МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ВОДНОГО ГОСПОДАРСТВА ТА ПРИРОДОКОРИСТУВАННЯ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ КІБЕРНЕТИКИ, ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА ІНЖЕНЕРІЇ

«До захисту допущена»
Зав. кафедри комп'ютерних наук та прикладної математики

_____

«__»_____20__р.

КВАЛІФІКАЦІЙНА РОБОТА

Проектування та розробка веб-додатку для 3D дизайну

Виконав: **Вітрук Сергій Олегович** _____
(підпис)
група КН-41

Керівник: **к.т.н., доцент Іванчук Н. В.** _____
(підпис)

Рівне 2024

ЗМІСТ

РЕФЕРАТ

Кваліфікаційна робота: 59 с., 15 рисунків, 1 таблиця, 5 додатків, 8 джерел.

Метою кваліфікаційної роботи є реалізація веб-сайту для дизайну 3D футболок з використанням React Three Fiber, що надає користувачам можливість створювати унікальні дизайни футболок і збереження результату.

Об'єкт дослідження – процес розробки веб-додатка для налаштування та візуалізації 3D дизайну футболок.

Предметом дослідження – методи та засоби створення інтерактивних 3D-сцен та інструментів налаштування дизайну футболок за допомогою технологій React та Three.js, інтегрованих у веб-додаток.

Методи дослідження – технології JavaScript, бібліотека React, рендер-двигун Three.js, а також інструменти для розробки, такі як Valtio і Tailwind CSS.

Актуальність теми зумовлена зростаючим попитом на персоналізовані продукти та інтерактивні веб-додатки, які надають користувачам більше свободи та контролю над процесом створення продукту. В умовах швидкого розвитку технологій інтернет-торгівлі, інструменти для створення індивідуальних дизайнів стають все більш популярними. Проект, виконаний у рамках даної кваліфікаційної роботи, дозволяє користувачам створювати та переглядати індивідуалізовані 3D моделі футболок, змінювати їхній дизайн, вибирати кольори, додавати графіку та текст, а також зберігати готовий дизайн на свій пристрій.

Ключові слова: дизайн 3D футболок, React, Three.js, JavaScript, Веб-додаток, тривимірна візуалізація.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

3D - тривимірний.

API - набір засобів розробки програмного забезпечення та стандартів, які полегшують взаємодію між різними програмами.

CSS — це мова стилів, яка використовується для визначення зовнішнього вигляду документів HTML та їх елементів.

DOM — це інтерфейс, який дозволяє програмам взаємодіяти з документами HTML і XML через об'єктну модель, яка представляє структуру та вміст документа.

HTML - є основною мовою розмітки для створення веб-сторінок. Він визначає структуру веб-сторінок і забезпечує функції гіпертексту.

JavaScript (JS) — це сценарна мова програмування, яка додає інтерактивність веб-сторінок та їх елементів.

React — це бібліотека JavaScript для розробки користувальницьких інтерфейсів, яка спрощує процес створення веб-додатків і покращує їх продуктивність.

React Three Fiber — це бібліотека для інтеграції Three.js у програми React, що дозволяє створювати вражаючу 3D-графіку в інтерактивних інтерфейсах.

Three.js — це бібліотека JavaScript для створення та відображення 3D-графіки безпосередньо у веб-браузері.

Valtio — це бібліотека керування станом у програмах React, яка використовує проксі-об'єкти для автоматичного оновлення компонентів і полегшення керування даними.

JSX — це розширення JavaScript, яке спрощує процес створення користувальницьких інтерфейсів, дозволяючи писати HTML-код безпосередньо в коді React.

Tailwind CSS — це структура CSS, яка надає набір готових стилів і утиліт для швидкого створення інтерфейсів без необхідності писати власні стилі.

VS Code — популярний редактор коду від Microsoft, який підтримує кілька мов програмування та інструментів розробки, що спрощує процес написання коду.

WebGL – JavaScript API для візуалізації інтерактивної 3D- і 2D-графіки у веб-браузері без необхідності додаткових плагінів, забезпечуючи високий рівень візуалізації та інтерактивності.

ВСТУП

У сучасному світі Інтернет-ресурси займають важливе місце в повсякденному житті людей, впливаючи на всі аспекти від комунікацій до бізнесу, розваг і так далі. Веб-дизайн відіграє ключову роль у залученні та утриманні користувачів, оскільки зовнішній вигляд і функціональність веб-сайту значно впливають на взаємодію з користувачем. З розвитком технологій очікування користувачів від веб-сайтів значно зросли. Відвідувачі прагнуть веб-сторінок, які є інтерактивними, захоплюючими та візуально привабливими. Однією з інноваційних тенденцій веб-дизайну є використання тривимірної (3D) графіки, яка дозволяє створювати більш реалістичні та деталізовані візуальні елементи.

Розробка веб-сайтів з використанням 3D-технологій стає все більш популярною та затребуваною, особливо в бізнесі, розважальному секторі та дизайні. Використання 3D-графіки у веб-дизайні відкриває нові можливості для взаємодії користувачів із вмістом, підвищує зацікавленість і надає унікальні можливості для персоналізації продуктів і послуг. Створення веб-сайту з 3D-дизайном футболок є актуальною темою, оскільки це дозволяє поєднати креативність 3D-моделювання з інтерактивним досвідом користувача. Інтерактивні 3D-програми виводять взаємодію користувача з продуктом на новий рівень, дозволяючи йому переглядати та редагувати дизайн у реальному часі.

У ході роботи над проектом було створено функціональний веб-сайт, який дозволяє користувачам створювати та переглядати індивідуалізовані 3D-моделі футболок, змінювати їхній дизайн, вибирати кольори, додавати графіку та текст, а також завантажувати результат на свій пристрій. Для забезпечення високої якості візуалізації та інтерактивності використовувались сучасні веб-технології, такі як React для побудови користувацького інтерфейсу, Three.js для роботи з 3D графікою та рендером, та React Three Fiber для спрощення та інтеграції цих технологій у єдину систему.

Проект пропонує якісну візуалізацію та інтерактивність, що робить процес створення дизайну футболок веселим та захоплюючим для користувачів. Користувачі можуть безпосередньо взаємодіяти з 3D-моделлю футболки, змінювати її параметри в реальному часі та отримувати миттєвий відгук про свої зміни. Це значно покращує зручність та ефективність процесу проектування, дозволяючи користувачам пробувати різні варіанти та знаходити найкраще рішення для своїх потреб.

Таким чином, цей проект має потенціал привернути увагу широкого кола клієнтів, які цінують унікальний і персоналізований одяг, а також може стати корисним інструментом для дизайнерів і підприємців, які прагнуть надавати своїм клієнтам інноваційні продукти та послуги. Впровадження таких рішень у бізнес-процеси може допомогти підвищити конкурентоспроможність компанії на ринку та надати нові можливості для залучення та утримання клієнтів.

РОЗДІЛ І ТЕОРЕТИЧНА ЧАСТИНА

1.1 ІСТОРІЯ РОЗВИТКУ 3D ВЕБ-ДОДАТКІВ

1.1.1 ІСТОРІЯ 3D-МОДЕЛЮВАННЯ

Історія розвитку 3D моделювання сягає глибоко в минуле, починаючи з появи персональних комп'ютерів. Початкові напрацювання в цій галузі відбувалися на основі математичних концепцій, що є основою 3D візуалізації. Навіть в Евкліда, відомого як "батька геометрії", що жив у III столітті до нашої ери, можна відстежити певні ключові ідеї. У XVII столітті Рене Декарт зробив вагомий внесок у цю область, розробивши основи аналітичної геометрії, відомої як координатна геометрія. Це дозволило точно вимірювати відстані та визначати місцезнаходження. У середині 1650-х роках британський математик Джеймс Джозеф Сильвестр розробив матричну математику, яка знайшла застосування в сучасній комп'ютерній графіці, де можна враховувати відблиски та зміни світла.

У 1960-х роках з появою перших комерційних систем автоматизованого проектування (CAD) народилася нова ера у розвитку 3D-моделювання. Вільям Феттер став піонером, вперше застосувавши формули для перетворення тривимірних координат у двовимірні, що стало основою для візуалізації (Феттер, 1962). У 1963 році американський вчений випустив на світ першу 3D-програму під назвою Sketchpad. Вона дозволяла створювати прості тривимірні об'єкти шляхом використання полігональних сіток - ребер, вершин і граней. Одним із ключових моментів її роботи було впровадження концепції "батько-дочірній", що дозволяла копіювати об'єкти з можливістю їхнього подальшого редагування. Іншим важливим досягненням Sketchpad був інструмент для автоматичного малювання геометричних фігур, де вказувалося лише положення та розміри.

Після успішного захисту дисертації Іван Сазерленд та доктор Девід Еванс вирішили заснувати перший факультет комп'ютерної графіки в Університеті Юти. Це сталося в 1968 році, коли вони також створили компанію під назвою "Evans & Sutherland", яка стала першою в галузі 3D-графіки. На початку своєї

діяльності їхнє підприємство спеціалізувалося на виробництві апаратного забезпечення для використання в їхніх розробках, проте згодом вони розширили свою діяльність й почали розробляти власне програмне забезпечення. Успіх компанії надихнув інших на відкриття власних підприємств та розвиток технології 3D-моделювання.

