МИНИСТЕРСТВО ОБРАЗОВАНИЯ НОВОСИБИРСКОЙ ОБЛАСТИ  
ГБПОУ НСО «НОВОСИБИРСКИЙ ПОЛИТЕХНИЧЕСКИЙ КОЛЛЕДЖ»

К ЗАЩИТЕ ДОПУЩЕН

Зам. директора по УМР

\_\_\_\_\_\_\_\_\_ А.В. Васильева

«\_\_»\_\_\_\_\_\_\_\_\_\_2025г.

**ДИПЛОМНЫЙ ПРОЕКТ**

Разработка интерфейса веб сервиса для реализации маркетинговых 3D-моделей

Выполнил студент

Петухов С. А.

№ группы 121 ИСП

Руководитель: Картавов С. Д.

Новосибирск

2025

Содержание

[Перечень условных обозначений, терминов 4](#_Toc200671153)

[Введение 7](#_Toc200671154)

[1. Теоретическая часть 9](#_Toc200671155)

[1.1 3D Модель 9](#_Toc200671156)

[1.1.1 3D-моделирование в контексте веб-технологий 9](#_Toc200671157)

[1.1.2 Методы оптимизации для веб-среды 9](#_Toc200671158)

[1.2 Текстуры 14](#_Toc200671159)

[1.2.1 Текстурирование в контексте современных 14](#_Toc200671160)

[1.2.2 Ключевые аспекты: 14](#_Toc200671161)

[1.2.3 PBR-рендеринг: от теории к практике 15](#_Toc200671162)

[1.3 Шейдеры 19](#_Toc200671163)

[1.3.1 Исторический контекст 19](#_Toc200671164)

[1.3.2 Типы шейдеров и их функции 19](#_Toc200671165)

[1.3.3 Шейдеры и PBR: симбиоз физики и оптимизации 21](#_Toc200671166)

[1.3.4 Оптимизации для веба: 21](#_Toc200671167)

[1.3.5 Инновации и тренды 21](#_Toc200671168)

[1.4 Рендеринг 23](#_Toc200671169)

[1.4.1 Методы рендеринга 23](#_Toc200671170)

[1.4.2 Этапы рендеринга 25](#_Toc200671171)

[1.4.3 Оптимизация рендеринга для веба 28](#_Toc200671172)

[1.5 Методологические аспекты разработки веб-сервиса 30](#_Toc200671173)

[1.5.1 Выбор средств разработки веб-сервиса 30](#_Toc200671174)

[1.5.2 Определение требований к разрабатываемому продукту 31](#_Toc200671175)

[1.5.3 Определение целевой аудитории и анализ конкурентов 31](#_Toc200671176)

[2. Практическая часть 33](#_Toc200671177)

[2.1 Структура разработки 33](#_Toc200671178)

[2.2 Разработка сайта 34](#_Toc200671179)

[2.2.1 Разработка редактора 34](#_Toc200671180)

[2.2.2 Разработка viewer моделей 37](#_Toc200671181)

[2.2.3 Разработка галереи моделей 38](#_Toc200671182)

[2.3 Тестирование сайта 40](#_Toc200671183)

[2.4 Информационная безопасность 42](#_Toc200671184)

[3.Экономическая часть 43](#_Toc200671185)

[3.1 Организация и экономика производства 43](#_Toc200671186)

[3.2 Экономическая эффективность продукта 46](#_Toc200671187)

[Заключение 48](#_Toc200671188)

[Список используемой литературы 50](#_Toc200671189)

# Перечень условных обозначений, терминов

**3D** — трёхмерный (Three-Dimensional).

**2D** — двумерный (Two-Dimensional).

**PBR** — Physically Based Rendering (физически корректный рендеринг).

**UV** — координаты текстуры на 2D-плоскости (U и V — оси текстуры).

**FBX** — формат файлов для хранения 3D-моделей, разработанный Autodesk.

**JPEG** — формат изображений с потерями (Joint Photographic Experts Group).

**JSON** — текстовый формат обмена данными (JavaScript Object Notation).

**WebGL** — технология для отрисовки 3D-графики в браузере (Web Graphics Library).

**API** — интерфейс программирования приложений (Application Programming Interface).

**GPU** — графический процессор (Graphics Processing Unit).

**LOD** — Level of Detail (уровень детализации).

**OBJ** — формат файлов для хранения 3D-моделей.

**GLTF** — формат файлов для хранения 3D-сцен и моделей (GL Transmission Format).

**STL** — формат файлов для хранения 3D-моделей, используемый в 3D-печати (Stereolithography).

**HTML** — язык разметки гипертекста (HyperText Markup Language).

**CSS** — каскадные таблицы стилей (Cascading Style Sheets).

**JS** — JavaScript.

**Albedo** — текстура, определяющая базовый цвет поверхности.

**Normal Map** — карта нормалей, используемая для создания иллюзии рельефа.

**Roughness Map** — карта шероховатости, определяющая, насколько поверхность отражает свет.

**Metallic Map** — карта металличности, указывающая, какие части поверхности являются металлическими

**3D-модель** — цифровое представление трёхмерного объекта, состоящее из вершин, рёбер и граней.

**Текстура** — изображение, накладываемое на поверхность 3D-модели для придания детализации.

**Шейдер** — программа, выполняемая на GPU, которая определяет, как отображаются пиксели на экране.

**Рендеринг** — процесс преобразования 3D-модели в 2D-изображение.

**Вершина** — точка в 3D-пространстве, определяющая форму модели.

**Полигон** — многоугольник, образованный вершинами (обычно треугольник или четырёхугольник).

**UV-развёртка** — процесс проекции 2D-текстуры на поверхность 3D-модели.

**Шов** — место разреза на 3D-модели, необходимое для её развёртки на 2D-плоскость.

**Интерактивность** — возможность взаимодействия пользователя с 3D-моделью (например, вращение, масштабирование).

**Камера** — виртуальная точка обзора в 3D-сцене, определяющая, как модель отображается на экране.

**Освещение** — имитация источников света в 3D-сцене, влияющая на отображение модели.

**Материал** — набор свойств поверхности модели, таких как цвет, текстура, отражательная способность.

**Анимация** — изменение положения или формы модели во времени.

**Фронтенд** — клиентская часть веб-приложения, отвечающая за отображение интерфейса.

**Бэкенд** — серверная часть веб-приложения, отвечающая за обработку данных и логику.

**База данных** — структурированное хранилище данных, используемое для сохранения информации.

**Регистрация** — процесс создания учётной записи пользователя.

**Авторизация** — процесс проверки прав доступа пользователя к системе.

**Виджет** — интерактивный элемент, встраиваемый на веб-страницу.

**Конструктор** — инструмент для создания и настройки 3D-виджетов.

**Оптимизация** — процесс улучшения производительности и уменьшения нагрузки на систему.

**Совместимость** — способность сервиса работать на различных устройствах и браузерах.

# ****Введение****

Современный мир цифровых технологий стремительно развивается, и с каждым годом требования пользователей к онлайн-сервисам становятся всё выше. Одним из ключевых аспектов успешного взаимодействия с клиентом является визуализация продукта. Потребители хотят не только прочитать описание товара, но и увидеть его со всех сторон, изучить детали и представить, как он будет выглядеть в реальной жизни. Это особенно актуально для таких сфер, как электронная коммерция, строительство, дизайн интерьеров, автомобильная промышленность и многих других, где важно продемонстрировать продукт в максимально реалистичном виде.

Традиционные способы визуализации, такие как статичные изображения или даже 360-градусные фотографии, уже не всегда удовлетворяют запросы пользователей. На смену им приходят интерактивные 3D-модели, которые позволяют не только рассмотреть объект со всех сторон, но и взаимодействовать с ним: изменять цвет, текстуру, масштаб или даже визуализировать различные конфигурации продукта. Такие технологии значительно повышают вовлечённость пользователей, улучшают их опыт взаимодействия с сайтом и, как следствие, увеличивают конверсию.

