0]);
    gl.uniform4fv(prg.uMaterialSpecular,[1.0,1.0,1.0,1.0]);
    gl.uniform1f(prg.uShininess, 230.0);
}
```

Рисунок 4.34 – Функція ініціалізація джерел світла

```
function updateLightAmbientTerm(){
 var la = $('#slider-la').slider("value");
 gl.uniform4fv(prg.uLightAmbient,[la,la,la,1.0]);
 $('#slider-la-value').html(la);
}

function updateLightDiffuseTerm(){
 var ld = $('#slider-ld').slider("value");
 gl.uniform4fv(prg.uLightDiffuse,[ld,ld,ld,1.0]);
 $('#slider-ld-value').html(ld);
}

function updateLightSpecularTerm(){
 var ls = $('#slider-ls').slider("value");
 gl.uniform4fv(prg.uLightSpecular,[ls,ls,ls,1.0]);
 $('#slider-ls-value').html(ls);
}


function updateMaterialAmbientTerm(){
 var ma = $('#slider-ma').slider("value");
 gl.uniform4fv(prg.uMaterialAmbient,[ma,ma,ma,1.0]);
 $('#slider-ma-value').html(ma);
}

function updateMaterialSpecularTerm(){
 var ms = $('#slider-ms').slider("value");
 gl.uniform4fv(prg.uMaterialSpecular,[ms,ms,ms,1.0]);
 $('#slider-ms-value').html(ms);
}
```

Рисунок 4.35 - Оновлення кожного елементу моделі освітлення

Фінальний результат даного програмного коду представлений на рис.4.36 – 4.37.

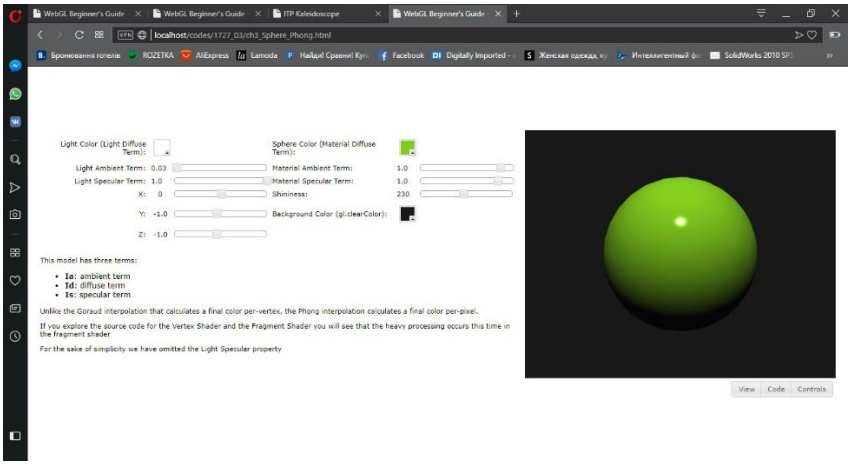

Рисунок 4.36 - Вигляд моделі за замовчуванням

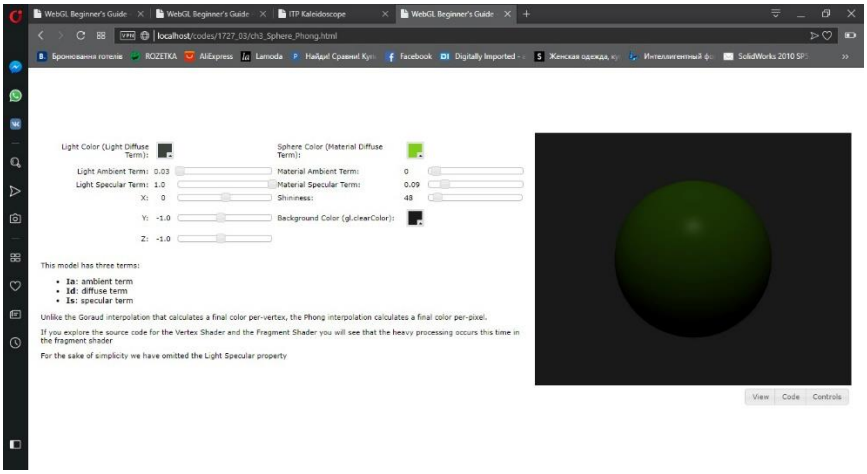

Рисунок 4.37 – Вигляд моделі після налаштування світла

Останнім етапом є розробка механізму роботи з камерами. Дана операція реалізується на основі проекційного перетворення. Ця операція визначає, яким саме чином модель буде відображена, спираючись на шість площин (ближня, дальня, верхня, нижня, права і ліва). У парадигмі технології WebGL такі операції як поворот, переміщення та масштабування моделі описуються матрицею, розмірність якої 4*4. Для опису камери також буде достатньо використання однієї із такої матриці. На рис.4.38 – 4.40 представлені фрагменти коду, що відповідають за реалізацію роботи з камерою на сцені WebGL.

```
var camera = null;
var interactor = null;

function configure(){

    gl.clearColor(0.3,0.3,0.3, 1.0);
    gl.clearDepth(100.0);
    gl.enable(gl.DEPTH_TEST);
    gl.depthFunc(gl.LEQUAL);

    //Creates and sets up the camera location
    camera = new Camera(CAMERA_TRACKING_TYPE);
    camera.goHome([0,2,50]);
    camera.hookRenderer = draw;
    camera.hookGUIUpdate = updateSliders;

    //Creates and sets up the mouse and keyboard interactor
    var canvas = document.getElementById('canvas-element-id');
    interactor = new CameraInteractor(camera, canvas);

    //Update gui with camera settings
    initGUIWithCameraSettings();

    //init transforms
    initTransforms();
}
```

Рисунок 4.38 – Ініціалізація камери

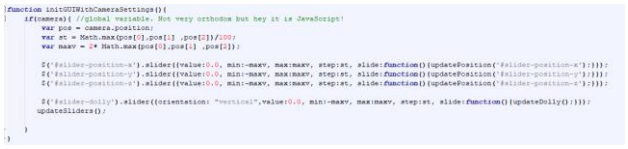

Рисунок 4.39 – Функція налаштування камери

```
function updatePosition(selector){
    var pos = $(selector).slider("value");
    var p = camera.position;
    $(selector+'-value').html(pos);