У 1972 році Фредерік Парке здійснив значний крок у розвитку 3D-моделювання, вперше змоделювавши людське обличчя. Він також розробив методи візуалізації, анімації та збору даних для створення "реалістичних" згенерованих комп'ютером напівтонових анімаційних послідовностей змінного виразу людських обличь. Було визначено, що для моделювання поверхні обличчя багатокутною оболонкою, що містила близько 250 полігонів та 400 вершин, було достатньо для досягнення реалістичного вигляду змодельованого обличчя (рис. 1.1).

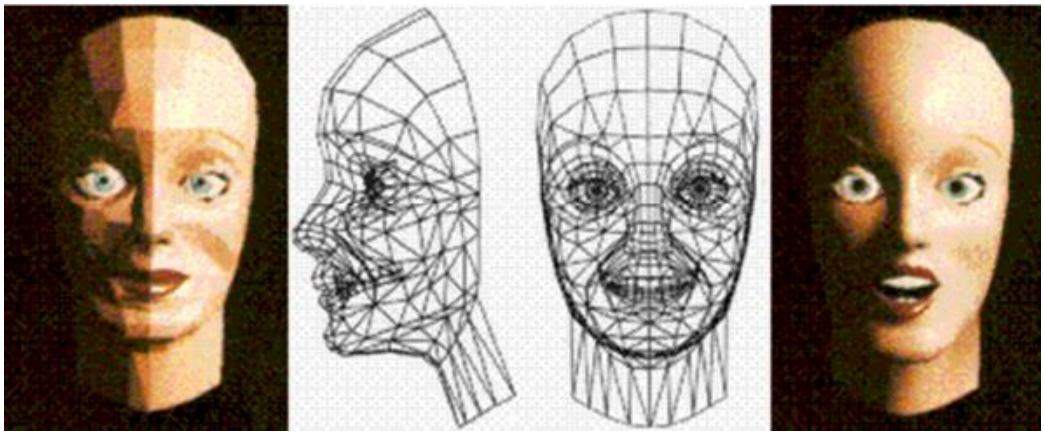

Рисунок 1.1 - Перше змодельоване людське обличчя

У подальшому розвитку комп'ютерної графіки в Університеті Юти, Анрі Гуро та Буй Тіон Фонг провели значні дослідження з методів затінення шляхом оптимізації алгоритмів візуалізації. Це дозволило спростити обробку та покращити рендеринг світла, відблисків і затінення. У 1972 році, під керівництвом Івана Сазерленда, Буй Фонг, Роберт Макдермотт, Джеймс Кларк і Рафаель Ром здійснили значний крок у напрямку комп'ютерної графіки, створивши 3D-зображення першого в історії детального відтворення фізичного оригіналу - Volkswagen Beetle (рис. 1.2).

Рисунок 1.2 - Перше в історії тривимірне зображення, що точно відтворює оригінал

У 1974 році Едвін Кетмелл опублікував докторську дисертацію, в якій детально розглянув фундаментальні аспекти, такі як суперпозиція текстур, бікубічна фрагментація тощо. Він визначив, що кожен піксель на екрані не лише має координати по осях абсцис і ординат, але також має значення по прикладній осі, яке відображає відстань від елемента до переднього плану. Якщо пікселі двох об’єктів перекриваються, на основі карти глибини відображається лише найближчий.

У 1975 році чайник Юта (рис. 1.3) став справжньою іконою у світі тривимірної комп'ютерної графіки, привертаючи особливу увагу. Завдяки своїм різноманітним структурам і поверхням, а також здатності відбивати світло і розкидати тінь, цей чайник став ідеальним об'єктом для досліджень. Його деталі були ретельно досліджені та поділені з колегами від Ньюелла, які вже незабаром використовували символічний об'єкт у своїх власних дослідженнях.

Рисунок 1.3 - Чайник Юта

У 1978 році Джеймс Брін вніс значний внесок у розвиток тривимірної комп’ютерної графіки, запропонувавши метод реалістичного зображення мікрорельєфів. Цей метод пізніше був удосконалений шляхом створення так званих «карт навколишнього середовища», які враховують не лише властивості поверхонь, але й середовища, в якому вони знаходяться. Такі карті враховують різноманітні фізичні явища, такі як дзеркальне відображення, прозорість, розсіювання та поглинання світла, а також заломлення світла при проходженні через прозорі матеріали. Ці фізичні явища стали основою для створення реалістичних зображень у 3D графічних редакторах.

З появою перших персональних комп’ютерів на ринку настане широке використання систем автоматизованого проектування (САПР) не лише у сферах аерокосмічної та автомобільної промисловості, але й у комерційних інженерних підприємствах. У цей період повноцінне 3D-моделювання почне активно розвиватися, стаючи ключовим напрямом у розробленні програмного забезпечення. Після 1980 року технології тривимірної графіки та моделювання почнуть постійно вдосконалюватися.

Спостерігалося:

- Винайдення 3D комп’ютерної графіки, яка здатна відображати кольорові зображення на екрані.

- Демонстрація перших тривимірних кінозображень.
- Перехід до фотореалістичної анімації, що відзначиться створенням комп'ютерних анімаційних повнометражних фільмів.
- У 1994 році винахід рухомих зображень, які можуть бути розмитими і імітувати рухи людей і тварин, відкриваючи нові можливості для 3D анімації.
- У 1995 році виходить перша повнометражна 3D анімація Pixar «Історія іграшок», що стає візитною карткою успішного використання 3D технологій у кіноіндустрії.

У XXI столітті 3D-моделювання стало необхідним елементом в багатьох галузях, включаючи архітектуру, інтер'єрний дизайн, виробництво відеоігор, анімацію, медицину, наукові дослідження та інші. Це привело до розвитку спеціалізованих інструментів та технік, які дозволяють створювати більш реалістичні та деталізовані 3D-моделі для різних потреб.

На цьому етапі розвитку 3D моделювання можна спостерігати величезне розмаїття виробів практичного застосування, створених завдяки цій технології. Від кісток до бетону, від скелету людини до будівельних конструкцій, 3D моделювання знайшло застосування у широкому спектрі сфер. Наприклад, було розроблено робочу нирку, яка була надрукована на 3D принтері. Перший 3D фільм Діснея – «Красуня і чудовисько» – відкрив нову еру в кіноіндустрії, показуючи можливості цієї технології у створенні емоційно зворушливих історій. Також вперше було розроблено та надруковано протез ноги, що відкрило нові можливості для медичного застосування 3D технологій. У 2010 році вперше було надруковано тривимірний автомобіль, корпус якого був повністю створений на 3D-принтері. Ці приклади свідчать про широкий потенціал 3D-моделювання у вирішенні різних завдань та викликів у сучасному світі.

Історія розвитку 3D моделювання свідчить про значний прогрес у цій галузі протягом останніх десятиліть. Від початкових примітивних методів до

сучасних комп’ютерних програм та технологій, використання 3D моделювання дозволяє створювати складні, реалістичні та деталізовані моделі. Сучасний стан розвитку 3D моделювання характеризується постійними зрушеннями в напрямку покращення якості моделей, швидкості рендерингу та масової доступності програмного забезпечення для тривимірного моделювання.

З’являються нові методи та алгоритми, які дозволяють реалістично відтворювати матеріали, освітлення та фізичну взаємодію об’єктів. Вплив 3D моделювання на різні галузі, такі як архітектура, медицина, ігрова індустрія, кіно та дизайн, є надзвичайно великим, адже дозволяє проектувати та створювати складні об’єкти, віртуальні середовища та інтерактивні симуляції, що відкриває безліч можливостей для творчості та інновацій [1].

1.1.2 3D ВЕБ-ДИЗАЙН САЙТІВ

Сучасні веб-користувачі вимагають відвідувати сайти, які не тільки надають інформацію, а й захоплюють їх увагу та забезпечують цікавий досвід користування. У такому контексті важливо створити веб-сайт, який виділяється з-поміж інших. Підвищення популярності та попиту на 3D веб-дизайн свідчить про те, що цей підхід стає все більш важливим.

Існує безліч способів створення захопливого дизайну для веб-сайтів, таких як паралаксна прокрутка, мікровзаємодії та анімація. Однак 3D веб-дизайн відкриває нові можливості для творчого виразу та залучення уваги аудиторії. Завдяки 3D графіці та анімації, веб-сайти можуть надавати користувачам реалістичні та захоплюючі візуальні ефекти, які створюють враження глибини, об'єму та реалізму.

Використання 3D елементів у веб-дизайні дозволяє створювати динамічні та вражаючі ефекти, які привертають увагу користувачів і створюють неповторний досвід перегляду веб-сайту. Це може бути використання тривимірних об'єктів, анімованих переходів між сторінками, а також взаємодія з тривимірними об'єктами на сторінці.

Загалом, 3D веб-дизайн є потужним інструментом для створення унікального та захоплюючого досвіду користувача на веб-сайті. Його популярність продовжує зростати, оскільки він дозволяє веб-дизайнерам та розробникам створювати вражаючі візуальні ефекти та покращувати користувацький досвід.

3D веб-дизайн - це використання тривимірної анімації та інших тривимірних візуальних елементів для створення ілюзії глибини та реалістичності на веб-сайті. Технічно це означає розміщення об'єктів у тривимірному просторі уздовж осей x, y та z.