В связи с этим актуальной задачей становится разработка инструментов, которые позволят легко интегрировать интерактивные 3D-виджеты на сайты компаний. Однако, несмотря на растущий спрос на подобные решения, многие веб-разработчики и владельцы бизнеса сталкиваются с рядом проблем. Среди них — сложность интеграции 3D-моделей в существующие платформы, высокая стоимость разработки индивидуальных решений, а также необходимость обладания специализированными знаниями в области 3D-графики и веб-программирования.

Целью данного дипломного проекта является создание веб-сервиса, который упростит процесс внедрения интерактивных 3D-виджетов на сайты клиентов. Сервис должен быть доступным для пользователей с разным уровнем технической подготовки, обеспечивать высокую производительность и совместимость с различными платформами и устройствами. Кроме того, важно предусмотреть возможность кастомизации виджетов под нужды конкретного бизнеса, чтобы каждый клиент мог адаптировать решение под свои задачи.

В рамках проекта будут рассмотрены современные технологии и инструменты для работы с 3D-графикой в вебе, такие как WebGL, Three.js, Babylon.js и другие. Также будет проведён анализ существующих решений на рынке, их преимуществ и недостатков. На основе этого анализа будет разработана архитектура сервиса, который позволит клиентам загружать 3D-модели, настраивать их отображение и легко встраивать на свои сайты с помощью готовых виджетов.

Практическая значимость проекта заключается в том, что он предоставит бизнесам доступный и удобный инструмент для улучшения визуальной презентации продуктов, что может стать конкурентным преимуществом в условиях насыщенного рынка. Для разработчиков такой сервис упростит процесс интеграции 3D-графики, сократит время и затраты на реализацию подобных решений.

В заключение, данный проект направлен на создание инновационного продукта, который сможет удовлетворить растущий спрос на интерактивные 3D-технологии в вебе. Реализация такого сервиса не только повысит качество взаимодействия пользователей с онлайн-платформами, но и откроет новые возможности для бизнеса в области цифровой визуализации.

Цельпроекта – это реализация веб сервиса для улучшения пользовательского опыта, а также развития компьютерной графики в вебе.

# **1. Теоретическая** часть

## 1.1 3D Модель

3D-модель — это цифровое представление трёхмерного объекта, созданное с помощью специализированного программного обеспечения.

1.1.1 3D-моделирование в контексте веб-технологий  
Основу модели формирует полигональная сетка — совокупность вершин, рёбер и граней, определяющих геометрию объекта. В зависимости от задач используются:

* Треугольные полигоны (tris) — стандарт для игровых движков благодаря их аппаратной оптимизации.
* Четырёхугольные полигоны (quads) — предпочтительны в моделировании для упрощения редактирования топологии.
* N-gons — полигоны с пятью и более сторонами, которые могут вызывать артефакты при рендеринге.

Для сложных объектов применяются иерархические структуры, такие как скелетная анимация и системы уровней детализации. Для веб-приложений важно, чтобы модели были оптимизированы для быстрой загрузки и отображения в браузере.

### 1.1.2 Методы оптимизации для веб-среды

Эффективная интеграция 3D-моделей в веб-приложения требует соблюдения баланса между визуальным качеством и производительностью. Ключевые аспекты:

1. Полигональная оптимизация
   * Ретопология — перестройка сетки для снижения числа полигонов при сохранении формы.
   * Децимация — алгоритмическое упрощение геометрии (например, инструмент Quadric Edge Collapse Decimation в Blender).[[4](#_Список_используемой_литературы)]
   * Нормализация сглаживания — использование карт нормалей (normal maps) для имитации деталей без увеличения полигонов.

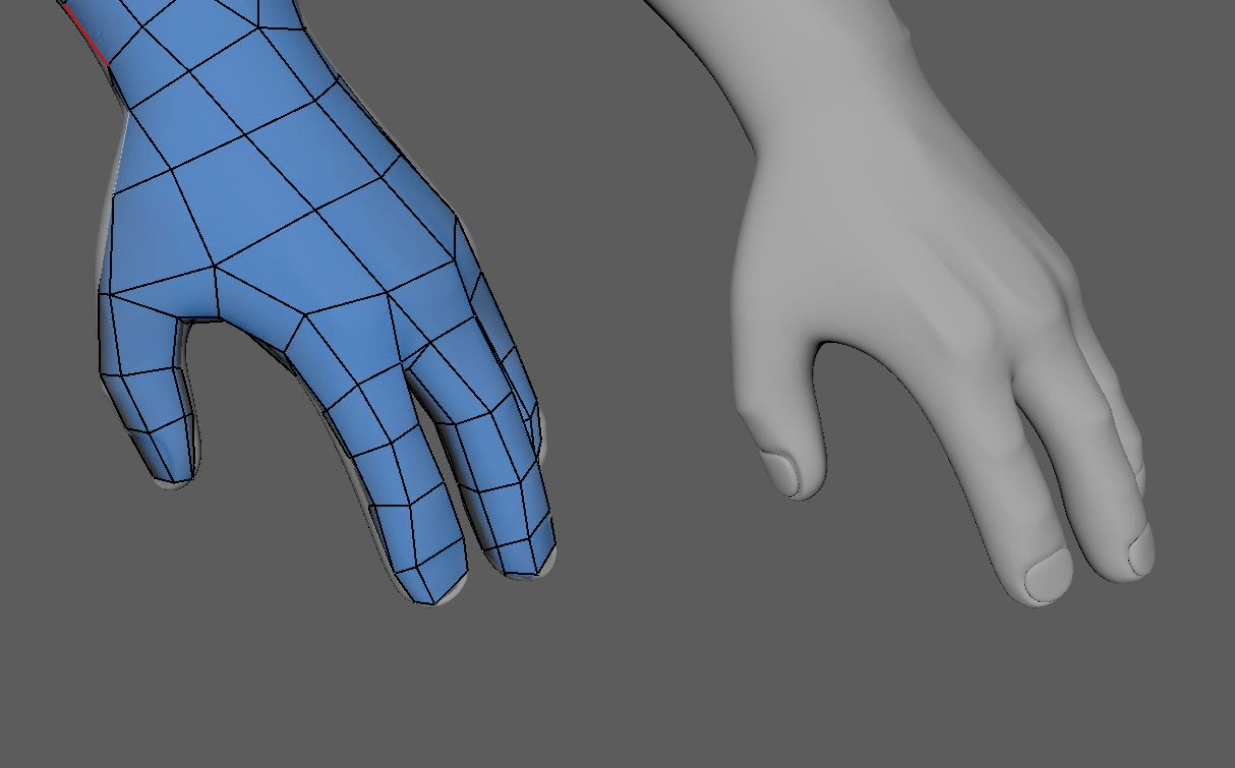


Рисунок 1. Ретопология

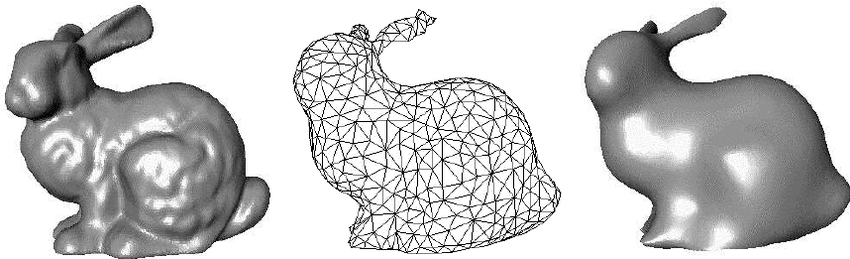


Рисунок 2. Децимация



Рисунок 3. Нормализация сглаживания

1. Работа с текстурами
   * Атласирование — объединение текстур в единую карту для минимизации HTTP-запросов.
   * Сжатие — использование форматов Basis Universal, KTX2 или ASTC, поддерживающих аппаратное ускорение.
   * PBR-материалы — физически корректные материалы (Metallic-Roughness workflow) для реалистичного рендеринга в WebGL.