    if(selector == '#slider-position-x'){
        p[0] = pos;
    }
    else if(selector == '#slider-position-y'){
        p[1] = pos;
    }
    else if (selector == '#slider-position-z'){
        p[2] = pos;
    }
    camera.setPosition(p);
    app.refresh();
}
```

Рисунок 4.40 – Функція зміни позиції

Фінальний результат даного програмного коду представлений на рис.4.41 – 4.42.

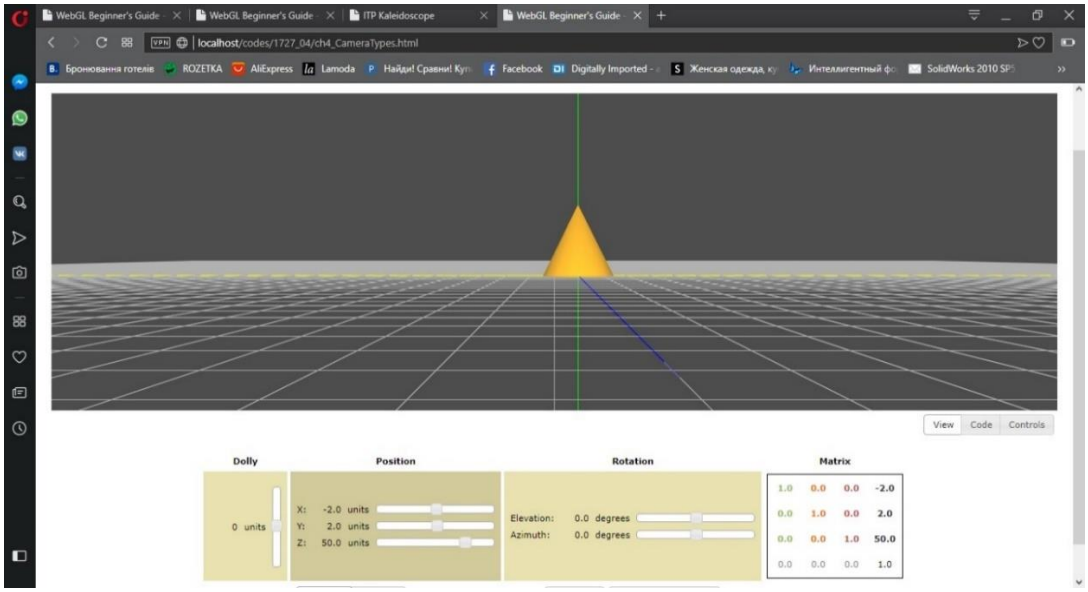

Рисунок 4.41 - - Вигляд сцени за замовчуванням

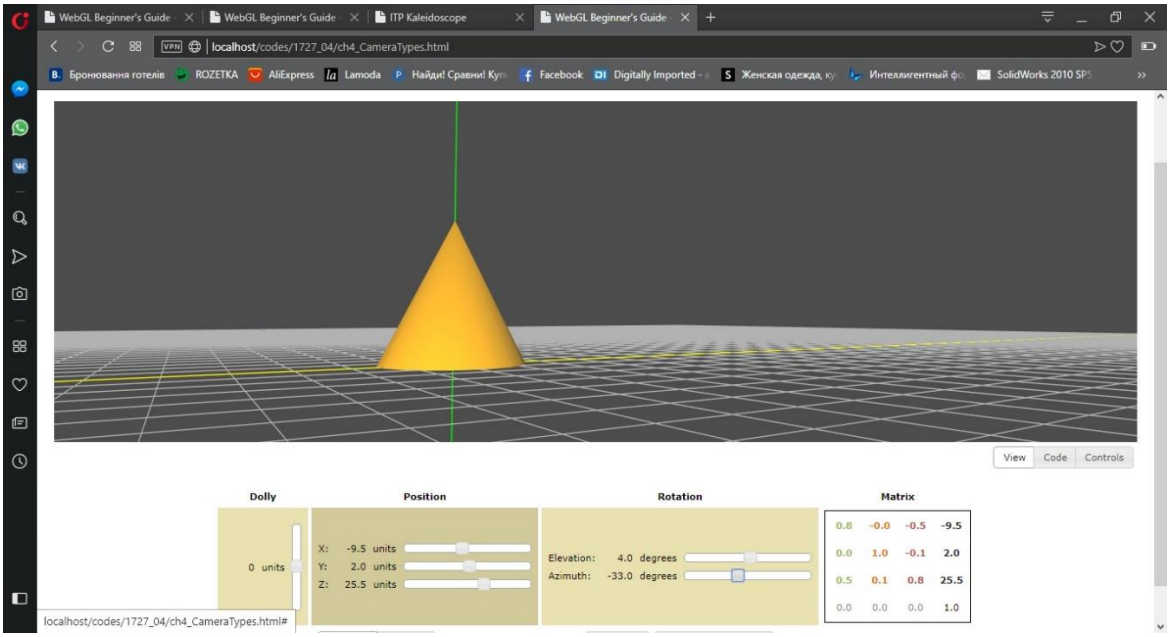

Рисунок 4.42 - - Вигляд сцени після налаштування камери

4.2.4 Демонстрація роботи користувача з веб-ресурсом

На основі розробки базових маніпуляцій, що були продемонстровані у попередньому розділі, командою проекту був розроблений веб-ресурс для перегляду 3d моделей з функціоналом, що був зазначений у технічному завданні (Додаток А).

Файли коду HTML та CSS представлені у Додатку С.

На рис.4.43 – 4.48 продемонстрована робота користувача з веб-ресурсом.

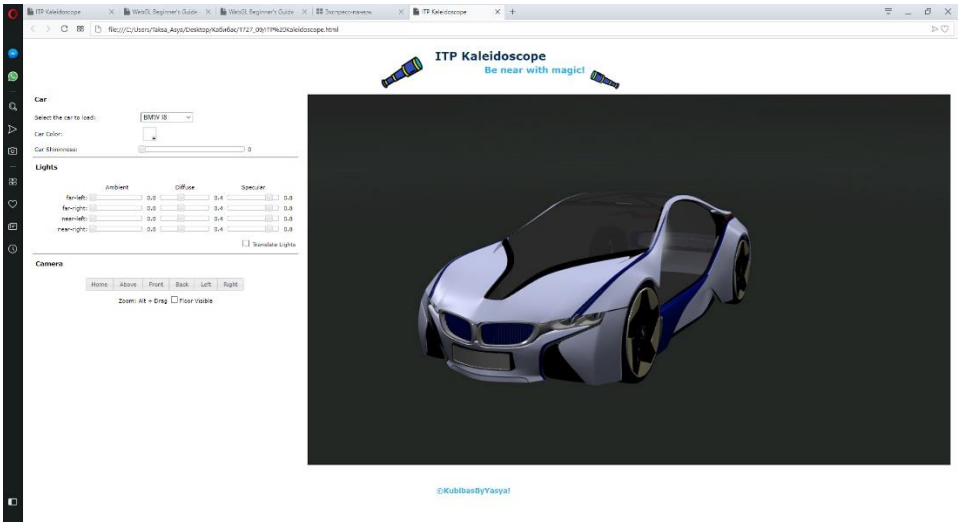

Рисунок 4.43 – Вигляд сцени на початку роботи

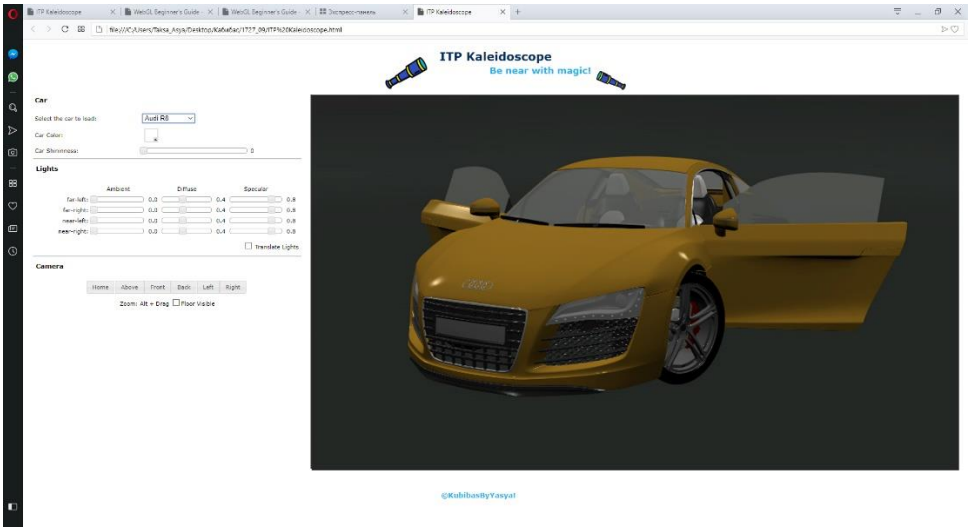

Рисунок 4.44 – Виконання операції зміни моделі

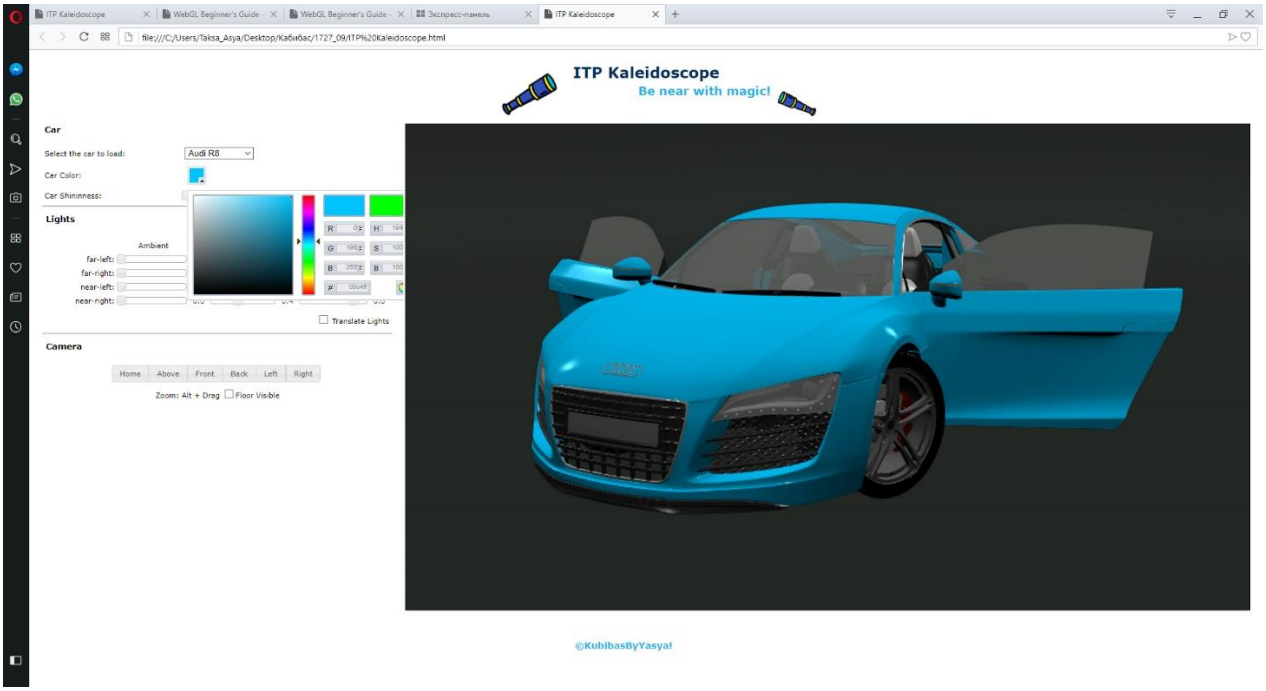

Рисунок 4.45 - Виконання операції зміни кольору моделі

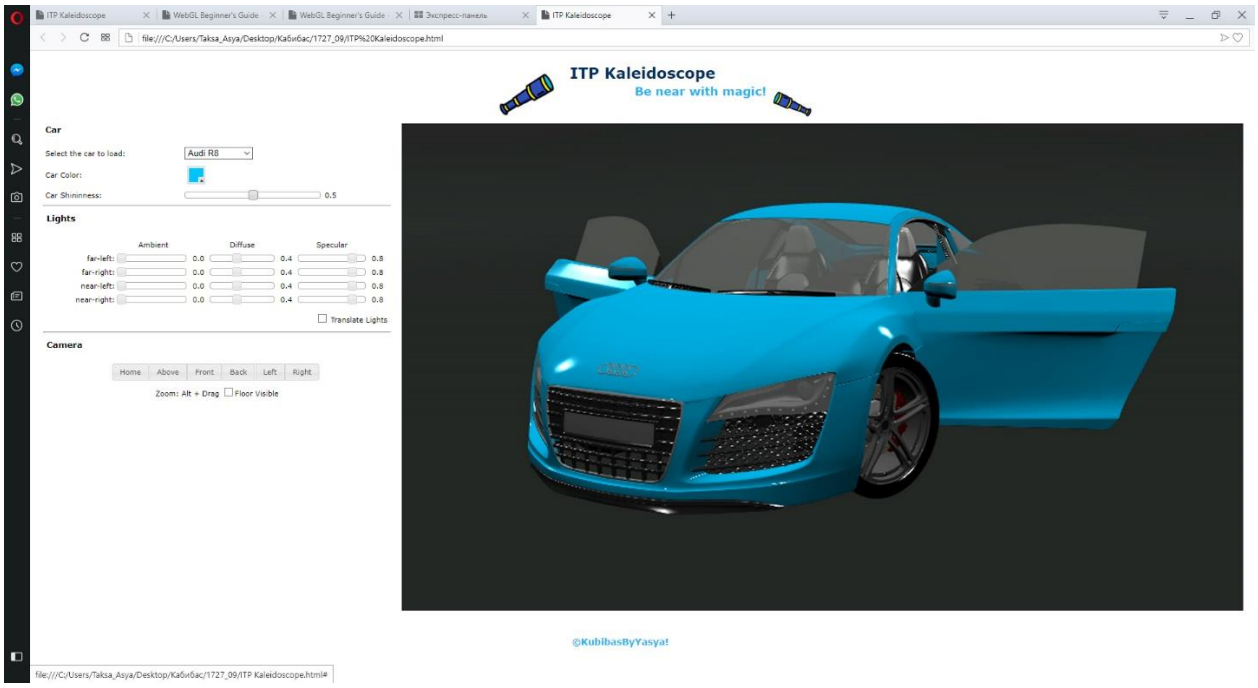

Рисунок 4.46 - Виконання операції зміни блиску моделі

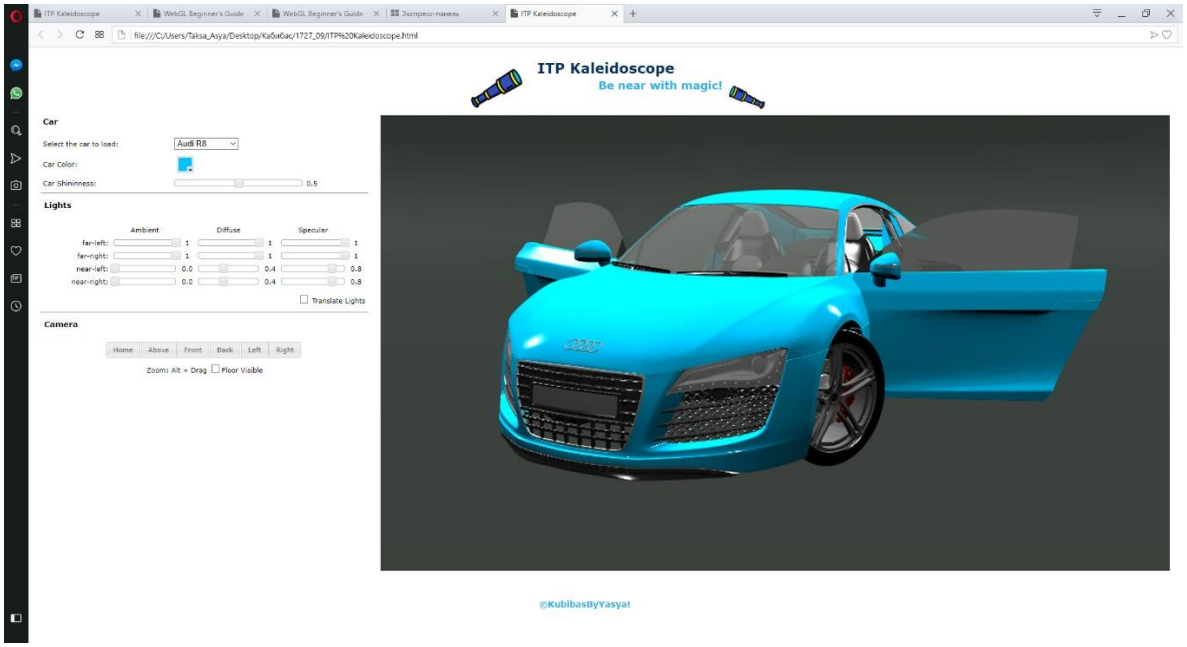

Рисунок 4.47 - Виконання операції зміни налаштувань джерел світла

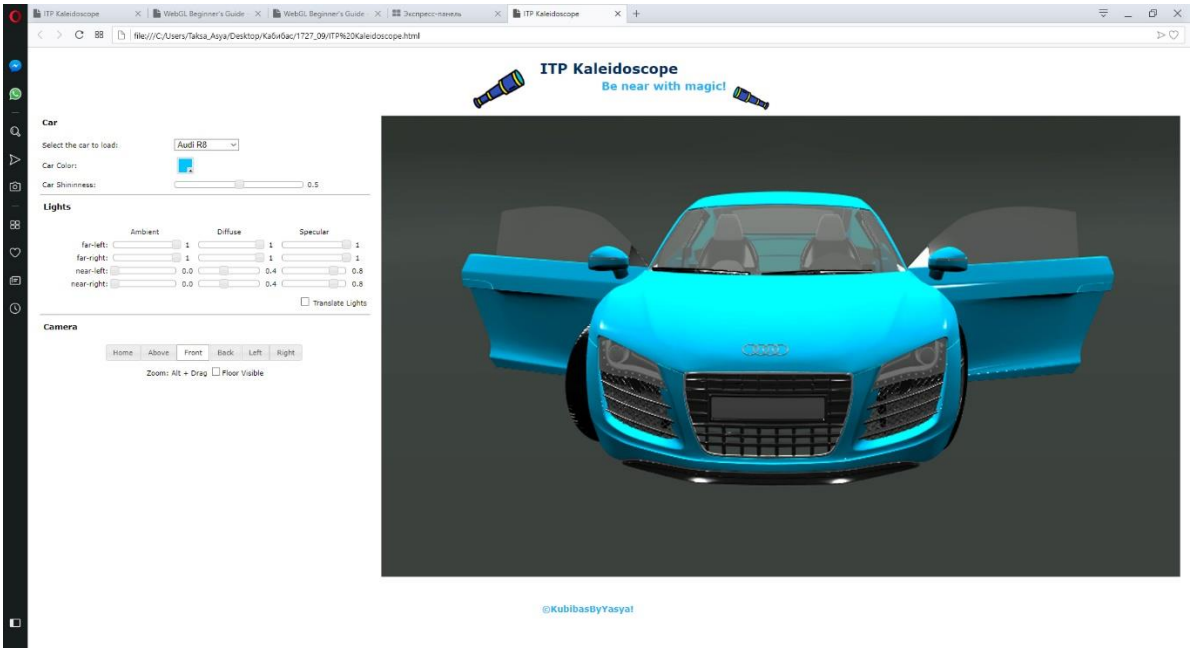

Рисунок 4.48 - Виконання операції зміни проекційного виду

ВИСНОВКИ

У дипломному проекті був розроблений веб-ресурс для перегляду 3d моделей з використанням технології WebGL. Даний продукт був розроблений на замовлення секції ІТП, кафедри Комп'ютерні науки Сумського державного університету. Веб-ресурс «ITP Kaleidoscope» відповідає всім вимогам замовника:

- зміна кольору моделі;
- налаштування чотирьох джерел світла;
- вибір бажаного проекційного виду для перегляду моделі;
- масштабування сцени.

Цей проект несе в собі соціально-економічну цінність, оскільки він буде мати попит перш за все у вищих навчальних закладах, які у своєму складі мають кафедри/спеціальності з нахилом на 3d моделювання, інженерію. Адже використовуючи даний продукт, викладачі зможуть зробити навчально-виховний процес більш цікавим, інформативним та ефективним у засвоєнні знань з боку слухачів/студентів.

Під час виконання проекту було детально досліджено питання актуальності проблеми, ретельно вивчено переваги та недоліки аналогових ресурсів, визначено технології та мови програмування, що дозволять реалізувати даний продукт. Разом з замовником проекту було сформовано технічне завдання, яке містить детальні вимоги до розробленого веб-ресурсу.

Командою проекту було виконано планування робіт, що дозволило визначити перелік ризиків та сценарій реагування них, сформовано календарний план для відслідковування дат завершення кожного етапу роботи.

Структура веб-сторінки була розроблена шляхом прототипування за допомогою програмного продукту Adobe Photoshopю Даний етап дозволив

команді проекту узгодити з замовником розміщенні основних блоків та встановити загальну кольорову гамму.

Засобами мов HTML та CSS було виконано розмітку та подальше форматування веб-сторінки у відповідності до фінального вигляду прототипу.

За допомогою функцій та методів мови програмування JavaScript до пульту керування були додані функції, що відповідають за коректну реакцію на дії користувача.

Використовуючи атрибути та методи технології WebGL, команда проекту мала змогу детально ознайомитись з методологією створення базових маніпуляцій з моделлю:

- завантаження моделі;
- зміна кольору моделі;
- налаштування освітлення;
- поворот сцени.

Після того, як веб-ресурс був остаточно розроблений, було проведено його тестування. За результатом тесту було прийнято таке рішення, що існують випадки відсутності візуалізованої моделі на сцені. Для вирішення цієї проблеми командою проекту були сформульовані вимоги до веб-браузерів користувачів, що наведені у Додаток А.

Отже, після виконання всіх етапів проекту, було розроблено веб-ресурс, що покращить показник ефективності засвоєння знань на кафедрах/спеціальностях з нахилом інженерія/3d моделювання. Даний проект отримав акт впровадження від секції інформаційні технології проектування кафедри комп’ютерні науки.

СПИСОК ЛІТЕРАТУРИ

1. 3d визуализация [Електронний ресурс] – режим доступу: https://bit.ly/2YHFVSN

2. 3d графика и ее применение [Електронний ресурс] – режим доступу: https://bit.ly/2IbPMKS

3. Adobe Flash vs HTML5 vs Silverlight. Возможное мультимедийное будущее Интернета [Електронний ресурс] – режим доступу: https://bit.ly/2FYBRWP

4. Blend4Web [Електронний ресурс] – режим доступу: https://bit.ly/2YJqzNr

5. HTML [Електронний ресурс] – режим доступу: https://bit.ly/2ONQJu4

6. HTML сайт. Преимущества и недостатки HTML сайтов [Електронний ресурс] – режим доступу: https://bit.ly/2WGyDN7

7. Microsoft Silverlight. Что это за программа и нужна ли она? [Електронний ресурс] – режим доступу: https://bit.ly/2HXmtwd

8. Modelo.io [Електронний ресурс] – режим доступу: https://bit.ly/2CWZ5L3

9. Sketchfab [Електронний ресурс] – режим доступу: https://bit.ly/2FVvxPU

10. Verge3D [Електронний ресурс] – режим доступу: https://bit.ly/2CO660L

11. Бочков А.Л., Меженин А.В. Графика и мультимедиа для Web. Учебно-методическое пособие.-СПб.: СПбГИТМО(ТУ), 2002.-44с.

12. Зачем нужно веб-программирование? [Електронний ресурс] – режим доступу: https://bit.ly/2WLjJ8p

13. Мультимедийная платформа Adobe Flash [Електронний ресурс] – режим доступу: https://bit.ly/2YO59Pd

14. Начало работы с WebGL [Електронний ресурс] – режим доступу: https://mzl.la/2CSw8A3

15. Преимущества использования HTML в создании Web-страниц [Електронний ресурс] – режим доступу: https://bit.ly/2HYphZJ