За допомогою 3D веб-дизайну можна створювати неперевершені візуальні ефекти, що роблять веб-сайт вражаючим та запам'ятовуваним. Це можуть бути анімовані тривимірні об'єкти, глибина та об'ємність, що додають глибини сторінці та створюють ілюзію реальності.

3D веб-дизайн може бути використаний різними способами. Наприклад, економне використання 3D елементів може підкреслити певні функції чи візуальні ефекти. З іншого боку, його можна використовувати для створення вражаючих користувацьких дій, які нагадують доповнену реальність. Наприклад, інтерактивні елементи, які реагують на рух миші або жести, можуть створювати захоплюючий досвід для користувача [2].

Отже, 3D веб-дизайн - це сучасний інструмент, який дозволяє створювати вражаючі та інтерактивні веб-сайти, що відзначаються глибиною та реалістичністю.

Історія 3D-дизайну справді налічує вже понад півстоліття. У 1960-х роках перші комп'ютерні програмісти вже вирішували завдання з 3D-моделювання. Перша програма для 3D-дизайну, яку ми вже згадували раніше, називалася Sketchpad. Вона стала піонером використання графічного інтерфейсу користувача, вкладаючи фундамент для подальшого розвитку автоматизованого проектування.

У 1994 році була представлена мова розмітки віртуальної реальності — формат файлів, який вперше дозволив веб-дизайнерам створювати 3D-об'єкти.

У 1997 році Flash (рис. 1.4) з'явився на сцені, але, хоча це дозволило розробникам використовувати 3D-графіку та анімацію, він також завантажувався довго.

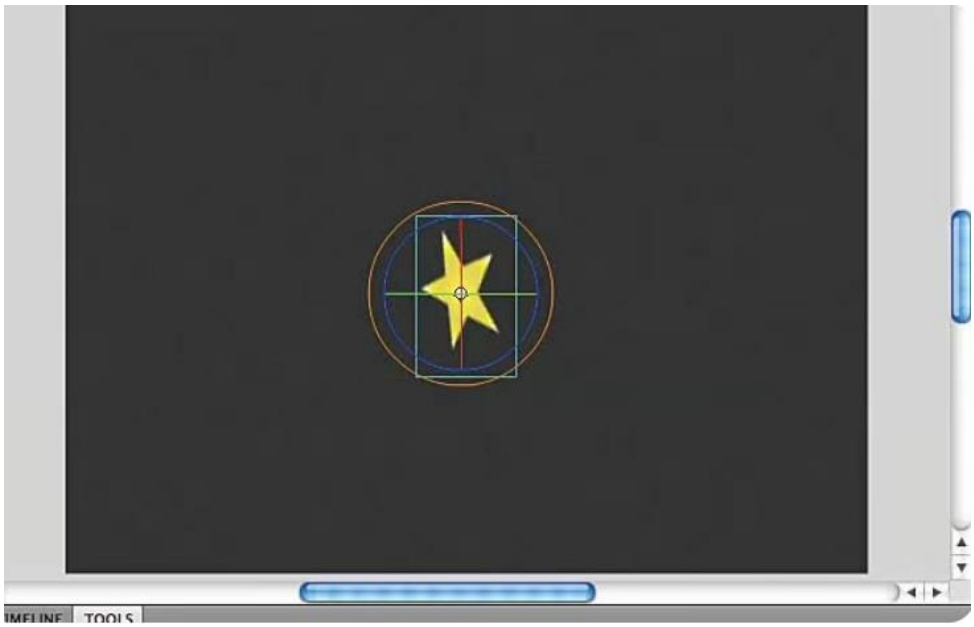

Рисунок 1.4 - Інтерфейс сцени у Flash

До 2010 року скевоморфізм, концепція, що полягає в створенні предметів, що виглядають так, ніби вони є їхніми аналогами у реальному світі, раптово став дуже популярним. Однак, оскільки цей підхід насправді не надавав багато користі сайтам, крім яскравих візуальних елементів, ця тенденція швидко втратила свою актуальність. Замість цього, повернувся плоский і статичний дизайн – принаймні на певний період часу.

WebGL, безкоштовний API, який дозволяв розробникам створювати 3D-дизайни, сумісні з більшістю веб-браузерів, був розроблений у 2011 році, але проблема полягала в тому, що він вимагав використання JavaScript, Java або C разом із GLSL для програмування вашого 3D об’єкта.

На щастя, сьогодні є кілька шляхів для створення власного персоналізованого 3D-дизайну веб-сайтів. Незалежно від того, чи володієте ви JavaScript а саме Three.js чи навіть не маєте досвіду з програмуванням, є програми та інструменти, які допоможуть вам створювати 3D-дизайни, які ви хочете [3].

Ось деякі з найпопулярніших програм та інструментів для 3D-дизайну:

AutoCAD: Це програмне забезпечення для автоматизованого проектування, що дозволяє створювати точні 2D і 3D креслення.

Blender: Це безкоштовний набір інструментів для 3D-комп’ютерної графіки з відкритим кодом, який можна використовувати для створення 3D-моделей та інтерактивних 3D-додатків.

SketchUp: Це програма для 3D-дизайну з простим, але надійним набором інструментів, який дозволяє створювати 3D-моделі.

Vectary: Це онлайн-інструмент для 3D-моделювання, який дозволяє створювати, ділитися та налаштовувати 3D-проекти.

Vev: Це платформа створення веб-сайтів, яка дозволяє легко інтегрувати 3D-об’єкти для створення та публікації повністю адаптивних сайтів без будь-якого програмування.

1.2 ВИКОРИСТАНІ ТЕХНОЛОГІЇ

1.2.1 REACT

React, також відомий як React.js або ReactJS, був створений у 2011 році Джорданом Валке, розробником Facebook. Спочатку він використовувався в стрічці новин Facebook, і його головна мета — покращити продуктивність і забезпечити зручну роботу користувача. Пізніше React був реалізований в Instagram щоб підвищивши його актуальність і функціональність [4].

Публічний реліз React відбувся в 2013 році. Відтоді він набув величезної популярності серед розробників веб-додатків. Основними факторами, що сприяють такому успіху, є висока продуктивність, простота використання та можливість створювати складні інтерфейси користувача.

Основними цілями при створенні React були:

- Простота: забезпечує простіший спосіб створення динамічних веб-інтерфейсів.
- Ефективність: використовуйте віртуальний DOM для зручного й ефективного оновлення інтерфейсу користувача.
- Декларативний підхід: Надає можливість розробникам визначати, як інтерфейс повинен виглядати для певного стану, а не крок за кроком описувати, як досягти цього стану.

React став популярним завдяки своїй компонентній архітектурі, яка дозволяє створювати складні інтерфейси шляхом поєднання простих і незалежних компонентів. Кожен компонент у React відповідає за власну логіку та відтворення, що полегшує розробку, тестування та повторне використання коду.

Основні принципи React:

- Компоненти: програми React складаються з компонентів, які можна багаторазово використовувати. Кожен компонент інкапсулює власний рендеринг і поведінку.

- JSX: React використовує розширення синтаксису JavaScript під назвою JSX, яке дозволяє писати HTML-подібний код у JavaScript, роблячи код більш зрозумілим і зручним для користувача.
- Віртуальний DOM: React використовує віртуальну версію DOM, щоб визначити найефективніший спосіб оновлення справжнього DOM. Це значно покращує продуктивність програми.
- Одностороннє прив’язування даних: потік даних у React є одностороннім, що полегшує розуміння та налагодження стану програми.

Тому React є потужним і гнучким інструментом для створення сучасних веб-додатків, що дозволяє розробникам ефективно управляти складністю і забезпечувати високий рівень інтерактивності та продуктивності [5].

1.2.2 THREE.JS

Що таке Three.js?

Three.js – це бібліотека JavaScript для створення тривимірної графіки, яка забезпечує простіший і потужніший інтерфейс для роботи з WebGL. Вона дозволяє розробникам створювати структуровані та легко читані 3D-додатки. Three.js використовується в різних сферах, включаючи онлайн-ігри, демонстрації та візуалізації моделей. Завдяки великій кількості функцій, вона надає можливість створювати складні й захоплюючі 3D-проекти.

Перша версія Three.js була випущена Рікардо Кабельо на GitHub 23 квітня 2010 року. Незабаром до проекту долучилися Браніслав Улічний та Джошуа Ку, які внесли значний вклад у розвиток геометрії, матеріалів та постобробки. З моменту першого релізу бібліотека постійно вдосконалюється: за останні дев’ять років було зроблено понад 25 000 комітів від більш ніж 1000 різних учасників.

Three.js значно спрощує процес вивчення та використання WebGL. Це робить її ідеальною для початківців, які хочуть освоїти 3D-розробку. Бібліотека має низький поріг входу, що дозволяє новачкам швидко створювати тривимірні додатки без глибоких знань WebGL. Для досвідчених розробників Three.js забезпечує швидке створення 3D-проектів з меншими зусиллями і меншим обсягом коду. Three.js дозволяє зосередитися на графіці та анімації, не переймаючись технічними деталями взаємодії з апаратним забезпеченням.

Новачкам, які хочуть розпочати створювати 3D-додатки, не потрібно вивчати складний WebGL. Використовуючи Three.js, вони можуть досягти тих самих результатів з меншими витратами часу та зусиль. Бібліотека надає всі необхідні інструменти для створення інтерактивних тривимірних сцен, об’єктів та анімацій.

Three.js значно полегшує процес створення тривимірних додатків, надаючи зручний інтерфейс для роботи з 3D-графікою та багатий набір функцій для розробки вражаючих проектів.