Рисунок 4. пример текстурного атласа

1. Форматы данных
   * glTF (GL Transmission Format) — открытый стандарт, поддерживающий меши, материалы, анимации и скелеты. Оптимизирован для веба благодаря бинарному сжатию (.glb) и разделению ресурсов (.gltf + .bin + текстуры).
   * FBX — проприетарный формат, используемый преимущественно на этапе разработки, но не рекомендованный для финальной веб-интеграции из-за большого размера.

Особенности моделей для веба включают:

Оптимизация: Модели должны быть легковесными, чтобы минимизировать время загрузки.

Совместимость: Форматы файлов должны поддерживаться веб-браузерами и библиотеками, такими как Three.js.

Интерактивность: Модели должны быть готовы к взаимодействию с пользователем, например, к изменению ракурса, масштабированию или настройке материалов.

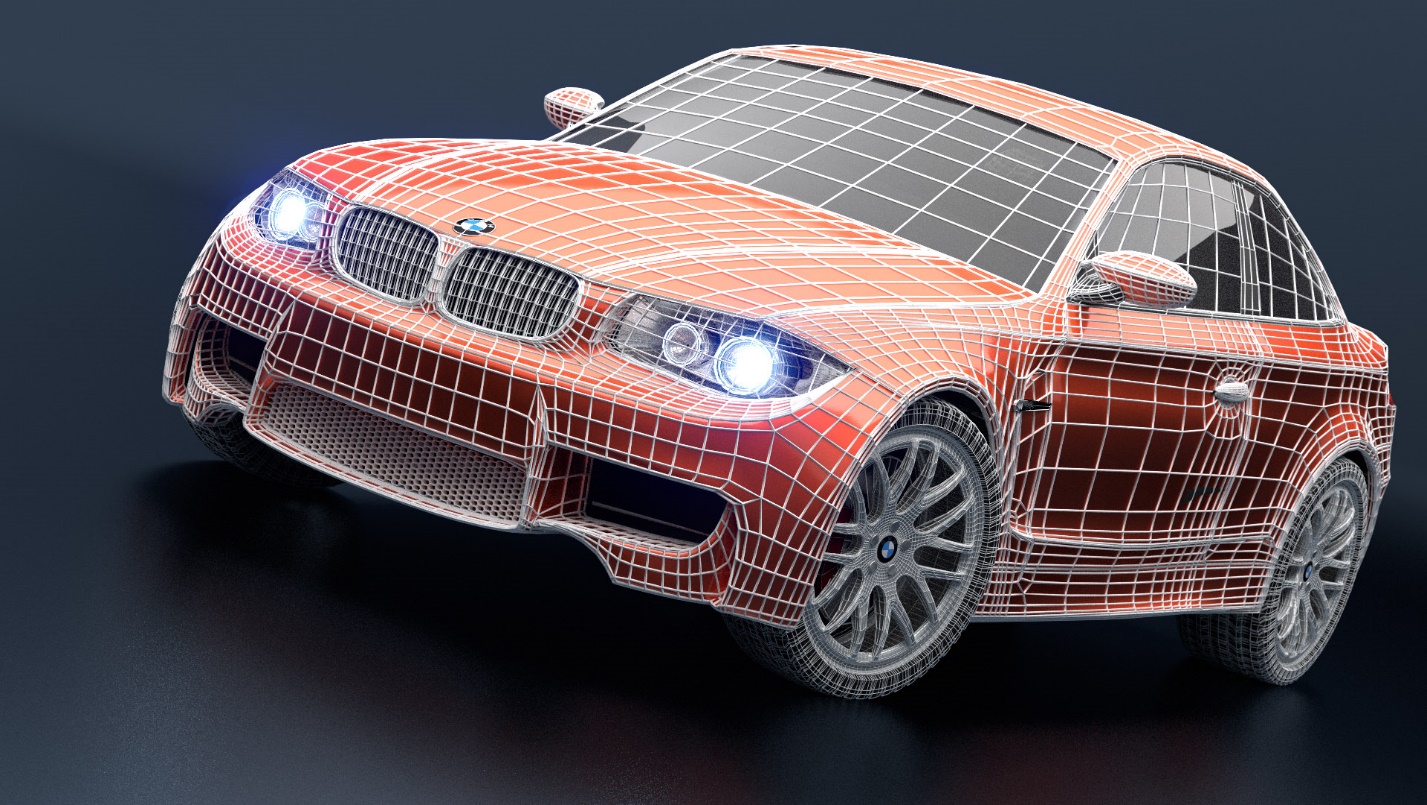


Рисунок 5. 3D Модель с полигональной сеткой

## 1.2 Текстуры

Текстуры — это изображения, которые накладываются на поверхность 3D-модели при помощи UV-развертки для придания ей детализации и реалистичности. Они могут включать цвет, рельеф, отражения и другие свойства поверхности.

1.2.1 Текстурирование в контексте современных веб-технологий  
В отличие от простой раскраски, современное текстурирование представляет собой многоуровневый процесс, объединяющий математические модели, алгоритмы сэмплинга и физически корректное взаимодействие со светом. Для веб-приложений этот процесс требует особой оптимизации без ущерба для реализма.

UV-развёртка: методы и вызовы

UV-развёртка — процесс проекции 2D-текстуры на 3D-модель через параметризацию поверхности. Это цифровой аналог "разрезания" объекта по швам для его плоского представления.

### 1.2.2 Ключевые аспекты:

1. Методы генерации:

Автоматическая развёртка (Blender’s Smart UV Project, Unwrap) — быстрая, но часто требует ручной коррекции.

Ручное рассечение — выделение швов вдоль естественных граней модели (например, края одежды).

UDIM — стандарт для кинопроизводства, использующий тайловую систему (каждая UV-область — отдельный файл, например, «1001.udim»).

1. Проблемы и решения:

Растяжение (Stretching): возникает при неравномерном распределении UV-плотности. Лечится инструментами Texel Density (пикселей на единицу модели).

Наложение (Overlapping): необходимо для симметричных объектов, но требует уникализации через UDIM-смещения.

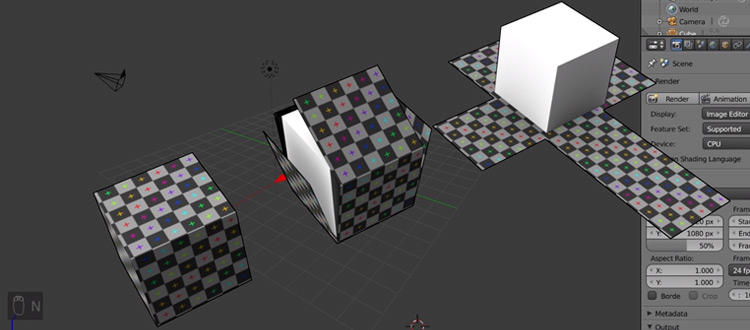
Оптимизация для веба: Использование атласов текстур — объединение UV-островов в единую карту для минимизации draw calls.