16. Создание веб-сервера для работы с 3d моделями [Електронний ресурс] – режим доступу: https://bit.ly/2OIBx1b

17. Трехмерна графика в вебе [Електронний ресурс] – режим доступу: https://bit.ly/2FRRw8X

18. Учебник HTML и CSS от Трепачева Дмитрия [Електронний ресурс] – режим доступу: https://bit.ly/2YMTFLV

19. Что такое CSS. Преимущества CSS. Версии CSS и их разработчики [Електронний ресурс] – режим доступу: https://bit.ly/2D4Hpxl

20. Что такое CSS, преимущества и недостатки [Електронний ресурс] – режим доступу: https://bit.ly/2IgKT3d

21. Что такое JavaScript? [Електронний ресурс] – режим доступу: https://mzl.la/2LXeiSY

22. Что такое JavaScript? Презентация JS для начинающих [Електронний ресурс] – режим доступу: https://bit.ly/2TTxaBh

Додаток А

Технічне завдання

Назва веб-ресурсу «ITP Kaleidoscope».
Область застосування веб-ресурсу – веб-ресурс призначений для наочної демонстрації бази 3d моделей, створених студентами та викладачами секції Інформаційні технології проектування.
Об'єкт, у якому використовують програму – секція Інформаційних технологій проектування.

1 Основи для розробки

Розробка виконується на основі завдання, виданого викладачами секції Інформаційних технологій проектування.

2 Призначення розробки

Розробка повинна надати можливість перегляду широкої бази 3d моделей, що були створенні студентами та викладачами секції, та бути універсальною для всіх можливих розширень 3d моделей.
Назва організації: Ясінська Тетяна, ІТ-52, кафедра Комп'ютерних Наук.
Тема проекту: «Розробка веб-ресурсу для перегляду 3d моделей з використанням технології WebGL»

3 Вимоги до веб-ресурсу

Веб-ресурс повинен бути реалізований у вигляді одинарної веб-сторінки, клієнтська частина якої є панель, реалізована елементами керування(кнопки, check-box, list та інші) та забезпечує виконання функціональних можливостей, визначених у пункті 3.3.

3.1 Вимоги до веб-ресурсу

Веб-ресурс розробляється ітеративно із врахуванням принципів та технологій уніфікованого процесу розроблення веб-ресурсу(сайту). Каркас веб-ресурсу має бути реалізований мовою HTML з підключенням до CSS. Функції, перелічені в пункті 3.3, мають бути написані мовою JavaScript з використанням технології WebGL.

3.2 Вимоги до програмного забезпечення

Для коректної роботи з веб-ресурсом необхідно інсталювати веб-сервер(наприклад, lighttpd) для швидкої реалізації запиту від клієнта. Для можливості перегляду всіх можливостей технології WebGL програмне забезпечення клієнтської частини повинне задовольняти наступним вимогам стосовно веб-браузера: Mozilla Firefox – починаючи з версії 4.0, Google Chrome –

починаючи з версії 9.0, Opera – починаючи з версії 12.0, Safari – починаючи з версії 8.0, Interten Explorer – починаючи з версії 11.0.

3.3 Вимоги до функціональних характеристик

Веб-ресурс повинен забезпечувати виконання наступних функції:

- зміна кольору моделі;
- налаштування чотирьох джерел світла за моделлю освітлення Фонга;
- вибір бажаного проекційного виду для перегляду моделі;
- масштабування сцени.

4 Перелік програмної документації

- Опис проекту продукту із використанням UML-діаграм;
- Макет інтерфейсу веб-ресурсу;
- Програмний код розробки веб-ресурсу;
- Технічне завдання;
- Інструкція користувача;

5 Порядок виконання робіт і етапи розробки

Стадії та етапи розробки повинні складатися з наступних пунктів:

- Оформлення завдання для дипломної роботи;
- Планування роботи. Розроблення ТЗ ,побудова мережевого графіку та діаграми Ганта;
- Розроблення структури веб-ресурсу;
- Реалізація структури у вигляді веб-сторінки;
- Розроблення функціоналу завантаження моделі та базових маніпуляцій з нею;
- Провести тестування веб-ресурсу
- Розроблення інструкції користувача;
- Оформлення пояснювальної записки про виконання дипломної роботи;
- Здача пояснювальної записки до дипломної роботи та розробленого веб-ресурсу;
- Презентація роботи та її захист.

6 Порядок контролю та приймання

Контроль коректності функціонування та придатності веб-ресурсу здійснюється замовником (секцією Інформаційні технології проектування) на основі наданої пояснювальної записки до дипломної роботи та програмних файлів. Контроль ходу виконання проекту здійснюється на основі календарного плану виконання дипломної роботи:

- Перевірка завдання дипломної роботи.
- Перевірка ТЗ, мережевого графіка та діаграми Ганта.

- Перевірка структури веб-ресурсу.
- Перевірка веб-сторінки для веб-ресурсу.
- Перевірка наявності функціоналу для завантаження моделі та маніпуляцій з нею.
- Перевірка коректності тестування.
- Перевірка інструкції користувача.
- Здача ПЗ.
- Презентація.

Додаток Б

Планування робіт

Мета проекту: розробити web-сервер для перегляду 3d моделей з використанням технології WebGL, де користувач з легкістю зможе обрати бажану модель для перегляду, буде мати можливість змінити колір моделі, використовуючи кольоровий спектр. Також за бажанням користувач зможе змінити один із чотирьох джерел світла на сцені, використовуючи модель відображення Фонга. Результати деталізації методом SMART розміщенні у табл. Б.1.

Таблиця Б.1 – Деталізація мети методом SMART

Specific (конкретна)	Розробити веб-ресурс для перегляду 3d моделей з використанням технології WebGL.
Measurable (вимірювання)	Оскільки даний проект не є комерційним, то результатом ого роботи є оцінка замовника.
Achievable (досяжна, узгоджена)	Ціль даного проекту вважається досяжною, оскільки розробник володіє необхідними навичками у створенні веб-сторінок засобами мов html, css, javascript та ознайомлений з необхідною кількістю методів, функцій та бібліотек технології WebGL. Мета була узгоджена з вимогами та потребами замовника.

Продовження таблиці Б.1

Relevant (реалістична)	Для реалізації продукту проекту є всі необхідні технічні та програмні засоби(веб-сервер LightTPD, універсальний інтерпретатор Notepad++, графічний редактор для тривимірної графіки Blender), доступ до мережі Інтернет. Розробник досить кваліфікований для виконання поставлених задач.