Існує багато етапів у процесі створення проекту на Three.js, але розбивши його на частини, набагато легше зрозуміти, як все працює. Давайте розглянемо основні компоненти.

Спочатку ми створюємо камеру, сцену та рендер.

```
// Встановимо розміри області перегляду для камери
var canvasContainer = document.getElementById("jscanvas");
// Створимо новий екземпляр камери
var perspectiveCamera = new THREE.PerspectiveCamera(90, canvasContainer.offsetWidth / canvasContainer.offsetHeight, 1, 2000);
// Встановимо початкове значення координати "z" камери на 500. За замовчуванням координати x, y і z дорівнюють 0
perspectiveCamera.position.z = 500;
```

Камера діє як об’єктив, який використовується для перегляду проекту. Залежно від її кута та положення, ми можемо бачити різні частини нашої сцени. У цьому прикладі ми не будемо змінювати камеру після її ініціалізації.

При ініціалізації камери ми встановлюємо значення поля зору, співвідношення сторін, ближньої та дальньої площини. Ми встановлюємо співвідношення сторін відповідно до розміру нашого елемента div. Координата z змінюється на 500, щоб ми могли побачити куб, розташований в центрі сцени.

Сцена — це простір, де ми розміщуємо всі наші об’єкти. Наприклад, у нашій сцені ми додамо куб з певним розміром і положенням. Якщо наша камера спрямована на об’єкти в сцені, вони будуть видимі користувачу. Згодом ми будемо змінювати об’єкт у сцені, обертаючи куб, який оновлюватиметься в сцені.

```
// Ініціалізуємо WebGL рендерер
var webGLRenderer = new THREE.WebGLRenderer();
// Встановлюємо співвідношення пікселів до розміру дисплея
webGLRenderer.setPixelRatio(window.devicePixelRatio);
// Встановлюємо розміри рендера відповідно до розмірів контейнера
webGLRenderer.setSize(canvasContainer.offsetWidth, canvasContainer.offsetHeight);
```

Рендерер відповідає за обробку сцени. Це частина, яка використовує WebGL для відображення сцени в проекті. Без рендерера сцена та проект залишаються невізуалізованими даними. Нам також потрібно встановити співвідношення пікселів і розмір, щоб об’єкти не спотворювалися у браузерах різних розмірів.

```
// Створюємо геометрію для куба
var boxGeometry = new THREE.BoxGeometry(100, 100, 100);
// Визначаємо матеріал для геометрії
var boxMaterial = new THREE.MeshNormalMaterial();
// Об'єднуємо геометрію з матеріалом
var boxMesh = new THREE.Mesh(boxGeometry, boxMaterial);
// Додаємо створений куб до сцени
scene.add(boxMesh);
```

Щоб створити об’єкт, нам потрібні три складові: геометрія, матеріал і сітка. Геометрія визначає форму об’єкта та його розташування (у цьому випадку — куб 100x100x100). Матеріал — це текстура об’єкта. Ми використовуємо MeshNormalMaterial, який показує простий спектр кольорів по всьому об’єкту. Сітка об’єднує геометрію та матеріал і додається до сцени для відображення.

```
// Додаємо сцену до HTML-сторінки
document.getElementById(«jscanvas»).appendChild(renderer.domElement);
// Починаємо анімацію сцени
animate();
```

Останні два рядки є надзвичайно важливими. Перший додає сцену до нашого HTML-файлу за допомогою рендерера, що дозволяє нам бачити проект на сторінці. Функція анімації використовується для постійного оновлення сцени, що дозволяє відображати динамічні зміни в реальному часі.

Three.js значно полегшує створення тривимірних додатків, надаючи зручний інтерфейс для роботи з 3D-графікою. Завдяки цьому, як новачки, так і досвідчені розробники можуть швидко створювати та впроваджувати захоплюючі 3D-проекти [6].

1.2.3 FIBER

React Three Fiber — це бібліотека, яка дозволяє використовувати Three.js у додатках на React. Вона забезпечує потужний інтерфейс для створення тривимірних сцен, використовуючи переваги декларативного стилю програмування React. Замість того, щоб писати великий обсяг коду для створення і управління 3D-сценами в Three.js, React Three Fiber дозволяє робити це більш інтуїтивно і ефективно.

Особливості та переваги React Three Fiber:

- Декларативний підхід: React Three Fiber дозволяє визначати 3D-сцени за допомогою JSX, що робить код більш читабельним і зрозумілим. Це також дозволяє легше управляти станом і життєвим циклом компонентів.
- Інтеграція з React екосистемою: Використовуючи React Three Fiber, ви можете легко інтегрувати 3D-графіку з іншими бібліотеками React.
- Легкість у використанні: React Three Fiber абстрагує багато складних аспектів Three.js, спрощуючи створення і управління 3D-сценами. Це дозволяє розробникам зосередитися на створенні вражаючих візуальних ефектів без необхідності глибоко занурюватися в технічні деталі.
- Розширюваність: React Three Fiber підтримує всі можливості Three.js і дозволяє використовувати будь-які об'єкти, шейдери або плагіни Three.js, що робить його надзвичайно гнучким.

Для створення 3D-сцени за допомогою React Three Fiber, спочатку потрібно встановити бібліотеку. Після встановлення можна почати створювати 3D-сцени у своєму React-додатку.

```
function Cube() {
  return (
    <mesh>
      <boxGeometry args={[2,2,2]} />
```

```
      <meshStandardMaterial color={‘red’} />
    </mesh>
  );}
function Scene() {
  return (
    <Canvas>
      <ambientLight />
      <pointLight position={[12, 12, 12]} />
      < Cube />
    </Canvas>
  ); }
export default Scene;
```

Canvas: Це компонент від React Three Fiber, який створює сцену та рендерер Three.js. Він обгортає всі 3D-компоненти і дозволяє їм бути частиною 3D-сцени.

Box: Це користувацький компонент, який створює куб. Він використовує Three.js об'єкти mesh, boxGeometry і meshStandardMaterial для створення 3D-фігури.

Світло: Компоненти ambientLight і pointLight додають освітлення до сцени, що дозволяє побачити куб і надає йому об'єму.

Цей приклад демонструє, наскільки легко можна створювати та управляти 3D-об'єктами за допомогою React Three Fiber. Бібліотека дозволяє використовувати всі потужні можливості Three.js, зберігаючи при цьому простоту та гнучкість React [7].

Таблиця 1. Порівняння традиційного Three.js і React Three Fiber

Аспект	**Традиційний Three.js**	**React Three Fiber**
Структура коду	Ручне створення та керування сценою, камерою, рендерером, об’єктами.	JSX-подібні компоненти, що представляють 3D-об’єкти, світло та камеру в **<Canvas>**.
Управління сценою	Вручну додавати та видаляти об’єкти зі сцени.	Компоненти автоматично монтуються, коли вони з’являються в JSX і виходять із нього.
Анімація та оновлення	Ручне налаштування циклу візуалізації та анімації.	Використання методів життєвого циклу React і хуків для оновлень і анімації.
Крива навчання	Крутіше для тих, хто не знайомий з програмуванням тривимірної графіки.	Простіше для розробників із фоном React; більш плавна інтеграція в програми React.
Інтеграція з React	Окремо від екосистеми React; потребує ручної інтеграції.	Повна інтеграція з екосистемою React (хуки, контекст тощо).
Випадок використання	Підходить для додатків, орієнтованих на 3D-графіку без React.	Ідеально підходить для додатків React, яким потрібні можливості 3D-графіки.
Налагодження та інструменти	Традиційні засоби налагодження JavaScript.	React DevTools можна використовувати для перевірки та налагодження сцени.
Гнучкість	Високий рівень контролю над відтворенням та оптимізацією графіки.	Простіший у налаштуванні та використанні, але дещо менший контроль над деталями відтворення.

РОЗДІЛ ІІ. ЗАГАЛЬНИЙ ОПИС ПРОЕКТУ

2.1 ПОСТАНОВКА ЗАДАЧІ

Цей проект спрямований на створення веб-сайту з тривимірним дизайном футболок, який дозволить користувачам легко та інтуїтивно зрозуміло створювати власні унікальні дизайни. Основні функції сайту включають можливість перегляду футболку, налаштувати кольори, додати зображення та текст, а також переглядати результати в реальному часі за допомогою 3D-графіки та завантажувати фотографії результатів на свій пристрій. Основна мета полягає в тому, щоб веб-сайт був дуже інтерактивним і візуально привабливим, що сприяло б підвищенню залученості користувачів і їхньому задоволенню.

Завдання проекту включають:

Розробка інтуїтивного інтерфейсу користувача:

- Створення зручної навігації та зрозумілого інтерфейсу для користувачів різного рівня технічної підготовки.
- Впровадження функціональності для вибору та налаштування футболок, кольору та додаткових елементів дизайну.

Інтеграція тривимірної графіки:

- Використання бібліотеки React Three Fiber для інтеграції 3D моделей у веб-додаток.
- Забезпечення реалістичного відображення 3D моделей футболок з можливістю обертання та перегляду під різними кутами.

Реалізація функції завантаження та редагування зображень та текстів:

- Надання користувачам можливості завантажувати власні зображення та тексти для додавання на футболки.
- Забезпечення інструментів для редагування та розташування цих елементів на 3D моделі.