Рисунок 6. Принцип работы UV-Развертки

### 1.2.3 PBR-рендеринг: от теории к практике

Physically Based Rendering[[1](#_Список_используемой_литературы)] — стандарт индустрии, основанный на уравнениях микроповерхностей (Cook-Torrance, GGX). Включает два основных подхода:

1. Металличность-Шероховатость (Metalness-Roughness):

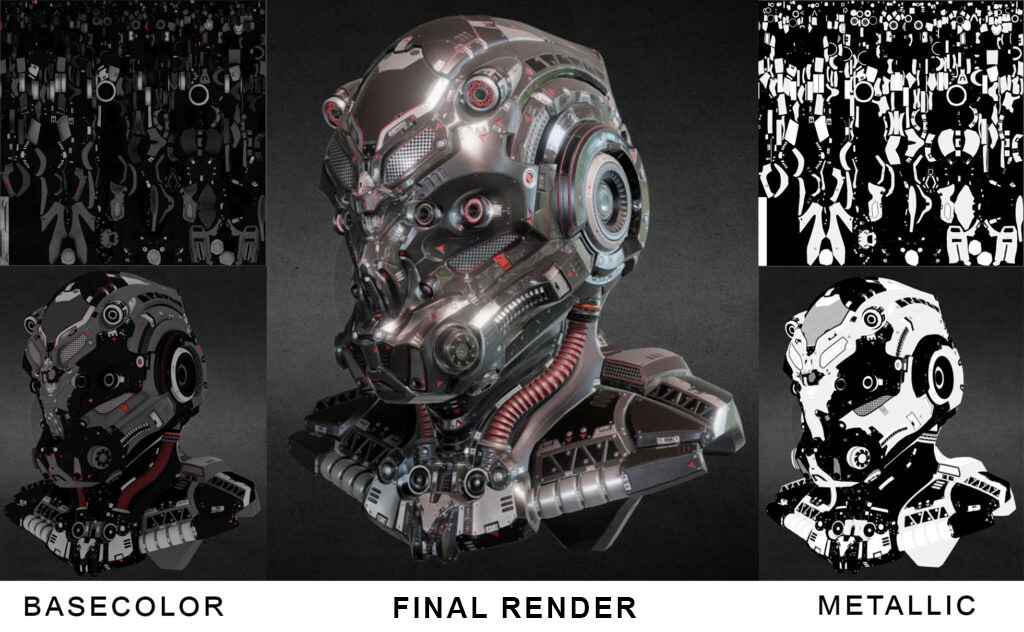
* Albedo: RGB-текстура без освещённости (диффузный цвет). Металлы имеют чёрный альбедо (отражение заменяет диффуз).
* Metallic: Градации серого (0 — диэлектрики, 1 — металлы).
* Roughness: управляет рассеянием отражений (0 — зеркало, 1 — матовость).
* Ambient Occlusion (AO): Затемнение замкнутых областей (опционально, часто объединяется с металличностью).

2. Спекулярный-Глянец (Specular-Glossiness):

* Diffuse: Альбедо с учётом затенения (устаревший подход).
* Specular: RGB-значения френелевского отражения.
* Glossiness: Инверсия шероховатости.

Дополнительные карты:

* Height Map: Смещение вершин для параллакс-эффектов (требует тесселяции).
* Emissive Map: Самосвечение (например, экраны приборов).
* Clear Coat: Отдельный слой для имитации лакированных поверхностей.[[5](#_Список_используемой_литературы)]



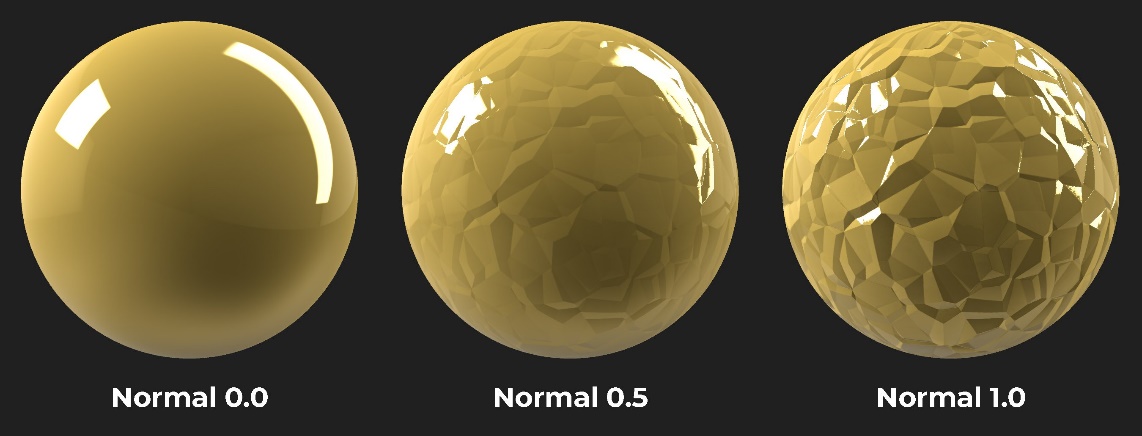
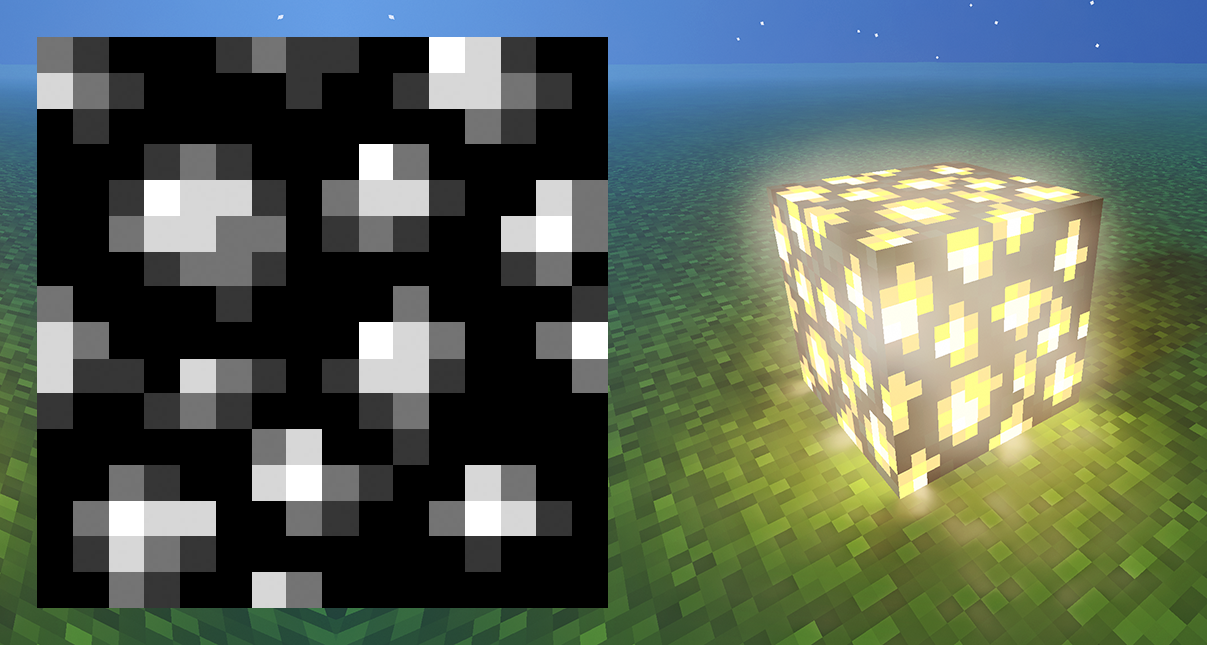
Рисунок 7. Влияние текстур Base Color, Metallic