Time-framed (обмежена в часі)	Веб-ресурс розроблюється з обмеженням у часі на основі сформованого календарного плану та матриці відповідальності.

Планування змісту структури робіт. Основним інструментом для планування змісту структури робіт служить WBS(Work Break Structure) - представлення проекту, виконане у вигляді ієрархічної структури робіт, що досягається за допомогою послідовної декомпозиції. Інструмент спрямований на детальне планування, оцінку вартості, визначення та розподіл персональної відповідальності виконавців та інші - тобто, на основні роботи і результати, що визначають зміст проекту. Виконуємо декомпозицію робіт для проекту. Діаграма WBS представлена на рис.Б.1.

Планування структури організації, для впровадження готового проекту (OBS). Після того, як була побудована WBS структура проекту наступним етапом є розроблення OBS (Organization structure) - склад, підпорядкованість, взаємодія і розподіл робіт по підрозділах і органам управління, між якими встановлюються певні відносини з приводу реалізації владних повноважень, потоків команд і інформації. Організаційна структура проекту стосується тільки внутрішньої

організаційної структури проекту і не стосується відносин проектних груп чи учасників з батьківськими організаціями. Список виконавців, що функціонують в проекті представлений в табл. Б.2. Організаційна структура проекту зображена на рис. Б.2.

Таблиця Б.2 – Виконавці проекту

Роль	Ім’я	Проектна роль
Розробник	Ясінська Т.А.	Виконує розробку основного функціоналу проекту.
Менеджер проекту	Федотова Н.А.	Відповідає за виконання термінів, виконує збір та аналіз даних.
Консультант проекту	Баранова І.В.	Формує завдання на розробку проекту
Тестувальник	Ващенко С.М.	Відповідає за тестування функціоналу проекту

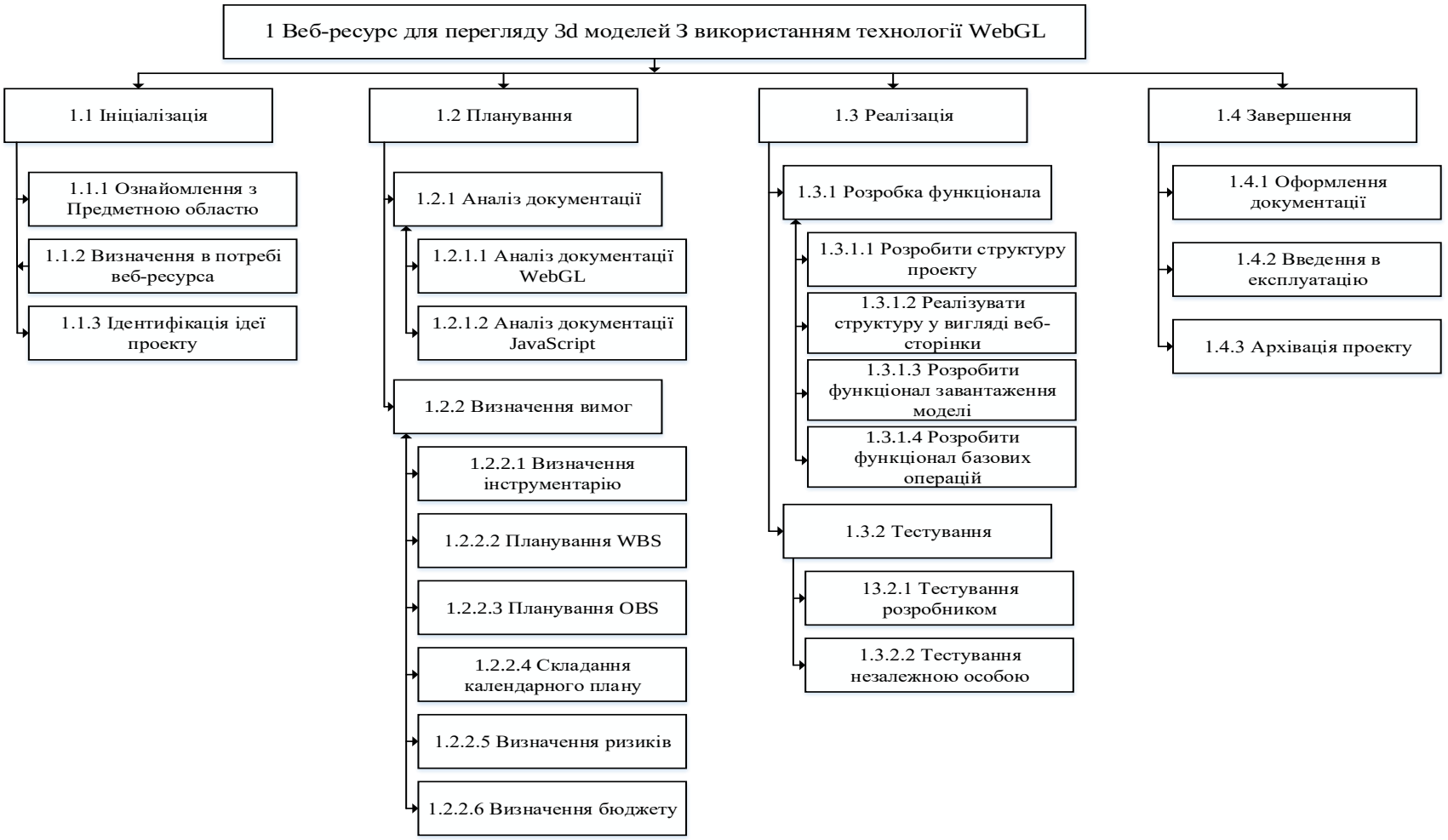

Рисунок Б.1 − WBS. Структура робіт проекту

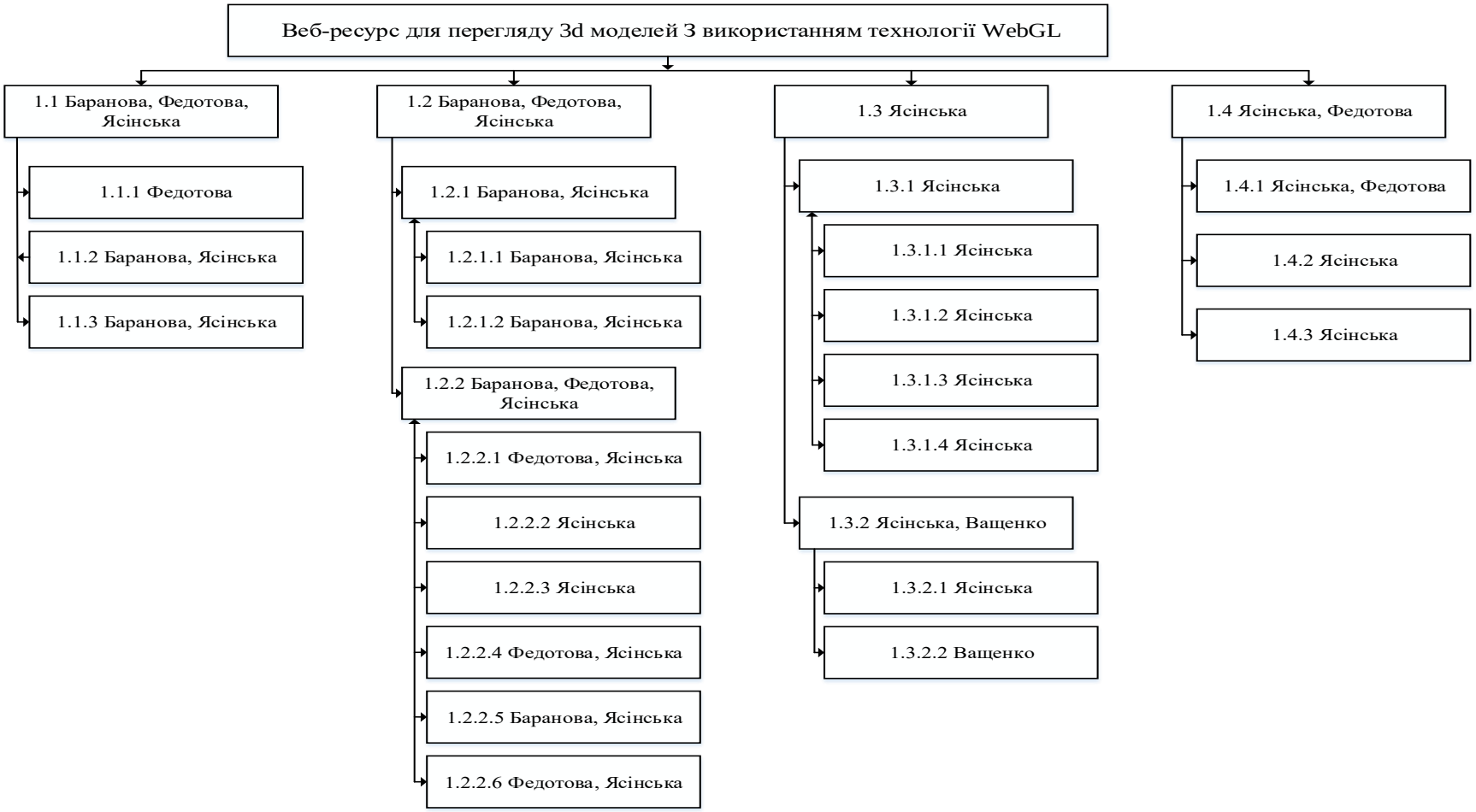

Рисунок Б.2 – OBS. Організаційна структура проекту

Матриця відповідальності. Модель RACI - засіб для виявлення активностей і розподілу їх по ролям і зонам відповідальності. Використання матриці RACI дозволяє уникнути нерозуміння в тому, кого необхідно залучити до проекту, а також хто і що повинен робити.

RACI - скорочення від основних ролей учасників проекту:

- Responsible(Виконавець) - Той кому призначена ця роль відповідає за виконання роботи та досягнення цілей проекту.
- Accountable (Відповідальний): Виконавець цієї ролі відповідає за якість і результати процесу.
- Consulted (Консультант, Експерт): Той кому призначена ця роль залучається, як носій унікальних знань або інформації.
- Informed (Інформатор): Це особа, якого необхідно тримати в курсі про хід і результати процесу, найчастіше в односторонньому порядку, тому що у нього немає повноважень безпосередньо впливати на хід проекту.

На основі розроблених WBS та OBS структур проекту була побудована матриця відповідальності проекту, яка представлена в таблиці Б.3.

Таблиця Б.3 – Матриця відповідальності

WBS\OBS	Ясінська	Федотова	Баранова	Ващенко
Ознайомлення з предметною областю		+		
Визначення в потребі веб-ресурса	+		+	
Ідентифікація ідеї проекту	+		+	
Аналіз документації WebGL	+		+	
Аналіз документації JavaScript	+		+	
Визначення інструментарію	+	+		
Планування WBS	+			

Продовження таблиці Б.3

WBS\OBS	Ясінська	Федотова	Баранова	Ващенко
Планування OBS	+			
Складання календарного плану	+	+		
Визначення ризиків	+		+	
Визначення бюджету	+	+		
Розробка структури веб-ресурсу	+			
Реалізація структури у вигляді веб-сторінки	+			
Розробка функціоналу завантаження моделі	+			
Розробка функціоналу базових операцій	+			
Тестування розробником	+			
Тестування незалежною особою				+
Оформлення документації	+	+		
Введення в експлуатацію	+			
Архівація проекту	+			

PDM мережа. Мережева модель – це графічне представлення проекту. Вона дозволяє знайти мінімальні строки завершення проекту та окремих робіт\процесів, а також визначити множину критичних робіт., збільшення тривалості виконання будь-якої з яких призводить до збільшення часу виконання всього проекту. PDM мережа для даного проекту була розроблена за допомогою програмного забезпечення GanttProject. Мережа представлена на рис. Б.3-Б.5.

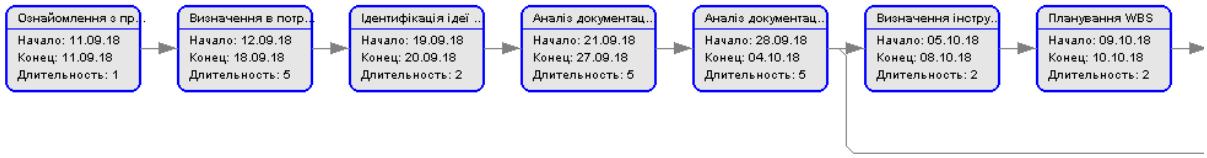

Рисунок Б.3 - PDM – мережа проекту

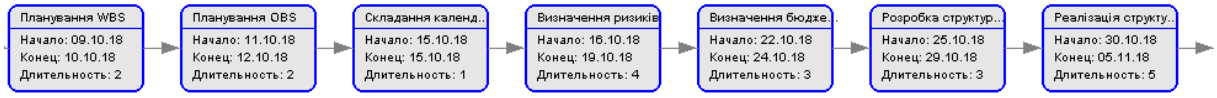

Рисунок Б.4 - Продовження PDM – мережі проекту

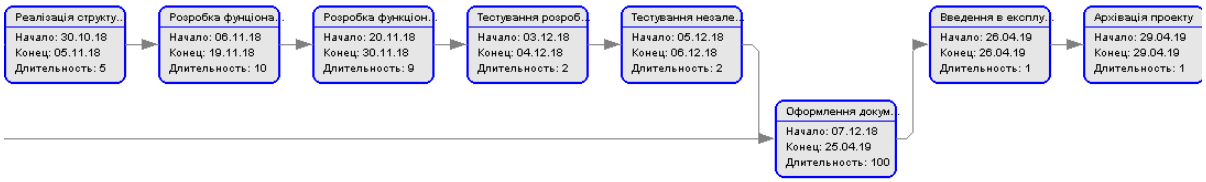

Рисунок Б.5 - Продовження PDM – мережі проекту

Діаграма Ганта. На відміну від PDM – мережі, що пропонує мережеву модель, управління проектами з діаграмами Ганта засноване на форматі гістограм. Це допомагає відслідковувати відсоток робіт, виконаних по кожному завданню. Керівникам проектів дуже важливо правильно розподілити завдання і бути впевненими в тому, що проект буде завершений вчасно. Основна увага діаграм Ганта зосереджено на процентному завершенні кожного завдання. Крім того, діаграми Ганта краще для проектів з невеликою кількістю взаємопов'язаних завдань. Завдяки засобам програмного продукту GanttProject була розроблена діаграма Ганта, яка у вигляді гістограми відображає тривалість кожного процесу, що був

визначений на етапі формування WBS. Діаграма Ганта представлена на рис Б.6. Список робіт для побудови діаграми Ганта представлений на рис. Б.7.

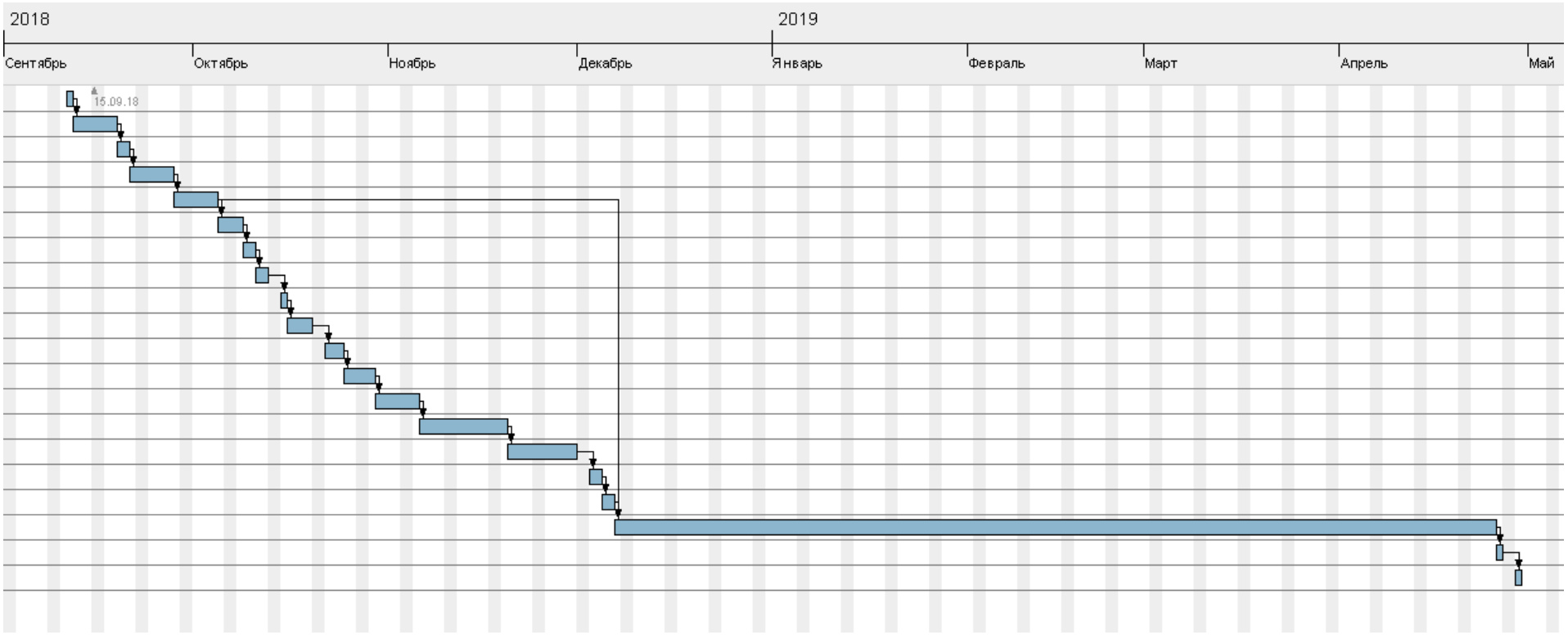

Рисунок Б.6 – Діаграма Ганта

Название	Дата начала	Дата окончания
Ознайомлення з предметною областю	11.09.18	11.09.18
Визначення в потребі веб-ресурса	12.09.18	18.09.18
Ідентифікація ідеї проекту	19.09.18	20.09.18
Аналіз документації WebGL	21.09.18	27.09.18
Аналіз документації JavaScript	28.09.18	04.10.18
Визначення інструментарія	05.10.18	08.10.18
Планування WBS	09.10.18	10.10.18
Планування OBS	11.10.18	12.10.18
Складання календарного плану	15.10.18	15.10.18
Визначення ризиків	16.10.18	19.10.18
Визначення бюджету	22.10.18	24.10.18
Розробка структури веб-ресурсу	25.10.18	29.10.18