Завантаження результат на пристрій:

- Можливість завантаження фото дизайну з різних кутів собі на пристрій для збереження результату.

Тестування та відлагодження:

- Проведення різного тестування веб-додатку для виявлення та усунення можливих помилок.

Цей проект має на меті поєднати найновіші технології веб-розробки з сучасними інструментами для роботи з тривимірною графікою, щоб створити інноваційний та корисний продукт для кінцевих користувачів. Результатом стане повнофункціональний веб-сайт, що надасть користувачам можливість легко створювати та переглядати свої власні дизайни футболок у тривимірному просторі.

2.2 МОВИ ПРОГРАМУВАННЯ

У цьому проекті основною мовою програмування є JavaScript, яка необхідна для створення інтерактивних веб-додатків. Використання JavaScript пропонує безліч можливостей для динамічної взаємодії з Інтернетом і маніпулювання веб-елементами без необхідності перезавантаження.

JavaScript використовується для таких основних функцій у проекті:

Дизайн інтерфейсу користувача:

- Розробка інтерактивних функцій, таких як кнопки, меню та форми, які дозволяють користувачам легко взаємодіяти з програмою.
- Динамічно оновлювати вміст сторінки у відповідь на дії користувача, наприклад, змінювати колір футболки або додавати новий компонент дизайну.

Інтеграція 3D-графіки з React 3 Three Fiber:

- Використання бібліотеки React Three Fiber, яка є оболонкою бібліотеки Three.js, для відтворення 3D-моделей безпосередньо в додатку React.
- Реалізація 3D-об’єктів і сцен, які користувачі можуть переглядати та керувати ними в реальному часі.

Обробка завантажених користувачем даних:

- Реалізація можливості завантаження зображень та текстів, які користувачі можуть використовувати для налаштування своїх футболок.
- Використання Canvas API для обробки та відображення зображень перед додаванням їх до 3D моделі.

Оптимізація продуктивності та коду:

- Використання сучасних можливостей JavaScript, таких як ES6+ синтаксис, для написання чистого та ефективного коду.

- Оптимізація роботи з 3D графікою та обробки зображень для забезпечення швидкої та плавної роботи додатка навіть на мобільних пристроях.

Окрім JavaScript, у проекті також використовуються HTML та CSS, що є основними технологіями для створення веб-сторінок:

HTML:

- HTML використовується для структурування вмісту веб-сторінки шляхом визначення її елементів та їх зв’язків.
- Він надає базову структуру сторінки, у яку вбудовано всі інші елементи, включаючи інтерактивні компоненти, створені за допомогою JavaScript.

CSS:

- CSS відповідає за візуальний дизайн веб-сторінок шляхом визначення стилів для різних елементів HTML.
- Використовуючи CSS, ми можемо надати привабливі та зручні інтерфейси, які включають кнопки стилю, меню, форми та інші елементи керування.
- CSS також допомагає адаптувати інтерфейс до різних розмірів екрана та пристроїв, забезпечуючи сумісність між платформами.

Три технології JavaScript, HTML і CSS поєднуються, щоб створити сучасну, інтерактивну та функціональну веб-програму для тривимірного дизайну футболок. Поєднання цих мов програмування з React і бібліотекою React Three Fiber може значно підвищити продуктивність розробки та захоплюючий інтерфейс користувача.

2.3 СЕРЕДОВИЩЕ РОЗРОБКИ

Для реалізації цього проекту в якості основного середовища розробки використовується Visual Studio Code (VS Code). VS Code — потужний і універсальний текстовий редактор, який пропонує широкий спектр можливостей для розробки веб-додатків. Він підтримує багато мов програмування та технологій, включаючи JavaScript, HTML, CSS і React, що робить його ідеальним інструментом для цього проекту.

Основні переваги використання VS Code для розробки проекту:

1. Розширюваність:
   - VS Code має величезну кількість всіляких розширень, які допоможуть значно розширити його функціональність.
2. Зручний дебагінг:
   - Інструменти для дебагінгу, вбудовані у VS Code, дозволили швидко виявляти та виправляти помилки у коді. Це дуже важливо для нашого веб-додатку, де потрібно слідкувати за багатьма взаємодіями між компонентами та даними.
3. Живий сервер:
   - Розширення Live Server забезпечує можливість оновлення сторінки в режимі реального часу у браузері при збереженні змін у коді. Це особливо пришвидшило процес розробки, що дозволяює миттєво бачити результати змін.
4. Інтелектуальне автозавершення:
   - Функція автоматичного автозавершення у VS Code допомогла швидко та якісно писати код, пропонуючи варіанти завершення для функцій, методів та змінних, що значно зменшило кількість помилок та пришвидшило написання коду.
5. Зручний інтерфейс:
   - Інтерфейс VS Code є доволі прости та зрозумілим і налаштовується відповідно до потреб розробника. Можливість розділення вікон,

використання різних тем та зміни кольору коду зробила робочий процес більш комфортним та продуктивним.

6. Вбудований термінал:
   - Вбудований термінал є зручним та дозволив запускати команди npm для управління залежностями та запуску проекту розробки без зайвого перемикатися між різними вікнами, що також сприяло більш ефективному робочому процесу.

Загалом, використання Visual Studio Code як середовища розробки забезпечило ефективність та зручність у роботі над проектом, що дозволило зосередитися на створенні функціонального та привабливого веб-додатку для дизайну футболки.

РОЗДІЛ ІІІ. ОПИС МОЖЛИВОСТЕЙ ПРОЕКТУ

3.1 АРХІТЕКТУРА ПРОЕКТУ

Архітектура проекту побудована за модульним принципом, що забезпечує гнучкість і масштабованість. Наш проект складається з кількох основних папок і файлів, кожен з яких виконує свою специфічну роль.

Папка node_modules:

Містить всі встановлені залежності та бібліотеки, необхідні для роботи проекту. Вміст цієї папки генерується автоматично при встановленні залежностей за допомогою менеджера пакетів npm. Залежності включають такі бібліотеки, як React, Three.js, Framer Motion та інші.

Папка public:

Містить статичні файли, які обслуговуються сервером напряму. В цю папку входять фото, іконки сайту, 3D модель футболки, логотип, текстури. Ці файли використовуються для відображення візуальних елементів на веб-сторінці та забезпечують інтерактивність додатка.

Папка src:

Основна папка з вихідним кодом проекту. Включає компоненти, сторінки, конфігурації та інші файли, необхідні для роботи програми.

Папка src організована таким чином:

- components: містить React-компоненти, які використовуються на різних сторінках додатка.
- pages: містить компоненти сторінок, такі як Home і Customizer.
- canvas: містить компоненти та конфігурації, пов’язані з рендерингом 3D-графіки.
- store: містить файли управління станом додатка, зокрема конфігурації valtio для зберігання глобального стану.
- config: містить файли конфігурації для анімацій та інших налаштувань.

Файл **App.jsx** є центральним компонентом додатка, який об'єднує всі інші компоненти та відповідає за рендеринг основної структури веб-сторінки.

3.2 ФУНКЦІОНАЛЬНІ МОЖЛИВОСТІ

3.2.1 ГОЛОВНА СТОРІНКА

Головна сторінка нашого веб-додатка є першим екраном, який бачить користувач при відвідуванні сайту. Вона виконує декілька важливих функцій, спрямованих на залучення користувача та надання йому зрозумілого і привабливого інтерфейсу про створення дизайну для взаємодії з додатком (рис. 3.1).

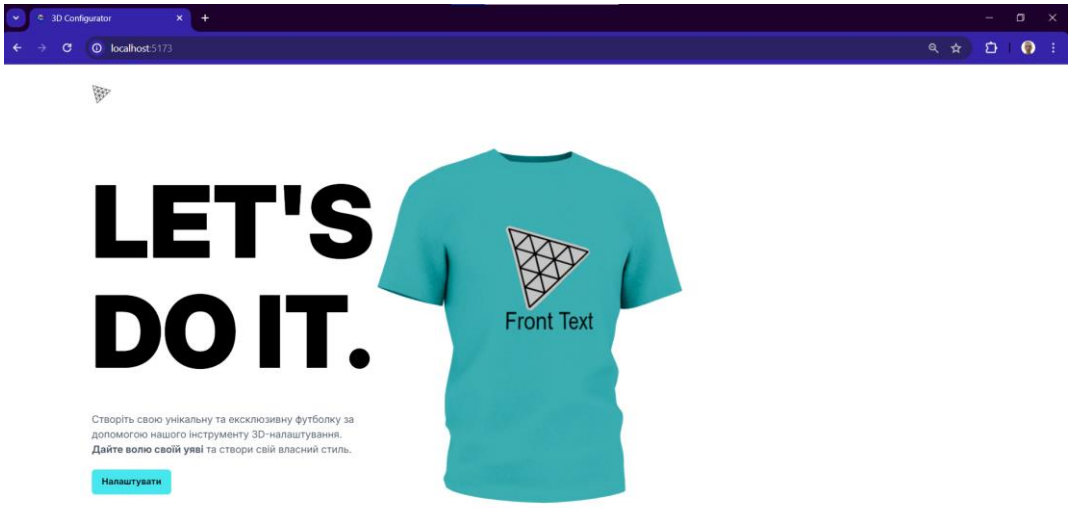

Рисунок 3.1 – Головна сторінка сайту

Основні функціональні можливості:

Головна сторінка використовує анімації для привітання користувача та створення динамічного враження. Анімації додають візуальної привабливості та допомагають направляти увагу користувача до ключових елементів сторінки.