Рисунок 8. Влияние текстуры Normal Map

Изображение выглядит как дерево, сфера, на открытом воздухе, мяч

Автоматически созданное описание

Рисунок 10. Влияние текстуры Emissive

Рисунок 9. Влияние текстуры Roughness

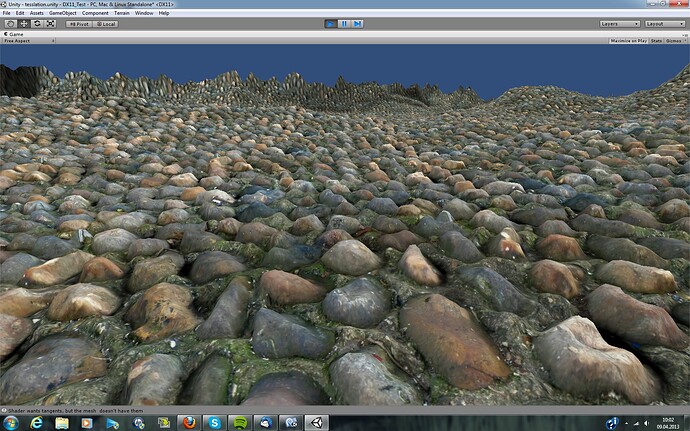
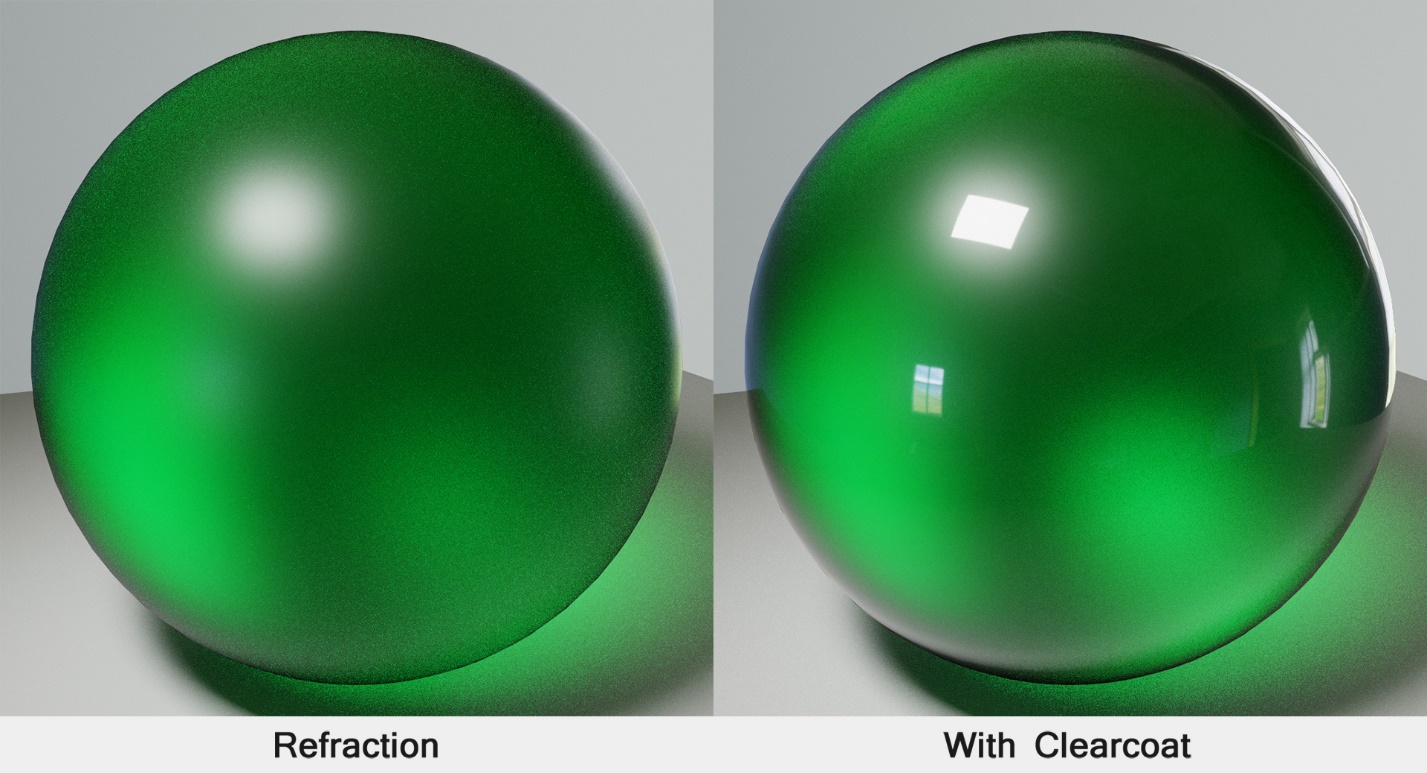


Рисунок 12. Влияние параметра Clearcoat

Рисунок 11. Влияние текстуры Height на примере тесселяции

## 1.3 Шейдеры

Шейдеры — это высокоуровневые программы, исполняемые на графическом процессоре (GPU), которые определяют этапы рендер-пайплайна, отвечая за трансформацию геометрии, расчет освещения и постобработку. Их эволюция от фиксированных функций (Fixed-Function Pipeline) до программируемых (Programmable Shading) стала ключевым прорывом в CGI, позволив реализовать фотореализм в реальном времени. В контексте веб-технологий шейдеры обеспечивают интерактивность 3D-контента, от виртуальных музеев до AR-примерочных.[[2](#_Список_используемой_литературы)]

### 1.3.1 Исторический контекст

1984: Концепция шейдинга введена Робертом Куком (Stanford) для симуляции сложных материалов.

2001: NVIDIA GeForce 3 — первая GPU с поддержкой вершинных шейдеров.

2006: WebGL 1.0 стандартизирует использование GLSL ES в браузерах.

2023: WebGPU и WGSL открывают доступ к compute-шейдерам и Ray Tracing.

### 1.3.2 Типы шейдеров и их функции

Вершинные шейдеры (Vertex Shaders**)**[[6](#_Список_используемой_литературы)]

Задача: Обработка вершин меша (положение, нормали, UV-координаты).

Основные операции:

1.Трансформации: Применение матриц Model-View-Projection (MVP).

2.Скиннинг: Деформация меша под влиянием костей (Bone Weights).

3.Процедурная генерация: Динамическое смещение вершин через шум Перлина.

Фрагментные шейдеры (Pixel Shaders)

Задача: Расчет цвета каждого пикселя с учетом освещения, материалов и постэффектов.

Геометрические шейдеры (Geometry Shaders)[[7](#_Список_используемой_литературы)]

Роль: Генерация новых примитивов (точки, линии, треугольники) на лету.

Применение:

1.Tessellation для LOD.

2.Particle Systems (например, снегопад).

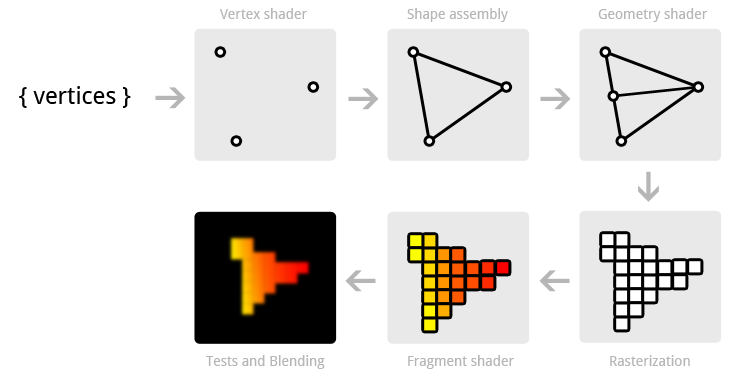


Рисунок 13. Расчет Фрагментного и геометрических шейдеров

**4. Compute-шейдеры (WebGPU/WGSL)**

* Особенность: не связаны с рендер-пайплайном; выполняют параллельные вычисления.
* Использование:
  + Физические симуляции (Cloth, Fluid Dynamics).
  + Нейросетевые вычисления (Style Transfer для текстур).

Изображение выглядит как искусство, картина

Автоматически созданное описание

Рисунок 14. Пример шейдера для манипуляции большим количеством геометрии

### 1.3.3 Шейдеры и PBR: симбиоз физики и оптимизации

PBR-рендеринг невозможен без кастомных шейдеров, которые реализуют уравнения микроповерхностей.