Реалізація структури у вигляді веб-сторінки	30.10.18	05.11.18
Розробка фунціоналу завантаження моделі	06.11.18	19.11.18
Розробка функціоналу базових операцій	20.11.18	30.11.18
Тестування розробником	03.12.18	04.12.18
Тестування незалежною особою	05.12.18	06.12.18
Оформлення документації	07.12.18	25.04.19
Введення в експлуатацію	26.04.19	26.04.19
Архівація проекту	29.04.19	29.04.19

Рисунок Б.7 – Список робіт для діаграми Ганта

Управління ризиками. Ризик – ймовірнісна подія, яка може позитивно чи негативно вплинути на проект. Ідентифікація ризиків - визначення ризиків, здатних вплинути на проект, і документування їх характеристик.

Ідентифікація ризиків визначає, які ризики здатні вплинути на проект, і документує характеристики цих ризиків. Ідентифікація ризиків не буде ефективною, якщо вона не буде проводитися регулярно протягом реалізації проекту.

Класифікація ризиків:

1. За імовірністю виникнення:

- слабо ймовірнісні;
- мало ймовірнісні;
- імовірні;
- досить імовірні;
- майже імовірні.

2. За величиною втрат:
   - мінімальна;
   - низька;
   - середня;
   - висока;
   - максимальна.

На основі цих даних була проведена класифікація ризиків для даного проекту, що наведена в табл. Б.4.

Таблиця Б.4 – Класифікація рисків

№	Назва ризику	Ймовірність	Величина втрат
1	Некоректно складене ТЗ	2	4
2	Недотримання календарного плану	1	3
3	Некоректна робота програмного забезпечення	4	4
4	Некоректна робота апаратного забезпечення	4	4
5	Хвороба розробника	2	2
6	Некоректне тестування	2	1
7	Пошкодження файлу	4	5

Використовуючи дану класифікацію, була побудована матриця ризиків, що представлена на рис. Б.8.

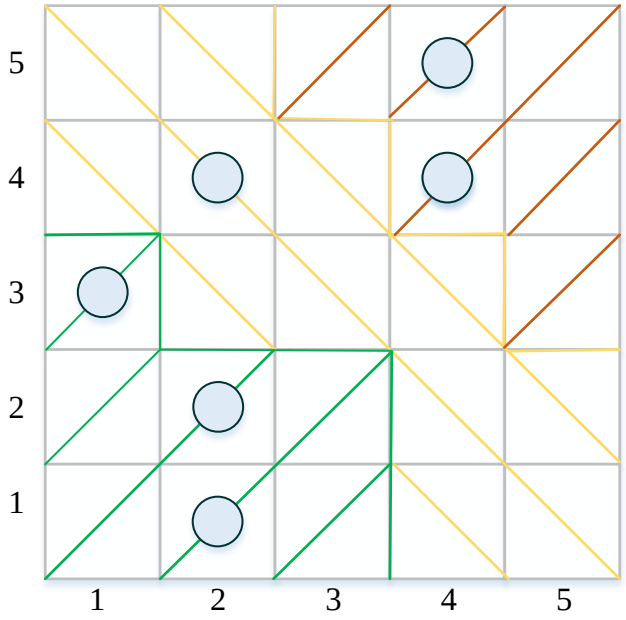

Рисунок Б.8 – Матриця ризиків

Далі було визначено рівні ризиків та ступінь їх дії.

Рівні можуть бути:

- допустимі 1<R<4;
- оправдані 5<R<10;
- недопустимі 11<R<25.

Ступінь дії ризиків:

- ті, що можна проігнорувати 1<R<4;
- незначні 5<R<8;
- помірні 9<R<10;

- істотні 11<R<16;
- критичні 17<R<25.

На основі цих даних була виконана оцінка ступенів та рівнів для кожного ризику в проекті. Результати роботи представлені в табл. Б.5.

Таблиця Б.5 – Визначення ступенів та рівнів ризиків

№	Назва ризику	Ймовірність ризику	Величина втрат	Рівень ризику	Ступінь дії
1	Некоректно складене ТЗ	2	1	Допустимий	Проігнорувати
2	Недотримання календарного плану	5	9	Оправданий	Помірний
3	Некоректна робота програмного забезпечення	2	15	Допустимий	Істотний
4	Некоректна робота апаратного забезпечення	2	15	Допустимий	Істотний
5	Хвороба розробника	2	5	Допустимий	Незначний
6	Некоректне тестування	2	3	Виправданий	Проігнорувати

Продовження таблиці Б.6.

№	Назва ризику	Ймовірність ризику	Величина втрат	Рівень ризику	Ступінь дії
7	Пошкодження файлу	20	25	Недопустимий	Критичний

Після виконання прогнозування виникнення ризиків та їх ступінь впливу на результат реалізації проекту, були розроблені варіанти запобігання та реакції на кожний із них. Результати даного етапу представлені в табл. Б.6.

Таблиця Б.6 - Варіанти запобігання та реакції на ризики

Ризики проекту	План запобігання ризику	План реакції на ризик
Некоректно складене ТЗ	Замовник повинен скласти детальне ТЗ, дотримуючись затвердженого плану: словник термінів, мета проекту, усі види вимог, терміни. Замовник та розробник повинні обговорити та затвердити його.	Уважно та чітко окреслити те, що було виконано невірно (після розмови із замовником) та зробити правки.
Недотримання календарного плану	1. Створення плану реалізації проекту на основі ретельного аналізу списку всіх робіт. 2. Затвердження зазначених термінів із замовником. 3. Командна робота над планом термінів виконання. (Можливість внесення правок перед затвердженням усіма членами команди).	1. Обговорення варіантів внесення правок до термінів реалізації із керівником та замовником. 2. Домовитися про умови зміни термінів із замовником. Якщо це недопустимо, тоді переорганізувати роботи таким чином, щоб в результаті терміни виконувалися.

Продовження таблиці Б.6.

Ризики проекту	План запобігання ризику	План реакції на ризик
Некоректна робота програмного забезпечення	1. Встановлення ліцензійного програмного забезпечення з перевірених джерел перед початком роботи. 2. Забезпечити наявність антивірусного програмного забезпечення. 3. Створення резервних копій розробленого веб-сервера та всіх інших допоміжних елементів для функціонала на кожній стадії розробки та зберігання їх на зовнішніх або віртуальних сховищах інформації.	Перезапуск або переустановлення програми, яка дала збій, відкат до останньої
Некоректна робота апаратного забезпечення	1. Раз на 4-6 місяців виконувати перевірку працездатності апаратного забезпечення 2. Робити резервні копії проекту на зовнішніх носіях або хмарних технологіях для запобігання втрати інформації.	Виконати ремонт апаратного забезпечення, якщо терміни виконання завдань не дозволяють чекати, знайти тимчасову заміну, завантажити останню резервну копію проекту та продовжувати роботу над проектом.
Хвороба розробника	Виконувати певну частину роботи в команді для того, щоб члени проекту змогли замінити один одного при необхідності. При плануванні термінів залишити декілька резервних днів для таких випадків.	Передати повноваження робітника іншому члену команди, якщо цього вимагають терміни виконання.
Некоректне тестування	Виконати пошук кваліфікованого тестувальника в даній предметній області.	Передати проект на додаткове тестування кваліфікованому спеціалісту.

Продовження таблиці Б.6.

Ризики проекту	План запобігання ризику	План реакції на ризик
Пошкодження файлу	1. Під час процесу генерації моделі в файл необхідно використовувати перевірене ліцензійне програмне забезпечення. 2. Виконати резервне копіювання на зовнішнє або віртуальне сховище даних.	Перезапуск або переустановлення програми, яка дала збій, відкат до останньої збереженої версії проекту, якщо останні дані були втрачені.

Додаток С

Файли коду реалізації

Файл коду style.css