У верхній частині сторінки розташований логотип додатка. Це допомагає у встановленні бренду та створює професійний вигляд. Нижче наведено код додавання логотипу на основну сторінку.

```
<motion.header {...slideAnimation("")}>
        <img
          src='./logo1.png'
          alt="Company logo"
          className="w-8 h-8 object-contain"
        />
```

```
</motion.header>
```

Головний заголовок привертає увагу та мотивує користувача почати налаштовувати свою 3D-футболку. Під заголовком розташований короткий опис, який пояснює можливості додатка та закликає користувача до дії.

Перша сторінка містить велику кнопку, яка запрошує користувача почати налаштовувати свою футболку. Натискання на цю кнопку переносить користувача до наступного етапу - налаштування футболки. Приклад коду кнопки.

```
<button
 className={`px-2 py-1.5 flex-1 rounded-md ${customStyles}`}
 style={generateStyle(type)}
 onClick={handleClick}
>
 {title}
</button>
```

Позаду всього тексті і по центрі сторінки розміщене велике фото футболки з якої користувач починає звій дизайн.

Коли користувач натискає кнопку "Налаштувати", він плавно переходить до інтерфейсу налаштування 3D-футболки. Це забезпечує безперервний користувацький досвід та зменшує ймовірність розгубленості.

Головна сторінка створює перше враження про додаток. Її дизайн та функціональні можливості спрямовані на залучення користувача, створення візуальної привабливості та забезпечення простого і зрозумілого початкового етапу взаємодії. Використання анімацій та чітких закликів до дії допомагає користувачу швидко зрозуміти, що він може зробити в додатку, і стимулює його до початку процесу налаштування своєї унікальної 3D-футболки.

3.2 СТОРІНКА З НАЛАШТУВАННЯМИ

Сторінка налаштування дизайну футболок є ключовим компонентом веб-додатку, що дозволяє користувачам створювати унікальні дизайни для футболок за допомогою інтерактивного 3D-редактора. Користувач переходить на сторінку налаштувань після натискання на кнопку «Налаштувати» на головні сторінці сайту. На цій сторінці користувачі можуть змінювати колір, додавати текстури, завантажувати власні зображення та накладати текст на 3D-модель футболки в режимі реального часу (рис. 3.2).

Рисунок 3.2 – Сторінка з налаштуваннями

Сторінка налаштування дизайну складається з кількох основних компонентів:

1. *Поле з 3D-оглядом (Canvas):*

Поле з 3D-оглядом (рис. 3.3) є центральним елементом сторінки налаштування дизайну футболок, де користувач може бачити результати своїх дій в режимі реального часу. Цей компонент реалізований за допомогою React Three Fiber, що дозволяє інтегрувати потужності Three.js в екосистему React.

Рисунок 3.3 – Оглядова область

Основні компоненти та їх функції:

Canvas:

Опис: Canvas є контейнером для відображення 3D-сцени на оглядовій області для користувача.

Реалізація: Створюється за допомогою компонента <Canvas> з бібліотеки React Three Fiber. Він обгортає всі 3D-елементи, такі як моделі, камери, світло тощо.

```
<Canvas
  ...
>
  <ambientLight intensity={0.6} />
  <CameraRig>
    <Center>
      <Shirt />
    </Center>
  </CameraRig>
</Canvas>
```

3D-модель футболки:

Опис: Центральний об'єкт у Canvas, який користувач може змінювати та налаштовувати.

Реалізація: Модель імпортується у форматі glb і додається до Canvas.

const { nodes, materials } = useGLTF('/shirtopen.glb');

Світло:

Опис: Джерела світла, які забезпечують правильне освітлення 3D-моделі.

Реалізація: Використовуються тип джерела світла, такий як ambient для створення реалістичних ефектів.

```
<ambientLight intensity={0.6} />
```

Камера:

Опис: Камера, що дозволяє користувачу бачити 3D-сцену під різними кутами.

Реалізація: Налаштовується за стандартного camera елемента Three.js.

```
const CustomCameraRig = ({ children }) => {
  const { camera } = useThree();
  useEffect(() => {
    ...
}, [camera]);
  return <>{children}</>;
};
```

2. *Панель інструментів для дизайну зліва сторінки:*

Панель інструментів зліва надає користувачам можливість змінювати різні аспекти 3D-моделі футболки (рис. 3.4).

Рисунок 3.4 – Панель інструментів для дизайну

Вона включає такі функціональні можливості:

- Зміна кольору футболки
- Додавання користувацького фото на футболку
- Редагування логотипа
- Редагування тексту
- Додавання текстур

Детально пройдемось по кожному пункті.

Зміна кольору футболки:

- Функціональність: Дозволяє користувачам вибирати різні кольори для футболки з допомогою кольорової палітри.
- Реалізація: Використовується кольоровий пікер, який змінює колір матеріалу на 3D-моделі футболки в режимі реального часу. Та реалізовано наступною функцією компонента.

```
return (
   <div className="fixed left-full ml-3">
    <SketchPicker
     color={snap.color}
     disableAlpha
     onChange={(color) => state.color = color.hex}
    />
   </div>
 )
```

- Інтерфейс: Кольорова палітра, де користувач може вибрати бажаний колір (рис. 3.5).

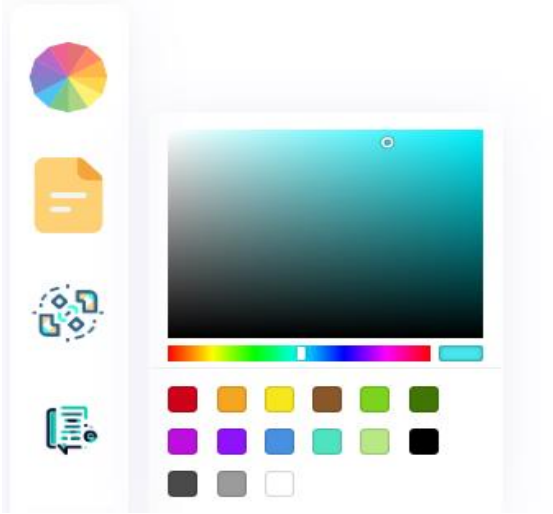

Рисунок 3.5 – Інтерфейс вікна кнопки зміни кольору

Додавання користувацького фото на футболку:

- Функціональність: Дозволяє користувачам завантажити своє власне зображення та розмістити його на футболці.
- Реалізація: Інтерфейс для завантаження файлів та механізм для відображення зображення на текстурі футболки.
- Інтерфейс: Кнопка для завантаження зображення (рис. 3.6).

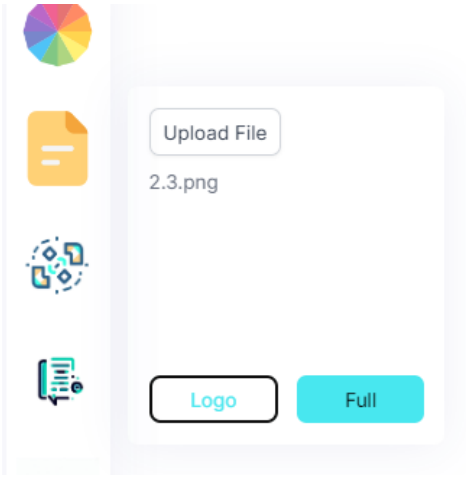

Рисунок 3.6 – Інтерфейс вікна кнопки додавання фото

Редагування логотипа:

- Функціональність: Дозволяє користувачам вибирати та редагувати логотипи, які будуть додані на футболку.
- Реалізація: Інтерфейс для вибору логотипів з бібліотеки та можливість зміни їх розміру та позиції. Розглянемо реалізацію на прикладі “FX: -“.

```
<span className="text-gray-700">FX:</span>
        <button
          className="border border-gray-400 rounded-md p-2"
          onClick={() => handlePositionChange('front', 0, snap.frontLogoPosition[0] - 0.01)}
        > -
        </button>
```

- Інтерфейс: Список доступних логотипів та опції для налаштування їх параметрів (рис. 3.7).

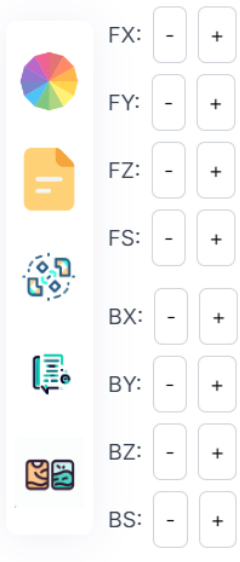

Рисунок 3.7 – Інтерфейс вікна кнопки редагування логотипу

Редагування тексту:

- Функціональність: Дозволяє користувачам додавати та налаштовувати текст на футболці спереду та ззаду.
- Реалізація: Поле для введення тексту та параметри для зміни шрифту, розміру та кольору тексту.
- Інтерфейс: Текстове поле та панель налаштувань тексту (рис. 3.8).