### 1.3.4 Оптимизации для веба:

* Pre-computed Radiance Transfer (PRT): Кэширование глобального освещения.
* Specular Occlusion: Учет Ambient Occlusion в бликах.
* Texture Streaming: Динамическая загрузка MIP-уровней.
* WebGL Inspector: Трассировка вызовов draw, анализ текстурных сэмплеров.
* ShaderToy: Онлайн-песочница для тестирования GLSL-кода.
* RenderDoc: Запись и декомпозиция кадров для поиска артефактов.

### 1.3.5 Инновации и тренды

1. Нейрошейдеры:
   * Использование ML-моделей (TensorFlow.js) для апскейла текстур в реальном времени.
2. Ray Tracing в WebGPU:
   * Патч-трассировка для мягких теней и глобального освещения.
3. Procedural Shading:
   * Генерация паттернов через шумовые функции (Worley, Simplex).

Шейдеры — это «мозг» современной компьютерной графики, трансформирующий сырые данные 3D-моделей в визуальные шедевры. Их эффективное использование в веб-приложениях требует не только знания GLSL/WGSL, но и глубокого понимания архитектуры GPU. С развитием WebGPU и нейронных сетей шейдеры становятся мостом между оффлайн-рендерингом и браузерным real-time CGI, открывая путь к метавселенным нового поколения.

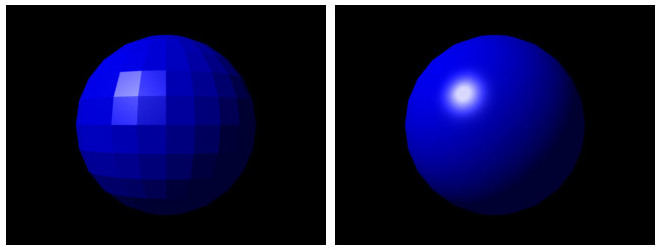


Рисунок 15. Примеры шейдинга. Flat и Smooth

## 1.4 Рендеринг

Рендеринг — это процесс синтеза 2D-изображения из 3D-данных, включающий не только геометрические преобразования, но и симуляцию физических явлений (свет, материалы, тени). В веб-среде рендеринг сталкивается с уникальными вызовами: ограниченная вычислительная мощность клиентских устройств, асинхронная загрузка ресурсов и необходимость баланса между качеством и latency. Современные подходы, такие как прогрессивный рендеринг и адаптивное сэмплирование, делают возможным отображение сложных сцен даже на мобильных устройствах.

### 1.4.1 Методы рендеринга

1. Растеризация (WebGL, WebGPU):
   * Преобразует векторные полигоны в растровые пиксели.
   * Этапы:
     + Vertex Processing: Трансформация вершин (MVP-матрицы, скиннинг).
     + Primitive Assembly: Формирование треугольников/линий из вершин.
     + Clipping: Отсечение невидимых частей за пределами frustum камеры.
     + Scan Conversion: Определение покрытия пикселей (фрагментов).
   * Оптимизации:
     + Early-Z Testing: Отбраковка невидимых фрагментов до шейдинга.
     + Tile-Based Rendering (ARM Mali): Разделение экрана на тайлы для кэширования.

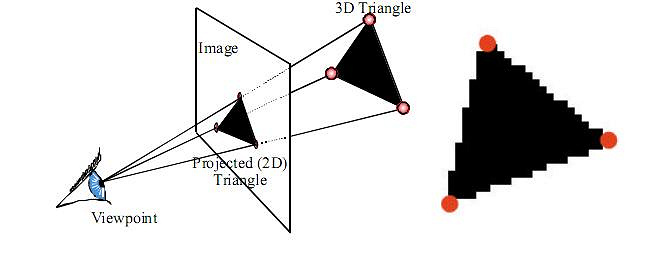


Рисунок 16. Растеризация

1. Трассировка лучей (WebGPU, экспериментально):
   * Симулирует физическое распространение света через рекурсивные лучи.
   * Использование в вебе:
     + Hybrid Rendering: Комбинирование растеризации для primary rays и RT для отражений.
     + Path Tracing SaaS: Оффлайн-рендеринг через облачные API (например, NVIDIA Omniverse).

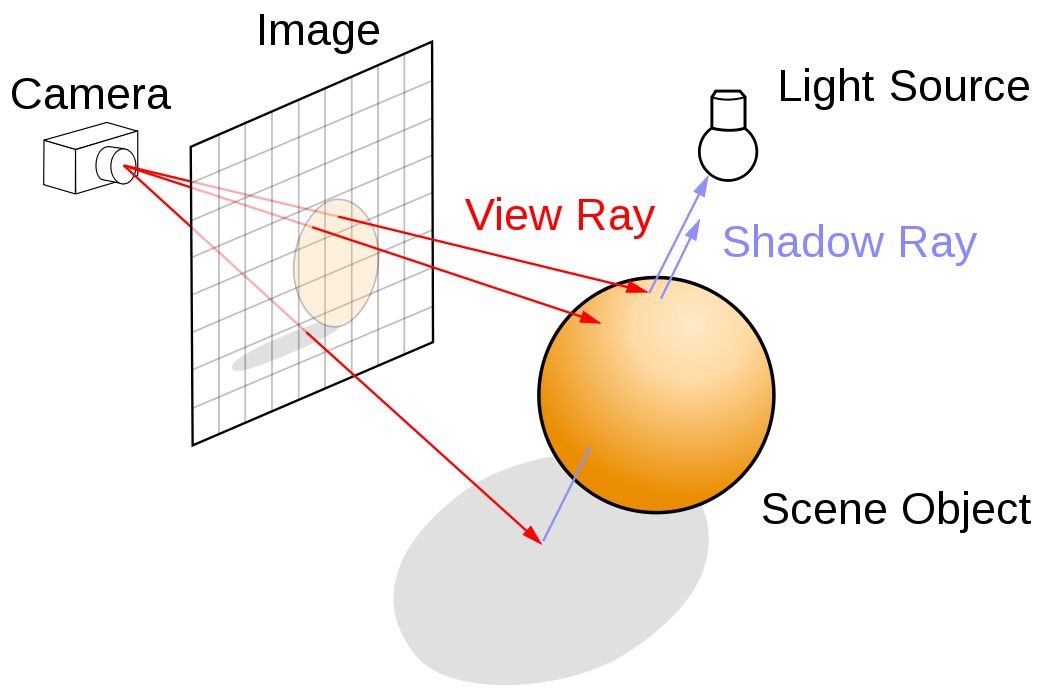


Рисунок 17. Трассировка лучей

### 1.4.2 Этапы рендеринга

1. Обработка вершин (Vertex Processing):
   * GPU Instancing**:** Отрисовка тысяч объектов с одним мешем через инстанс-буферы.

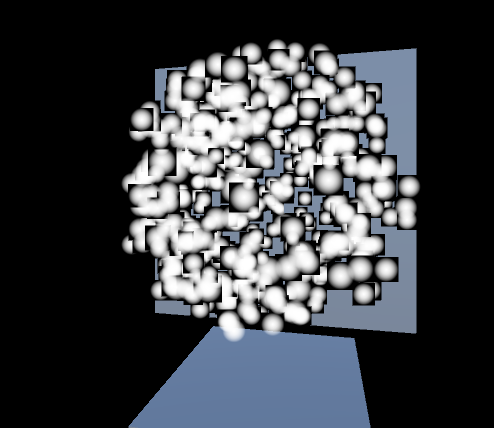


Рисунок 18. Instancing на примере частиц в Unity

* + Morph Targets: Анимация деформаций (мимика лица) через весовые коэффициенты.[[8](#_Список_используемой_литературы)]

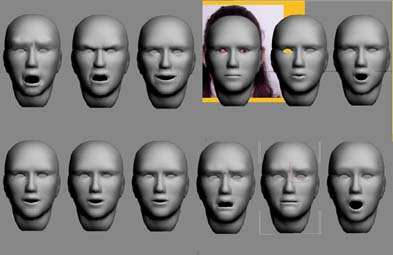


Рисунок 39. Деформация лица с помощью весовых коэффициентов

* + Compute-Driven: Вычисление позиций вершин через compute-шейдеры (например, вода в реальном времени).