```
body
{
font-family:Verdana,Arial,sans-serif;
font-weight:bold;
margin:20px;
font-size:11px;
}

h1 {font-size:23px; color:#072f62; text-align:center;}
h2   {font-size:18px;   color:#22b3f6;   position:absolute;   right:750px;
top:36px;}
/*h3 {font-size:15px;}*/
h4{font-size:14;color:#22b3f6;position:absolute;right:900px;bottom:36px;}
a {font-size:11px;}

/*div{border:1px dashed rgb(250,0,0);} /*debug*/

#telescope1{position:absolute; top:35; right:1075px}
#telescope2{position:absolute; top:62; right:680px}
#top{position:relative; top:0px; left:0px}
#contents{margin:0px;  position:relative;  top:0px;  left:0px;  width:100%;
height:400px; z-index:0;}
#codeContainer{margin:0px;    position:absolute;    top:0px;    left:0px;
overflow:auto; width:60%; height:400px; z-index:100;}
#canvasContainer{margin:0px;   position:absolute;   top:0px;   right:0px;
width:39%; height:400px; display:none;}
#buttons{position:relative; top:5px; left:0px; height:40px;}
#buttonsCode{position:relative; top:0px; left:0px; width:60%;}
#buttonsCanvas{position:absolute;  top:0px;  right:0px;  width:40%;  text-
align:right; z-index:101;}
#bottom{position:relative; top:5px; left:0px; margin-top:0px; display:none;
z-index:102;}*/
```

```
.wait{font-family:'Arial';    font-weight:bold;    color:#ffffff;    text-
align:center; vertical-align:middle%;}

table {
    font-size: 11px;
    margin-left: auto;
    margin-right: auto;
    padding:0px;
}

td {padding:3px;}

.col1, .col3{background:#D1D1D1;}
.col2 ,.col4{background: #ffffff;}

.border{
   border: 1px solid #D1D1D1;
   font-size: 11px;
}

p  {
    font-size: 11px;
}

ul {
      font-size:12px;
}

li {
      font-size:12px;
}
#tbl-matrix         {font-size: 11px; border:1px solid black; }
#tbl-matrix  td  {width:34px; height:30px; text-align:center;}
#m0,#m1,#m2        {color:#9BBB59;}
#m4,#m5,#m6        {color:#F0720A;}
#m8,#m9,#m10      {color:#C0504D;}
#m12,#m13,#m14  {color:#1f1f1f; font-weight:bold;}
#m3,#m7,#m11        {color:#888888;}
#slider-x-value, #slider-y-value, #slider-z-value {width:35px;}
```

```
#slider-position-x-value, #slider-position-y-value, #slider-position-z-value
,    #slider-dolly-value,    #slider-angle-x-value,    #slider-angle-y-value,
#slider-angle-z-value {width:35px; text-align:right;}
```

Файл коду div.css

```
#header
{
	height: 80px;
	background-color: #fff;
	margin-bottom: 10px;
}

#nav
{
	float: left;
	width: 29%;
	height: 80%;
	background-color: #fff;
	margin-bottom: 1px;
}

#content
{
	float: right;
	margin-left: 1%;
	width: 70%;
	height: 80%;
	background-color: #ccc;
	margin-bottom: 1px;
}
```

Файл коду ITP Kaleidoscope. Html

```
<html>

<head>
```

```
<title>ITP Kaleidoscope</title>
<meta http-equiv='content-type' content='text/html; charset=ISO-8859-1'>
<link href='css/styles.css'   type='text/css' rel='stylesheet'>
<link href='css/div.css' type='text/css' rel='stylesheet' />
<link   href='css/smoothness/jquery-ui-1.8.13.custom.css'   type='text/css'
rel='stylesheet' />
<link href='css/colorpicker.css'  type='text/css' rel='stylesheet'/>
<!-- GUI Libraries //-->
<script type='text/javascript' src='js/gui/jquery-1.5.1.min.js'></script>
<script            type='text/javascript'            src='js/gui/jquery-ui-
1.8.13.custom.min.js'></script>
<script type='text/javascript' src='js/gui/colorpicker.js'></script>
<!-- MATH Libraries //-->
<script type='text/javascript' src='js/math/gl-matrix-min.js'></script>
<!-- WEBGL Libraries //-->
<script type='text/javascript' src='js/webgl/Globals.js'></script>
<script type='text/javascript' src='js/webgl/Utils.js'></script>
<script type='text/javascript' src='js/webgl/Program.js'></script>
<script type='text/javascript' src='js/webgl/Scene.js'></script>
<script type='text/javascript' src='js/webgl/Axis.js'></script>
<script type='text/javascript' src='js/webgl/Floor.js'></script>
<script type='text/javascript' src='js/webgl/Camera.js'></script>
<script type='text/javascript' src='js/webgl/CameraInteractor.js'></script>
<script type='text/javascript' src='js/webgl/SceneTransforms.js'></script>
<script type='text/javascript' src='js/webgl/Texture.js'></script>
<script type='text/javascript' src='js/webgl/Lights.js'></script>
<script type='text/javascript' src='js/webgl/WebGLApp.js'></script>
<script type='text/javascript' src='js/webgl/Picker.js'></script>

<script id="shader-vs" type="x-shader/x-vertex">

const int NUM_LIGHTS = 4;

attribute vec3 aVertexPosition;
attribute vec3 aVertexNormal;
attribute vec4 aVertexColor;

uniform mat4 uMVMatrix;
uniform mat4 uPMatrix;
```

```
uniform mat4 uNMatrix;

uniform bool uTranslateLights;
uniform vec3 uLightPosition[NUM_LIGHTS];

varying vec3 vNormal;
varying vec3 vLightRay[NUM_LIGHTS];
varying vec3 vEye[NUM_LIGHTS];

void main(void) {

     vec4 c = aVertexColor;
     vec4 vertex = uMVMatrix * vec4(aVertexPosition, 1.0);
     vNormal = vec3(uNMatrix * vec4(aVertexNormal, 1.0));
     vec4 lightPosition = vec4(0.0);

     if (uTranslateLights){
         for(int i=0; i < NUM_LIGHTS; i++){
           lightPosition =   uMVMatrix * vec4(uLightPosition[i], 1.0);
           vLightRay[i] = vertex.xyz - lightPosition.xyz;
           vEye[i] = -vec3(vertex.xyz);
         }
     }
     else {
        for(int i=0; i < NUM_LIGHTS; i++){
          lightPosition = vec4(uLightPosition[i], 1.0);
          vLightRay[i] = vertex.xyz - lightPosition.xyz;
          vEye[i] = -vec3(vertex.xyz);
        }
    }
    gl_Position = uPMatrix * uMVMatrix * vec4(aVertexPosition, 1.0);
}
</script>

<script id="shader-fs" type="x-shader/x-fragment">
#ifdef GL_ES
precision highp float;
#endif
```

```
//Light uniforms
const int NUM_LIGHTS = 4;
uniform vec3  uLa[NUM_LIGHTS];   //ambient
uniform vec3  uLd[NUM_LIGHTS];   //diffuse
uniform vec3  uLs[NUM_LIGHTS];   //specular
uniform vec3 uLightPosition[NUM_LIGHTS];

//Material uniforms
uniform vec3  uKa;   //ambient
uniform vec3  uKd;   //diffuse
uniform vec3  uKs;   //specular
uniform float uNs;   //specular coefficient
uniform float d;     //Opacity
uniform int   illum; //Illumination mode


uniform bool  uWireframe;


varying vec3 vNormal;
varying vec3 vLightRay[NUM_LIGHTS];
varying vec3 vEye[NUM_LIGHTS];

float calculateAttenuation(in vec3 ray){
    float dist = length(ray);
    return (1.0 / (0.1 * dist));
}

void main(void) {
    if (uWireframe || illum == 0){
        gl_FragColor = vec4(uKd,d);
        return;
    }

   vec3 COLOR = vec3(0.0,0.0,0.0);
   vec3 N =  normalize(vNormal);
   vec3 L =  vec3(0.0,0.0,0.0);
   vec3 E =  vec3(0.0,0.0,0.0);
```