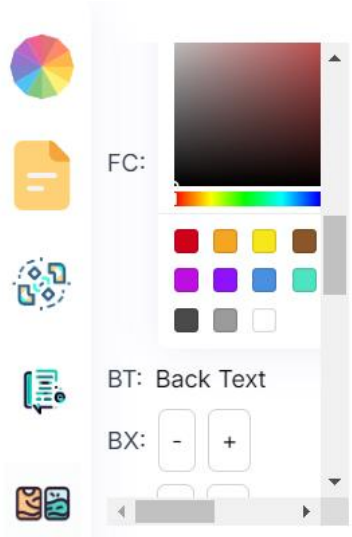

Рисунок 3.8 – Інтерфейс вікна кнопки редагування тексту

Додавання текстур:

- Функціональність: Дозволяє користувачам вибирати та додавати текстури до поверхні футболки.

- Реалізація: Інтерфейс для вибору текстур та механізм для застосування текстур до матеріалу футболки. Додавання текстур і не тільки здійснюється за допомогою наступної функції.

```
const displayImages = (images) => {
  return (
    <div className='grid grid-cols-2 gap-2'>
      {images.map((img, index) => (
        <div key={img.id} onClick={() => handleImageClick(img)}>
          <img src={img.url} alt={img.name} className='rounded-lg w-full' />
        </div>
      ))}
    </div>
  );
};
```

- Інтерфейс: Список текстур.

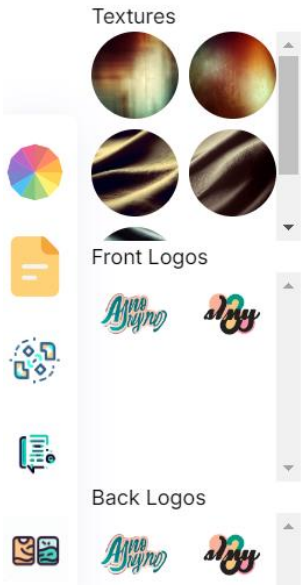

Рисунок 3.9 – Інтерфейс вікна кнопки додавання текстур

Користувацький інтерфейс панелі інструментів організований таким чином, щоб забезпечити легкий доступ до всіх функцій налаштування. Кожен інструмент має чітке та інтуїтивне розташування, що дозволяє користувачам швидко та легко вносити зміни до дизайну футболки.

Таким чином, панель інструментів зліва є ключовим елементом сторінки налаштувань, що забезпечує користувачам усі необхідні можливості для повної персоналізації своєї 3D-моделі футболки.

3. *Панель інструментів внизу сторінки:*

Панель інструментів знизу надає користувачам можливість додаткових налаштувань футболки (рис. 3.10).

Рисунок 3.10 – Панель інструментів внизу сторінки

- Додавання або видалення логотипа спереду футболки
- Додавання або видалення логотипа ззаду футболки
- Додавання або видалення тексту спереду футболки
- Додавання або видалення тексту ззаду футболки
- Додавання або видалення текстури футболки
- Збереження результату дизайну на пристрій користувача

Розглянемо кожну кнопку детально.

Додавання або видалення логотипа спереду футболки:

- Функціональність: Дозволяє користувачам додавати логотип на передню частину футболки або видаляти його.
- Інтерфейс: Кнопка з виділенням додає логотип, при клікові ще раз по кнопці здійснюється видалення логотипа.

Додавання або видалення логотипа ззаду футболки:

- Функціональність: Дозволяє користувачам додавати логотип на задню частину футболки або видаляти його.
- Інтерфейс: Кнопка з виділенням додає логотип, при клікові ще раз по кнопці здійснюється видалення логотипа.

Додавання або видалення тексту спереду футболки:

- Функціональність: Дозволяє користувачам додавати текст на передню частину футболки або видаляти його.
- Інтерфейс: Кнопка з виділенням додає текст, при клікові ще раз по кнопці здійснюється видалення тексту.

Додавання або видалення тексту ззаду футболки:

- Функціональність: Дозволяє користувачам додавати текст на задню частину футболки або видаляти його.
- Інтерфейс: Кнопка з виділенням додає такст, при клікові ще раз по кнопці здійснюється видалення тексту.

Додавання або видалення текстури футболки:

- Функціональність: Дозволяє користувачам додавати текстуру на всю футболку або видаляти її.
- Інтерфейс: Кнопка з виділенням додає текстуру, при клікові ще раз по кнопці здійснюється видалення текстури.

Збереження результату дизайну на пристрій користувача:

- Функціональність: Дозволяє користувачам зберегти створений дизайн на свій пристрій.
- Інтерфейс: Кнопка що здійснює завантаження фото на пристрій користувача.

Інструменти, розміщені внизу сторінки, є важливим елементом інтерфейсу користувача, що забезпечує додаткові можливості для персоналізації дизайну 3D-моделі футболки. Вони дозволяють користувачам легко додавати та видаляти елементи дизайну, а також зберігати свої роботи, що робить процес налаштування більш зручним та ефективним.

4. *Кнопка повернутися назад*

Останнім важливим елементом сторінки налаштувань є кнопка "Go Back", яка розташована у верхньому правому куті. Вона забезпечує швидке і зручне повернення користувача на головну сторінку веб-сайту.

Реалізована за допомогою ідентичного компонента, що і кнопка на головній сторінці «Налаштувати» яка описана вище.

3.3 АДАПТАЦІЯ СТОРІНКИ

Адаптація сторінки є важливим аспектом розробки сучасних веб-додатків, особливо коли йдеться про додатки з інтерактивною 3D-графікою, такі як наш проект з налаштування дизайну 3D-футболок. Адаптивний дизайн забезпечує зручне використання веб-сайту на різних пристроях і екранах різних розмірів, включаючи мобільні телефони, планшети та настільні комп'ютери [8].

Реалізація адаптивного дизайну в нашому проекті:

- Гнучка сітка: У нашому проекті ми використовуємо класові утиліти Tailwind CSS, такі як flex, w-full, h-full, щоб створити гнучкий макет. Це дозволяє легко змінювати розташування елементів на сторінці відповідно до розміру екрану.
- Медіа-запити: Використовуючи класи Tailwind CSS, ми можемо легко застосовувати різні стилі для різних розмірів екрану, використовуючи префікси sm:, md:, lg:, xl:.

Особливості адаптивного дизайну в нашому проекті:

Панель інструментів: Панель інструментів зліва може зберігати свої розміри то розташовуватися по центрі вертикалі не залежно від розмірів екрана.

Панель кнопок: Кнопки для дизайну внизу сторінки підлаштовуються під ширину сторінки.

Кнопка "Go Back": Завжди знаходиться у верхньому правому куті, забезпечуючи зручну навігацію незалежно від розміру екрану.

Під час розробки ми тестували наш сайт на різних пристроях і екранах, щоб переконатися, що він виглядає та функціонує належним чином. Використання інструментів розробника в браузерах дозволило нам перевіряти адаптивність у реальному часі та вносити необхідні корективи.

Нижче наведено фото результату адаптації сторінки (рис. 3.11).

Рисунок 3.11 – Адаптація сторінки

Адаптивний дизайн є невід'ємною частиною сучасної веб-розробки, особливо для проектів, які використовують складну графіку, як наш сайт для налаштування 3D-футболок. Використання класових утиліт Tailwind CSS дозволило нам створити веб-сайт, який зручно використовувати на будь-якому пристрої, забезпечуючи користувачам високий рівень зручності та задоволення.

ВИСНОВКИ

У процесі розробки веб-сайту для дизайну 3D-футболок було досягнуто поставленої мети та необхідних результатів, які підтверджують актуальність і практичність обраної теми. В умовах сучасного ринку інтернет-торгівлі зростає попит на персоналізовані продукти, що дозволяють користувачам створювати унікальні речі. Наш проект надає таку можливість, дозволяючи створювати унікальні дизайни футболок за допомогою інтерактивного веб-додатку. Використання сучасних технологій, таких як React для побудови інтерфейсу користувача та React Three Fiber для інтеграції 3D-графіки, дозволило створити високоякісний, інтерактивний і зручний у використанні веб-додаток. Було створено логічну структуру проекту з використанням компонентного підходу, що полегшує підтримку та розширення проекту в майбутньому. Веб-сайт розроблений з урахуванням адаптивного дизайну, що забезпечує зручне використання на різних пристроях. Використання Tailwind CSS дозволило швидко та ефективно реалізувати адаптивність. Реалізовані функції включають налаштування кольору футболки, додавання користувацьких зображень і тексту, редагування логотипів і текстур, а також можливість зберігати створені дизайни на пристрої користувача. Це робить наш веб-додаток повноцінним інструментом для створення персоналізованих 3D-футболок. У результаті було створено функціональний, інтерактивний і зручний у використанні веб-сайт, який дозволяє користувачам створювати унікальні дизайни 3D-футболок, змінювати їхній вигляд, додавати графіку та тексти. Цей проект має значний потенціал для залучення широкого кола клієнтів, які цінують унікальний та персоналізований одяг, і може стати корисним інструментом для дизайнерів і підприємців, що прагнуть пропонувати своїм клієнтам інноваційні продукти та послуги.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Wilson David, Elman Jake. Web Design: The Evolution of the Digital World 1990-Today. Taschen, 2019. Pp. 149-211.
2. Bowers Steven, Vardeman Joshua. HTML5 and CSS3: Building Responsive Web Design with 3D Elements. Packt Publishing, 2016. Pp. 374-392.
3. Matsuda Kouichi, Lea Rodger. WebGL Programming Guide: Interactive 3D Graphics Programming with WebGL. Addison-Wesley, 2013. Pp. 127-149.
4. Banks Alex, Porcello Eve. Learning React: Functional Web Development with React and Redux. O'Reilly Media, 2017. Pp. 323-330.
5. React Documentation. URL: https://reactjs.org/docs/getting-started.html (дата звернення 03.06.2024).
6. Three.js Documentation. URL: https://threejs.org/docs/ (дата звернення 03.06.2024).
7. React Three Fiber Documentation. URL: https://docs.pmnd.rs/react-three-fiber/getting-started/introduction (дата звернення 03.06.2024).
8. Tailwind CSS Documentation. URL: https://tailwindcss.com/docs (дата звернення 03.06.2024).