1. Растеризация и управление фрагментами:
   * Z-Buffer Algorithm: Устранение перекрытий через глубину [[9](#_Список_используемой_литературы)]

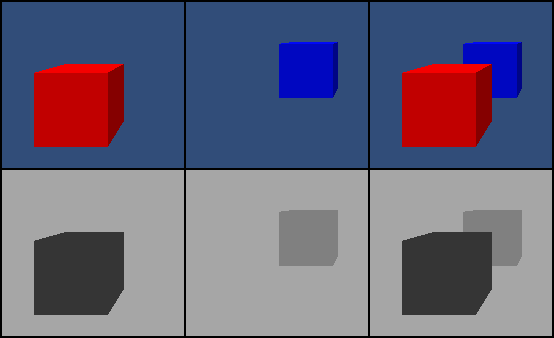


Рисунок 20. Z-Буфер

* + MSAA (Multisample Anti-Aliasing): Субпиксельное сэмплирование для сглаживания краёв.[[10](#_Список_используемой_литературы)]

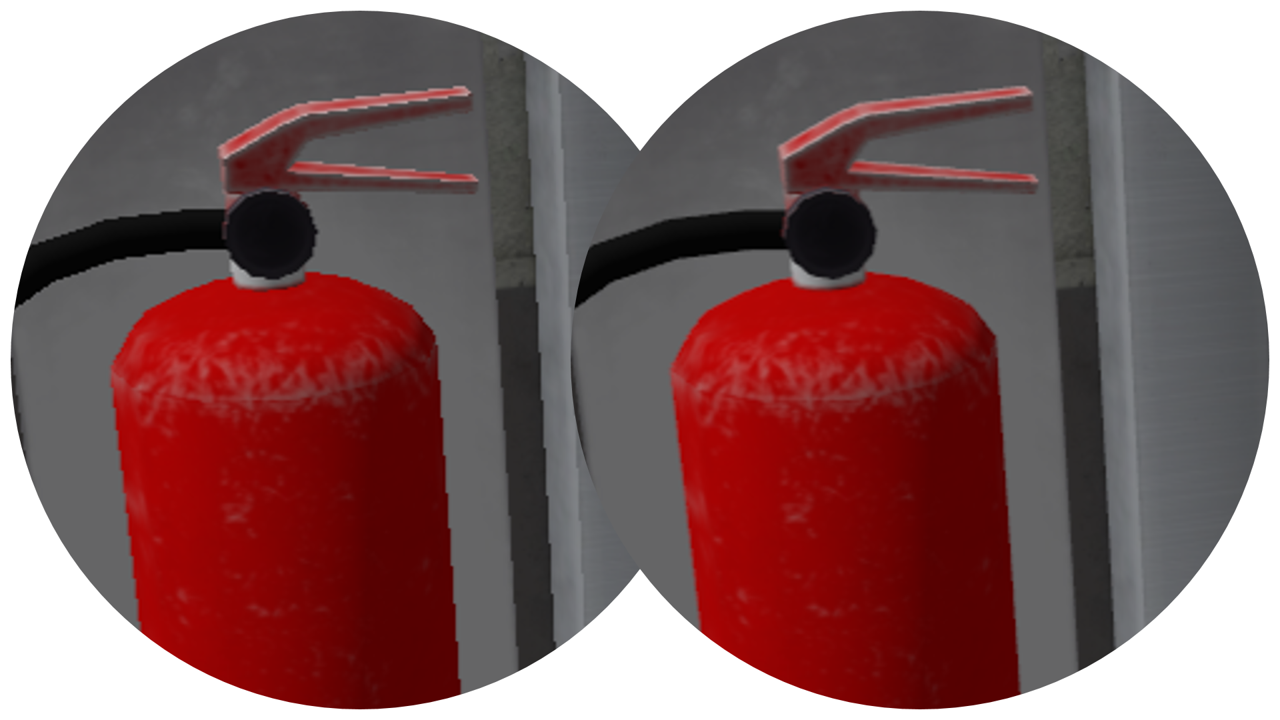


Рисунок 21. Пример MSAA(Слева- без MSAA)

* + Fragment Discard: Отбрасывание прозрачных фрагментов через discard в шейдере.

1. Фрагментный шейдинг:
   * Deferred Shading: Разделение геометрии и освещения в G-буфер (позволяет использовать сотни источников света в одной сцене).

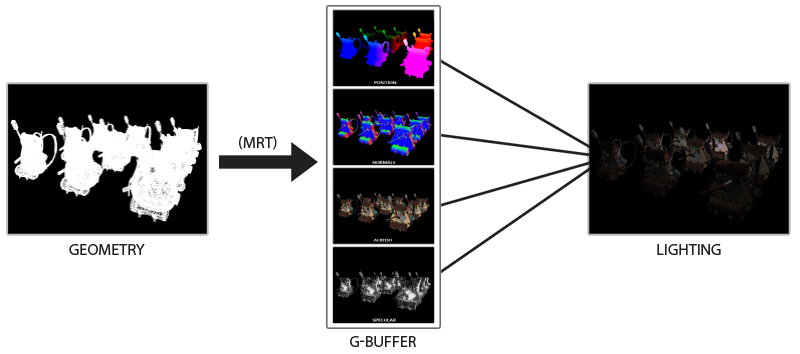


Рисунок 22. Работа Deferred Shading

* + Forward+: Кластеризация света для оптимизации forward-рендеринга. [[11](#_Список_используемой_литературы)]

Изображение выглядит как снимок экрана, диаграмма, зеленый

Автоматически созданное описание

Рисунок 23. Кластеризация света

### 1.4.3 Оптимизация рендеринга для веба

1. Управление ресурсами:
   * Texture Atlases: Снижение числа draw calls через объединение текстур.
   * GPU Memory Budgeting: Мониторинг WEBGL\_lose\_context для предотвращения падений.
2. Culling-техники:
   * Frustum Culling: Отсечение объектов вне зоны видимости камеры (использует AABB/OBB-боксы).
   * Occlusion Culling: Определение перекрытий через hardware queries (например, glOcclusionQuery).
3. Адаптивные стратегии:
   * Dynamic LOD: Автоматическое упрощение мешей на основе расстояния[[3](#_Список_используемой_литературы)]

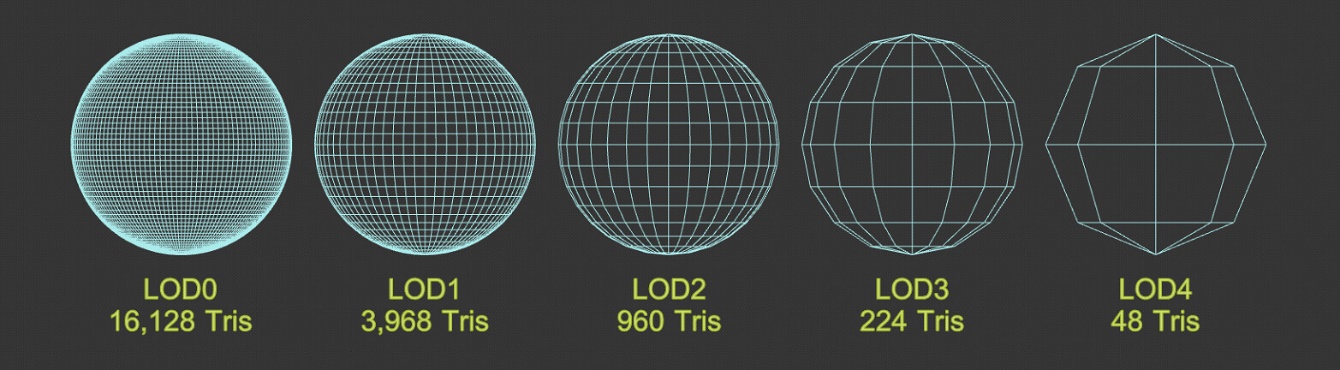


Рисунок 24. Примеры LOD

* + Priority Streaming: Загрузка текстур «по требованию» для видимых MIP-уровней.