```
   vec3 R =  vec3(0.0,0.0,0.0);
   vec3 deltaRay = vec3(0.0);
   const int  lsize = 2;
   const float step = 0.25;
   const     float     inv_total     =     1.0/((float(lsize*lsize)     +
1.0)*(float(lsize*lsize) + 1.0));  //how many deltaRays

   float dx = 0.0;
   float dz = 0.0;
   float LT = 0.0;

   if (illum == 1){
        for(int i = 0; i < NUM_LIGHTS; i++){
            L = normalize(vLightRay[i]);
            N = normalize(vNormal);
            COLOR  +=  (uLa[i]  *  uKa)  +  (uLd[i]  *  uKd  *  clamp(dot(N,  -
L),0.0,1.0));
        }
        gl_FragColor =  vec4(COLOR,d);
        return;
   }

   if (illum == 2){
        for(int i = 0; i < NUM_LIGHTS; i++){

            E = normalize(vEye[i]);
            L = normalize(vLightRay[i]);
            R = reflect(L, N);
            COLOR += (uLa[i] * uKa);
            COLOR  +=  (uLd[i]  *  uKd  *  clamp(dot(N,-L),0.0,1.0));//  *
calculateAttenuation(vLightRay[i]));
            COLOR += (uLs[i] * uKs * pow( max(dot(R, E), 0.0), uNs) * 4.0);
        }
        gl_FragColor =  vec4(COLOR,d);
        return;
   }
}
</script>
```

```
<script type="text/javascript">

var camera = null,
light1 = null,
light2 = null,
light3 = null,
light4 = null,
transforms = null,

cameraHome = [0.0,0.4,7],
cameraAzimuth = 25,
cameraElevation = -11,
carSurface = undefined,
floorVisible = false,
translateLights = false;



function configure(){
    gl.clearColor(0.2,0.2,0.2, 1.0);
    gl.clearDepth(1.0);

    gl.enable(gl.DEPTH_TEST);
    gl.depthFunc(gl.LEQUAL);
    gl.enable(gl.BLEND);
    gl.blendFunc(gl.SRC_ALPHA, gl.ONE_MINUS_SRC_ALPHA);



    //Creates and sets up the camera location
    camera = new Camera(CAMERA_ORBITING_TYPE);
    camera.goHome(cameraHome);
    camera.setFocus([0.0,0.0,0.0]);
      camera.setAzimuth(cameraAzimuth);
      camera.setElevation(cameraElevation);
    camera.hookRenderer = render;

    //Creates and sets up the mouse and keyboard interactor
```

```
    var interactor = new CameraInteractor(camera,
document.getElementById('the-canvas'));

    //Scene Transforms
    transforms = new SceneTransforms(camera);

    //init transforms
    transforms.init();

    light1 = new Light('far-left');
      light1.setPosition([-25,25,-25]);
      light1.setDiffuse([0.4,0.4,0.4]);
    light1.setAmbient([0.0,0.0,0.0]);
    light1.setSpecular([0.8,0.8,0.8]);

      light2 = new Light('far-right');
      light2.setPosition([25,25,-25]);
      light2.setDiffuse([0.4,0.4,0.4]);
    light2.setAmbient([0.0,0.0,0.0]);
    light2.setSpecular([0.8,0.8,0.8]);

      light3 = new Light('near-left');
      light3.setPosition([-25,25,25]);
      light3.setDiffuse([0.4,0.4,0.4]);
    light3.setAmbient([0.0,0.0,0.0]);
    light3.setSpecular([0.8,0.8,0.8]);

    light4 = new Light('near-right');
    light4.setPosition([25,25,25]);
    light4.setDiffuse([0.4,0.4,0.4]);
    light4.setAmbient([0.0,0.0,0.0]);
    light4.setSpecular([0.8,0.8,0.8]);


      Lights.add(light1);
      Lights.add(light2);
    Lights.add(light3);
      Lights.add(light4);
```

```
    //init Program
      var attributeList = ["aVertexPosition",
                           "aVertexNormal",
                           "aVertexColor"];

      var uniformList = [     "uPMatrix",
                              "uMVMatrix",
                              "uNMatrix",
                    "uLightPosition",
                    "uWireframe",
                    "uLa",
                    "uLd",
                    "uLs",
                    "uKa",
                    "uKd",
                    "uKs",
                    "uNs",
                    "d",
                    "illum",
                    "uTranslateLights"
                              ];

      Program.load(attributeList, uniformList);


    gl.uniform3fv(Program.uLightPosition, Lights.getArray('position'));
    gl.uniform3fv(Program.uLa ,     Lights.getArray('ambient'));
    gl.uniform3fv(Program.uLd,      Lights.getArray('diffuse'));
    gl.uniform3fv(Program.uLs,      Lights.getArray('specular'));

    gl.uniform3fv(Program.uKa ,     [1.0,1.0,1.0]);
    gl.uniform3fv(Program.uKd ,     [1.0,1.0,1.0]);
    gl.uniform3fv(Program.uKs ,     [1.0,1.0,1.0]);

    gl.uniform1f(Program.uNs, 1.0);
    gl.uniform1i(Program.uTranslateLights, translateLights);

}
```

```
function loadBMW(){
    Scene.objects = [];
    Scene.addObject(Floor);
    for(var i = 1; i <= 24; i+=1){
        Scene.loadObject('models/cars/bmw/part'+i+'.json');

    }
}

function loadSphere(){
    Scene.objects = [];
    Scene.addObject(Floor);
    Scene.loadObject('models/sphere.json','sphere');
}

function loadMustang(){
    Scene.objects = [];
    Scene.addObject(Floor);
    Scene.loadObject('models/cars/bmw/part5.json','world');
    for(var i = 1; i <= 103; i+=1){
        Scene.loadObject('models/cars/mustang/part'+i+'.json');
    }

}

function loadBMWV3(){
    Scene.objects = [];
    Scene.addObject(Floor);
    Scene.loadObject('models/cars/bmw/part5.json','world');
    for(var i = 1; i <= 62; i++){
        Scene.loadObject('models/cars/bmwv3/part'+i+'.json');
    }
}

function loadAudi(){
    Scene.objects = [];
    Scene.addObject(Floor);
    Scene.loadObject('models/cars/bmw/part5.json','world');
    for(var i = 1; i <= 150; i+=1){
```

```
        Scene.loadObject('models/cars/audi/part'+i+'.json');
    }
}

function loadLamborghini(){
    Scene.objects = [];
    Scene.addObject(Floor);
    for(var i = 1; i <= 67; i+=1){
        Scene.loadObject('models/cars/lamborghini/part'+i+'.json');
    }
}




function loadNissan(){
    for(var i = 1; i <= 46; i+=1){
        Scene.loadObject('models/cars/nissan/part'+i+'.json');
    }
}

var selectedCar;
function load(){
    Floor.build(80,2);
    Floor.Ka = [1,1,1];
    Floor.Kd = [0.6,0.6,0.6]
    Floor.Ks = [1,1,1];
    Floor.Ni = 1;
    Floor.Ns = 1;
    Floor.d = 1.0;
    Floor.illum = 1;

    Floor.visible = floorVisible;

    Scene.addObject(Floor);
    loadBMW();
    selectedCar = 'bmwi8';
}
```

```
function render(){
    gl.viewport(0, 0, c_width, c_height);
    gl.clear(gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT);

   // orderObjectsInScene();
    transforms.updatePerspective();

    try{

        for (var i = 0; i < Scene.objects.length; i++){

            var object = Scene.objects[i];
                  if  (object.visible  !=  undefined  &&  !object.visible)
continue;
                  transforms.calculateModelView();
            transforms.push();
            transforms.setMatrixUniforms();
            transforms.pop();



            gl.enableVertexAttribArray(Program.aVertexPosition);
            gl.disableVertexAttribArray(Program.aVertexNormal);
            gl.disableVertexAttribArray(Program.aVertexColor);

            gl.uniform1i(Program.uWireframe, false);
            gl.uniform3fv(Program.uKa, object.Ka);
            gl.uniform3fv(Program.uKd, object.Kd);
            gl.uniform3fv(Program.uKs, object.Ks);
            gl.uniform1f(Program.uNs, object.Ns);
            gl.uniform1f(Program.d, object.d);
            gl.uniform1i(Program.illum, object.illum);

           if(object.d < 1.0){  //tweaking parameters here
                switch(selectedCar){
                    case 'bmwv3': gl.uniform1f(Program.d, 0.14); break;
                    case 'audi' : gl.uniform1f(Program.d, 0.8); break;
                    case 'mustang': gl.uniform1f(Program.d, 0.3); break;
```

```
                    case 'bmwi8': break; //nothing to do here
                }
                  }


            gl.bindBuffer(gl.ARRAY_BUFFER, object.vbo);
            gl.vertexAttribPointer(Program.aVertexPosition,   3,   gl.FLOAT,
false, 0, 0);
            gl.enableVertexAttribArray(Program.aVertexPosition);

                  if(!object.wireframe){
                        gl.bindBuffer(gl.ARRAY_BUFFER, object.nbo);
                        gl.vertexAttribPointer(Program.aVertexNormal,     3,
gl.FLOAT, false, 0, 0);
                        gl.enableVertexAttribArray(Program.aVertexNormal);
            }
            else{
                gl.uniform1i(Program.uWireframe, true);
            }

            gl.bindBuffer(gl.ELEMENT_ARRAY_BUFFER, object.ibo);

                  if (object.wireframe){
                gl.drawElements(gl.LINES,             object.indices.length,
gl.UNSIGNED_SHORT,0);
            }
            else{
                gl.drawElements(gl.TRIANGLES,         object.indices.length,
gl.UNSIGNED_SHORT,0);
            }

            gl.bindBuffer(gl.ARRAY_BUFFER, null);
            gl.bindBuffer(gl.ELEMENT_ARRAY_BUFFER, null);

        }
    }
    catch(err){
        alert(err);
        console.error(err.description);
```

```
    }
}

function resizeCanvas(){
    c_width = $('#content').width();
    c_height = $('#content').height();
    $('#the-canvas').attr('width',c_width);
    $('#the-canvas').attr('height',c_height);
}

$(window).resize(function(){resizeCanvas();});

var app;
function runShowRoom(){
    app = new WebGLApp("the-canvas");
    app.configureGLHook = configure;
    app.loadSceneHook   = load;
    app.drawSceneHook   = render;
    app.run();
}
</script>
</head>

<body onLoad='runShowRoom()'>
<div id="header">
    <h1>ITP Kaleidoscope</h1>
    <h2>Be near with magic!</h2>
    <div      id='telescope1'><img      src='telescope1.png'      width="85"
height="65"/></div>
      <div     id='telescope2'><img     src='telescope2.png'     width="60"
height="40"/></div>
</div>

<div id="nav">

    <table width='100%' cellspacing ='0'>
    <tr><td colspan=3><h3>Car</h3></td></tr>
    <tr><td  width='40%'>  Select  the  car  to  load:  </td><td  colspan='2'
width='60%'>
```

```
        <select id='select-car' >
                    <option value ='bmwi8'> BMW I8</option>
                    <option value ='mustang'> Ford Mustang </option>
                    <option value ='audi'> Audi R8 </option>
                    <option value ='bmwv3'> BMW v3 </option>
        </select>
        </td>
    </tr>
    <tr>
       <td>Car Color:</td>
       <td    colspan=2><div    id='carColor'    class='colorSelector'><div
style='background-color:rgb(255,255,255)'></div></div></td>
    </tr>
    <tr>
        <td>Car Shininness:</td>
        <td width='130px'><div id='slider-shininess'/></td>
        <td id='slider-shininess-value' width='60px'>  0</td>
      </tr>
    </table>

    <hr/>

    <table width='100%'>
        <tr>
            <td><h3>Lights</h3></td>
        </tr>
            <tr>
            <td>
                <table style='text-align:center'>
                <tr>
                    <td></td><td>Ambient</td><td></td>
                    <td>Diffuse</td><td></td>
                    <td>Specular</td><td></td>
                </tr>
                <tr>
                    <td   style='text-align:right'   width='160px'>far-left:
</td>
                    <td     width='150px'><div     id='slider-la1'/></td><td
id='slider-la1-value' width='20px'>0.0</td>
```