ДОДАТКИ

Додаток А. Фрагмент коду з файлу App.jsx

```
import Canvas from './canvas';
import Customizer from './pages/Customizer';
import Home from './pages/Home';
function App() {
  return (
    <main className="app transition-all ease-in">
      <Home />
      <Canvas />
      <Customizer />
    </main>
  )
}
export default App
```

Додаток Б. Фрагмент коду з файлу Home.jsx

```
import { motion, AnimatePresence } from 'framer-motion';
import { useSnapshot } from 'valtio';
import state from '../store';
import { CustomButton } from '../components';
import {
  headContainerAnimation,
  headContentAnimation,
  headTextAnimation,
  slideAnimation
} from '../config/motion';
const Home = () => {
  const snap = useSnapshot(state);
  return (
    <AnimatePresence>
      {snap.intro && (
        <motion.section className="home" {...slideAnimation('left')}>
          <motion.header {...slideAnimation("down")}>
            <img
              src='./threejs.png'
              alt="logo"
              className="w-8 h-8 object-contain"
            />
          </motion.header>
          <motion.div                                    className="home-content"
{...headContainerAnimation}>
            <motion.div {...headTextAnimation}>
              <h1 className="head-text">
                LET'S <br className="xl:block hidden" /> DO IT.
              </h1>
```

```
            </motion.div>
            <motion.div
              {...headContentAnimation}
              className="flex flex-col gap-5"
            >
              <p className="max-w-md font-normal text-gray-600 text-base">
              Створіть свою унікальну та ексклюзивну футболку за допомогою нашого інструменту 3D-налаштування. <strong>Дайте волю своїй уяві</strong>{" "} та створи свій власний стиль.
              </p>
              <CustomButton
                type="filled"
                title="Налаштувати"
                handleClick={() => state.intro = false}
                customStyles="w-fit px-4 py-2.5 font-bold text-sm"
              />
            </motion.div>
          </motion.div>
        </motion.section>
      )}
    </AnimatePresence>
  )
}
export default Home
```

Додаток В. Фрагмент коду з файлу Customizer.jsx

```
import React, { useState, useEffect } from 'react';
import { AnimatePresence, motion } from 'framer-motion';
import { useSnapshot } from 'valtio';
import LogoControls from '../canvas/LogoControls';
import TextControls from '../canvas/TextControls';
import state from '../store';
import { downloadCanvasToImage, reader } from '../config/helpers';
import { EditorTabs, FilterTabs, DecalTypes, texturesLogos } from '../config/constants';
import { fadeAnimation, slideAnimation } from '../config/motion';
import { ColorPicker, CustomButton, FilePicker, TextureLogoPicker, Tab } from '../components';

const Customizer = () => {
  const snap = useSnapshot(state);
  const [file, setFile] = useState('');

  const [activeEditorTab, setActiveEditorTab] = useState("");
  const [activeFilterTab, setActiveFilterTab] = useState({
    frontLogoShirt: true,
    backLogoShirt: true,
    frontTextShirt: true,
    backTextShirt: true,
    stylishShirt: false,
  })
  // show tab content depending on the activeTab
  const generateTabContent = () => {
    switch (activeEditorTab) {
      case "colorpicker":
```

```
      return <ColorPicker />
    case "filepicker":
      return <FilePicker
        file={file}
        setFile={setFile}
        readFile={readFile}
      />
    case "logocontrols":
      return <LogoControls />;
    case "textcontrols":
      return <TextControls />;
    case "texturelogopicker":
      return (
        <TextureLogoPicker
          texturesLogos={texturesLogos}
          handleTextureLogoClick={handleTextureLogoClick}
        />
      );
    default:
      return null;
  }
}
const handleTextureLogoClick = (textureLogo) => {
  // update the state with the selected texture or logo
  if (textureLogo.type === "texture") {
    // update the state with the selected texture
    state.fullDecal = textureLogo.image;
  } else if (textureLogo.type === "frontLogo") {
    // update the state with the selected logo
    state.frontLogoDecal = textureLogo.image;
```

```
  } else if (textureLogo.type === "backLogo") {
    // update the state with the selected logo
    state.backLogoDecal = textureLogo.image
  }
};
const handleDecals = (type, result) => {
  const decalType = DecalTypes[type];
  state[decalType.stateProperty] = result;
  if(!activeFilterTab[decalType.filterTab]) {
    handleActiveFilterTab(decalType.filterTab)
  }
}
const handleActiveFilterTab = (tabName) => {
  switch (tabName) {
    case "frontLogoShirt":
        state.isFrontLogoTexture = !activeFilterTab[tabName];
      break;
    case "backLogoShirt":
        state.isBackLogoTexture = !activeFilterTab[tabName];
      break;
    case "frontTextShirt":
        state.isFrontText = !activeFilterTab[tabName];
      break;
    case "backTextShirt":
        state.isBackText = !activeFilterTab[tabName];
      break;
    case "stylishShirt":
        state.isFullTexture = !activeFilterTab[tabName];
      break;
      case "downloadShirt":
```

```
        downloadCanvasToImage();
      break;
    default:
      state.isFrontLogoTexture = true;
      state.isBackLogoTexture = true;
      state.isFrontText = true;
      state.isBackText = true;
      state.isFullTexture = false;
      break;
  }
  // after setting the state, activeFilterTab is updated
  setActiveFilterTab((prevState) => {
    return {
      ...prevState,
      [tabName]: !prevState[tabName]
    }
  })
}
const readFile = (type) => {
  reader(file)
    .then((result) => {
      handleDecals(type, result);
      setActiveEditorTab("");
    })
}
return (
  <AnimatePresence>
    {!snap.intro && (
      <>
        <motion.div
```

```
  key="custom"
  className="absolute top-0 left-0 z-10"
  {...slideAnimation('left')}
>
  <div className="flex items-center min-h-screen">
    <div className="editortabs-container tabs">
      {EditorTabs.map((tab) => (
        <Tab
          key={tab.name}
          tab={tab}
          handleClick={() => setActiveEditorTab(tab.name)}
        />
      ))}
      {generateTabContent()}
    </div>
  </div>
</motion.div>

<motion.div
  className="absolute z-10 top-5 right-5"
  {...fadeAnimation}
>
  <CustomButton
    type="filled"
    title="Go Back"
    handleClick={() => state.intro = true}
    customStyles="w-fit px-4 py-2.5 font-bold text-sm"
  />
</motion.div>
<motion.div
```

```
          className='filtertabs-container'
          {...slideAnimation("up")}
        >
          {FilterTabs.map((tab) => (
            <Tab
              key={tab.name}
              tab={tab}
              isFilterTab
              isActiveTab={activeFilterTab[tab.name]}
              handleClick={() => handleActiveFilterTab(tab.name)}
            />
          ))}
        </motion.div>
      </>
    )}
  </AnimatePresence>
 )
}
export default Customizer
```

Додаток Г. Фрагмент коду з файлу FilePicker.jsx

```
import React from 'react'
import CustomButton from './CustomButton'
const FilePicker = ({ file, setFile, readFile }) => {
  return (
    <div className="filepicker-container">
      <div className="flex-1 flex flex-col">
        <input
          id="file-upload"
          type="file"
          accept="image/*"
          onChange={(e) => setFile(e.target.files[0])}
        />
        <label htmlFor="file-upload" className="filepicker-label">
          Upload File
        </label>
        <p className="mt-2 text-gray-500 text-xs truncate">
          {file === '' ? "No file selected" : file.name}
        </p>
      </div>
      <div className="mt-4 flex flex-wrap gap-3">
        <CustomButton
          type="outline"
          title="Logo"
          handleClick={() => readFile('logo')}
          customStyles="text-xs"
        />
        <CustomButton
          type="filled"
          title="Full"
```

```
          handleClick={() => readFile('full')}
          customStyles="text-xs"
        />
      </div>
    </div>
  )
}
export default FilePicker
```

Додаток Ґ. Фрагмент коду з файлу Tab.jsx

```
import React from 'react'
import { useSnapshot } from 'valtio'
import state from '../store';
const Tab = ({ tab, isFilterTab, isActiveTab, handleClick }) => {
  const snap = useSnapshot(state);
  const activeStyles = isFilterTab && isActiveTab
    ? { backgroundColor: snap.color, opacity: 0.5 }
    : { backgroundColor: "transparent", opacity: 1 }
  return (
    <div
      key={tab.name}
      className={`tab-btn ${isFilterTab ? 'rounded-full glassmorphism' : 'rounded-
4'}`}
      onClick={handleClick}
      style={activeStyles}
    >
      <img
        src={tab.icon}
        alt={tab.name}
        className={`${isFilterTab ? 'w-2/3 h-2/3' : 'w-11/12 h-11/12 object-
contain'}`}
      />
    </div>
  )
}
export default Tab
```