```
                    <td     width='150px'><div     id='slider-ld1'/></td><td
id='slider-ld1-value' width='20px'>0.4</td>
                    <td     width='150px'><div     id='slider-ls1'/></td><td
id='slider-ls1-value' width='20px'>0.8</td>
                </tr>
                <tr>
                    <td  style='text-align:right'  width='160px'>far-right:
</td>
                    <td     width='150px'><div     id='slider-la2'/></td><td
id='slider-la2-value' width='20px'>0.0</td>
                    <td     width='150px'><div     id='slider-ld2'/></td><td
id='slider-ld2-value' width='20px'>0.4</td>
                    <td     width='150px'><div     id='slider-ls2'/></td><td
id='slider-ls2-value' width='20px'>0.8</td>
                </tr>
                <tr>
                    <td  style='text-align:right'  width='160px'>near-left:
</td>
                    <td     width='150px'><div     id='slider-la3'/></td><td
id='slider-la3-value' width='20px'>0.0</td>
                    <td     width='150px'><div     id='slider-ld3'/></td><td
id='slider-ld3-value' width='20px'>0.4</td>
                    <td     width='150px'><div     id='slider-ls3'/></td><td
id='slider-ls3-value' width='20px'>0.8</td>
                </tr>
                <tr>
                    <td  style='text-align:right'  width='160px'>near-right:
</td>
                    <td     width='150px'><div     id='slider-la4'/></td><td
id='slider-la4-value' width='20px'>0.0</td>
                    <td     width='150px'><div     id='slider-ld4'/></td><td
id='slider-ld4-value' width='20px'>0.4</td>
                    <td     width='150px'><div     id='slider-ls4'/></td><td
id='slider-ls4-value' width='20px'>0.8</td>
                </tr>
                </table>
            </td>
        </tr>
        <tr>
```

```
            <td align='right'><input type="checkbox" id="translate-lights"/>
Translate Lights<br /></td>
        </tr>
    </table>
    <hr/>
    <table width='100%'>
        <tr>
            <td><h3>Camera</h3></td>
        </tr>
        <tr>
        <td align='center'>
        <div id='opt-pose' >
            <input  type='radio'  id='opt-home'    name='posecam'  /><label
for='opt-home'>Home</label>
            <input  type='radio'  id='opt-above'  name='posecam'  /><label
for='opt-above'>Above</label>
            <input  type='radio'  id='opt-front'  name='posecam'  /><label
for='opt-front'>Front</label>
            <input  type='radio'  id='opt-back'    name='posecam'  /><label
for='opt-back'>Back</label>
            <input  type='radio'  id='opt-left'    name='posecam'  /><label
for='opt-left'>Left</label>
            <input  type='radio'  id='opt-right'  name='posecam'  /><label
for='opt-right'>Right</label>

         </div>
         </td>
         </tr>
         <tr>
            <td  align='center'>Zoom:  Alt  +  Drag  <input  type="checkbox"
id="show-floor"/>Floor Visible<br /></td>
         </tr>
    </table>


</div>

<div id="content">
    <canvas id='the-canvas'></canvas>
```

```
</div>

<script type='text/javascript'>resizeCanvas();</script>
<script type='text/javascript'>
    $('#opt-pose').buttonset();
    $('#slider-shininess').slider({value:0.0,  min:0.0,  max:1.0,  step:0.1,
slide:function(event,                    ui){updateShininessSelectedCar(ui)},
change:function(event,ui){updateShininessSelectedCar(ui)}});
    $('#slider-la1').slider({value:0.0,    min:0.0,    max:1.0,    step:0.1,
slide:function(){updateLightProperty(1,                                  'a')},
change:function(){updateLightProperty(1,'a')}});
    $('#slider-ld1').slider({value:0.4,    min:0.0,    max:1.0,    step:0.1,
slide:function(){updateLightProperty(1,                                  'd')},
change:function(){updateLightProperty(1,'d')}});
    $('#slider-ls1').slider({value:0.8,    min:0.0,    max:1.0,    step:0.1,
slide:function(){updateLightProperty(1,                                  's')},
change:function(){updateLightProperty(1,'s')}});
    $('#slider-la2').slider({value:0.0,    min:0.0,    max:1.0,    step:0.1,
slide:function(){updateLightProperty(2,                                  'a')},
change:function(){updateLightProperty(2,'a')}});
    $('#slider-ld2').slider({value:0.4,    min:0.0,    max:1.0,    step:0.1,
slide:function(){updateLightProperty(2,                                  'd')},
change:function(){updateLightProperty(2,'d')}});
    $('#slider-ls2').slider({value:0.8,    min:0.0,    max:1.0,    step:0.1,
slide:function(){updateLightProperty(2,                                  's')},
change:function(){updateLightProperty(2,'s')}});
    $('#slider-la3').slider({value:0.0,    min:0.0,    max:1.0,    step:0.1,
slide:function(){updateLightProperty(3,                                  'a')},
change:function(){updateLightProperty(3,'a')}});
    $('#slider-ld3').slider({value:0.4,    min:0.0,    max:1.0,    step:0.1,
slide:function(){updateLightProperty(3,                                  'd')},
change:function(){updateLightProperty(3,'d')}});
    $('#slider-ls3').slider({value:0.8,    min:0.0,    max:1.0,    step:0.1,
slide:function(){updateLightProperty(3,                                  's')},
change:function(){updateLightProperty(3,'s')}});
    $('#slider-la4').slider({value:0.0,    min:0.0,    max:1.0,    step:0.1,
slide:function(){updateLightProperty(4,                                  'a')},
change:function(){updateLightProperty(4,'a')}});
```

```
    $('#slider-ld4').slider({value:0.4,    min:0.0,    max:1.0,    step:0.1,
slide:function(){updateLightProperty(4,                                 'd')},
change:function(){updateLightProperty(4,'d')}});
    $('#slider-ls4').slider({value:0.8,    min:0.0,    max:1.0,    step:0.1,
slide:function(){updateLightProperty(4,                                 's')},
change:function(){updateLightProperty(4,'s')}});


$('#select-car').change(function(){
    selectedCar = $('#select-car').val();
    switch (selectedCar){
        case 'bmwi8'    :loadBMW();     break;
        case 'mustang'  :loadMustang(); break;
        case 'audi'     :loadAudi();    break;
        case 'bmwv3'    :loadBMWV3();   break;
    }

});

$('#show-floor').change(function(){
          floorVisible = !floorVisible;
        Floor.visible = floorVisible;
});

$('#translate-lights').change(function(){
    translateLights = !translateLights;
    gl.uniform1i(Program.uTranslateLights, translateLights);
});

function updateShininessSelectedCar(shine){

    $('#slider-shininess-value').html(shine.value);


    for(var i = 0, N = Scene.objects.length; i < N; i+=1){
        for(var j =0; j <3; j+=1){
        Scene.objects[i].Ks[j] = shine.value;
        }
    }
```

```
    app.refresh();

}

function updateLightProperty(index,property){
    var v = $('#slider-l'+property+''+index).slider("value");
    $('#slider-l'+property+''+index+'-value').html(v);
    var light = undefined;
    switch(index){
                case 1: light = light1; break;
                case 2: light = light2; break;
                case 3: light = light3; break;
                case 4: light = light4; break;
    }

    switch(property){
                case 'a':light.setAmbient([v,v,v]);
                         gl.uniform3fv(Program.uLa,
Lights.getArray('ambient'));
                         break;
                case 'd':light.setDiffuse([v,v,v]);
                         gl.uniform3fv(Program.uLd,
Lights.getArray('diffuse'));
                         break;
                case 's':light.setSpecular([v,v,v]);
                         gl.uniform3fv(Program.uLs,
Lights.getArray('specular'));
                         break;
    }

    render();
}

function updateColorSelectedCar(color){
    var parts = [];
    switch(selectedCar){
        case 'bmwv3': parts.push(Scene.getObject('Shell_Mesh.023_Body'));
                  break;
        case 'mustang': for(var i=0, N = Scene.objects.length; i < N; i+=1){
```

```
                            if
(Scene.objects[i].alias.indexOf('_pintura_carro') != -1){
                                parts.push(Scene.objects[i]);
                            }
                        }
                        break;
        case 'audi': for(var i=0, N = Scene.objects.length; i < N; i+=1){
                            if (Scene.objects[i].alias.indexOf('_Lack') != -
1){
                                parts.push(Scene.objects[i]);
                            }
                        }
                        break;
        case 'bmwi8': for(var i=0, N = Scene.objects.length; i < N; i+=1){
                            if (Scene.objects[i].alias.indexOf('_BMW') != -
1){
                                parts.push(Scene.objects[i]);
                            }
                        }
                        break;
    }
    var object;
    for(var i = 0, N = parts.length; i < N; i+=1){
        parts[i].Kd = color;
    }
    app.refresh();
};

var carColorHex = '#ffffff';
$('#carColor').ColorPicker({
    onSubmit: function(hsb, hex, rgb, el) {
            $(el).val(hex);
            $(el).ColorPickerHide();

    },
    color: '#00ff00',
    onShow: function (colpkr) {
        $(colpkr).fadeIn(500);
        return false;
```

```
    },
    onHide: function (colpkr) {
        $(colpkr).fadeOut(500);
        return false;
    },
    onChange: function (hsb, hex, rgb) {
        carColorHex = hex;
        $('#carColor div').css('backgroundColor', '#' + hex);
        updateColorSelectedCar([rgb.r/256,rgb.g/256,rgb.b/256]);
    },

    onBeforeShow: function (colpkr) {
            $(colpkr).ColorPickerSetColor(carColor);
        }
    });


var timer_anim_camera = undefined;
var goal_camera_azimuth = 0;
var goal_camera_elevation = 0;
var goal_camera_home = cameraHome;



$('#opt-home').click(function(){
    if(timer_anim_camera) { clearInterval(timer_anim_camera); }
    goal_camera_azimuth = cameraAzimuth;
    goal_camera_elevation = cameraElevation;
    timer_anim_camera = setInterval(animCamera, 5);
});

$('#opt-above').click(function(){
    if(timer_anim_camera) { clearInterval(timer_anim_camera); }
    goal_camera_azimuth = 0;
    goal_camera_elevation = -90;
    timer_anim_camera = setInterval(animCamera, 5);
});

$('#opt-front').click(function(){
```

```
    if(timer_anim_camera) { clearInterval(timer_anim_camera); }
    goal_camera_azimuth = 0;
    goal_camera_elevation = -10;
    timer_anim_camera = setInterval(animCamera, 5);
});

$('#opt-back').click(function(){
    if(timer_anim_camera) { clearInterval(timer_anim_camera); }
    goal_camera_azimuth = 180;
    goal_camera_elevation = -10;
    timer_anim_camera = setInterval(animCamera, 5);
});

$('#opt-left').click(function(){
    if(timer_anim_camera) { clearInterval(timer_anim_camera); }
    goal_camera_azimuth = -90;
    goal_camera_elevation = 0;
    timer_anim_camera = setInterval(animCamera, 5);
});

$('#opt-right').click(function(){
    if(timer_anim_camera) { clearInterval(timer_anim_camera); }
    goal_camera_azimuth = 90;
    goal_camera_elevation = 0;
    timer_anim_camera = setInterval(animCamera, 5);
});



function animCamera(){

    var ca = goal_camera_azimuth-camera.azimuth;
    var ce = goal_camera_elevation-camera.elevation;
    var deltaPos = vec3.create([goal_camera_home[0]-camera.position[0],
                    goal_camera_home[1]-camera.position[1],
                    goal_camera_home[2]-camera.position[2]]);


    if (Math.abs(ca) < 0.5 &&
```

```
        Math.abs(ce) < 0.5 &&
        Math.abs(deltaPos[0]  < 0.1) &&
        Math.abs(deltaPos[1]  < 0.1) &&
        Math.abs(deltaPos[2]  < 0.1))
    {
        camera.setAzimuth(goal_camera_azimuth);
        camera.setElevation(goal_camera_elevation);
        camera.setPosition(goal_camera_home);
        clearInterval(timer_anim_camera);
    }
    else{
        camera.setAzimuth  (camera.azimuth   + ca/100);
        camera.setElevation(camera.elevation + ce/100);
        vec3.scale(deltaPos, 1/100);
        vec3.add(camera.position,deltaPos,deltaPos);
        camera.setPosition(deltaPos);
        render();
    }
}


</script>
<footer>
      <div id="footer">
            <h4>©KubibasByYasya!</h4>
      </div>
</footer>
</body>
</html>
```