## 1.5 Методологические аспекты разработки веб-сервиса

Разработка требует четкого понимания методологии, которая включает анализ требований, выбор подходящих технологий, проектирование архитектуры, реализацию и тестирование. Проанализируем основные этапы и методологические аспекты, которые необходимо учитывать при создании веб-сервиса

### 1.5.1 Выбор средств разработки веб-сервиса

Обоснование архитектуры и технологий

Для реализации веб-платформы с возможностью загрузки, хранения и просмотра 3D-моделей выбран классический подход клиент-сервер с подключением к централизованной базе данных. Такой подход обеспечивает надежное хранение пользовательских данных, масштабируемость и возможность интеграции дополнительных сервисов (например, монетизации или продвижения работ), что характерно для ведущих решений на рынке: Sketchfab, Clara.io, Vectary.

Ключевые технологические решения:

Three.js — основной инструмент для визуализации и работы с 3D-графикой в браузере. Эта библиотека обеспечивает кроссплатформенность, гибкость и поддержку современных стандартов 3D, что позволяет реализовать функционал, сопоставимый с конкурентами.

Централизованная база данных — все пользовательские модели (FBX, GLB, OBJ), текстуры (JPEG) и параметры сцен (JSON) хранятся на сервере. Это позволяет обеспечить целостность данных, их резервное копирование и централизованное управление, что важно для коммерческих сервисов и маркетплейсов.

Фронтенд — реализован с использованием JavaScript и Three.js для загрузки моделей, настройки сцены и генерации embed-кода.

Анализ конкурентов:

Sketchfab и Clara.io используют аналогичную методологию, что подтверждает ее эффективность для задач хранения и публикации 3D-контента.

### 1.5.2 Определение требований к разрабатываемому продукту

Функциональные требования:

1. Загрузка и хранение 3D-моделей в форматах GLB, FBX, OBJ через веб-интерфейс.

2. Хранение моделей, текстур и параметров сцены в централизованной базе данных.

3. Возможность настройки материалов, освещения, фона и камеры для каждой сцены.

4. Создание и управление коллекциями моделей с предпросмотром (галерея).

5. Детальный просмотр моделей с возможностью настройки и генерации embed-кода для встраивания на внешние сайты.

6. Взаимодействие с сервером через API для загрузки, получения и обновления данных.

Нефункциональные требования:

1. Кроссплатформенность: поддержка всех современных браузеров, включая мобильные устройства.

2. Высокая производительность визуализации за счет использования Three.js и WebGL.

3. Масштабируемость архитектуры для поддержки роста числа пользователей и объема данных.

4. Безопасность хранения и передачи пользовательских данных.

5. Интуитивно понятный и адаптивный интерфейс для работы с 3D-сценами и коллекциями моделей.

### 1.5.3 Определение целевой аудитории и анализ конкурентов

Платформа ориентирована на следующие основные сегменты пользователей:

1. 3D-художники и дизайнеры: профессионалы и энтузиасты, создающие 3D-модели для портфолио, заказчиков, игр, анимации и визуализации. Для них важны гибкие инструменты настройки сцены, поддержка популярных форматов, возможность быстрой публикации и интеграции работ на внешние сайты.

2. Веб-дизайнеры и разработчики: специалисты, которым необходимы интерактивные 3D-виджеты для сайтов, презентаций, маркетплейсов. Ключевые требования — простота встраивания, стабильная работа в браузере, поддержка embed-кода.

3. Преподаватели и студенты: образовательные учреждения и учащиеся, использующие 3D-модели для учебных целей, создания учебных материалов, проведения лабораторных работ и визуализации сложных концепций. Для этой группы важны доступность, простота интерфейса и возможность групповой работы.

4. Маркетплейсы и бизнес-клиенты: компании, использующие 3D-контент для презентаций, рекламы, демонстрации товаров, а также для организации онлайн-продаж моделей. Для них критичны интеграция с внешними сервисами, инструменты продвижения и коммерциализации контента.

Характеристики и потребности:

1. Уровень подготовки: от новичков до профессионалов.

2. Основные задачи: загрузка, хранение, настройка, публикация и встраивание 3D-моделей без сложных технических настроек.

3. Потребности: поддержка популярных форматов, интуитивный интерфейс, стабильная работа, возможность совместной работы и обмена моделями

# 2. Практическая часть

## 2.1 Структура разработки

На начальной стадии структура сайта будет следующей:

1. Редактор моделей: веб-приложение, позволяющее производить настройку 3D модели внутри сцены Three.js. Интерфейс должен быть понятным для пользователя, не посвященного в 3D графику, он должен быть последовательным и иметь минимальные необходимые настройки. Модель, сохраненная в редакторе, будет передаваться в базу данных с помощью api.
2. Просмотрщик(Viewer) моделей – страница, позволяющая пользователю просматривать модели и взаимодействовать с ними. Так же viewer будет использоваться в качестве виджета на других сайтах с помощью embed-кода. Это позволит бесшовно встраивать интерактивные 3D-модели на другие сайты.
3. Галерея – главный сайт, на котором будет размещен список 3D-моделей. В будущем галерею можно будет разделить по категориям, встроить монетизацию, продвижение, а также полноценную систему рекомендаций.

После продумывания структуры сайта можно приступать к созданию пользовательской части. Для этого нужно провести анализ того, что может делать пользователь. Чтобы наглядно продемонстрировать схему взаимодействия пользователя с веб-сервисом, написана диаграмма использования в виде UML: Use-case диаграммы на рис.25

## Изображение выглядит как диаграмма, текст, линия, снимок экрана Автоматически созданное описание2.2 Разработка сайта

Рисунок 25. Use-case диаграмма

### 2.2.1 Разработка редактора

Так как основной частью проекта на данный момент является редактор, я начал разработку проекта именно с него. На рис.26 показан первоначальный дизайн редактора. Реализовано основное окно настроек, вьюпорт(окно с моделью), и header с названием модели.

Изображение выглядит как снимок экрана, лягушка, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 26. UX дизайн редактора

Для последовательности импорта настройки окно настроек разбито на вкладки:

* 1. Scene (Сцена) – будет содержать базовые настройки, такие как расположение модели, настройки камеры, вспомогательные элементы, и др.
  2. Lighting (Освещение) – будет содержать настройки освещения в сцене
  3. Materials (Материалы) – будет содержать настройки материалов модели: возможность переключить режимы рендеринга и загрузить текстуры.
  4. Post-processing (Пост-обработка) – настройки постобработки (экспозиция, контрастность, визуальные эффекты и др.)
  5. Animations (Анимации) – настройки анимации моделей. Тут можно будет настроить параметры, связанные с анимациями.
  6. AR/VR – параметры, связанные с виртуальной и дополненной реальностью.

Внутри вкладок каждый параметр будет находиться в своей раскрывающейся вкладке.

Доработав дизайн и добавив нужные параметры, получилось такое окно редактирования. Рис.27.

Добавлены параметры:

1. Transform – изменение положения, поворота и размера объекта.

2. Camera settings – настройка угла обзора камеры и near clipping distance.

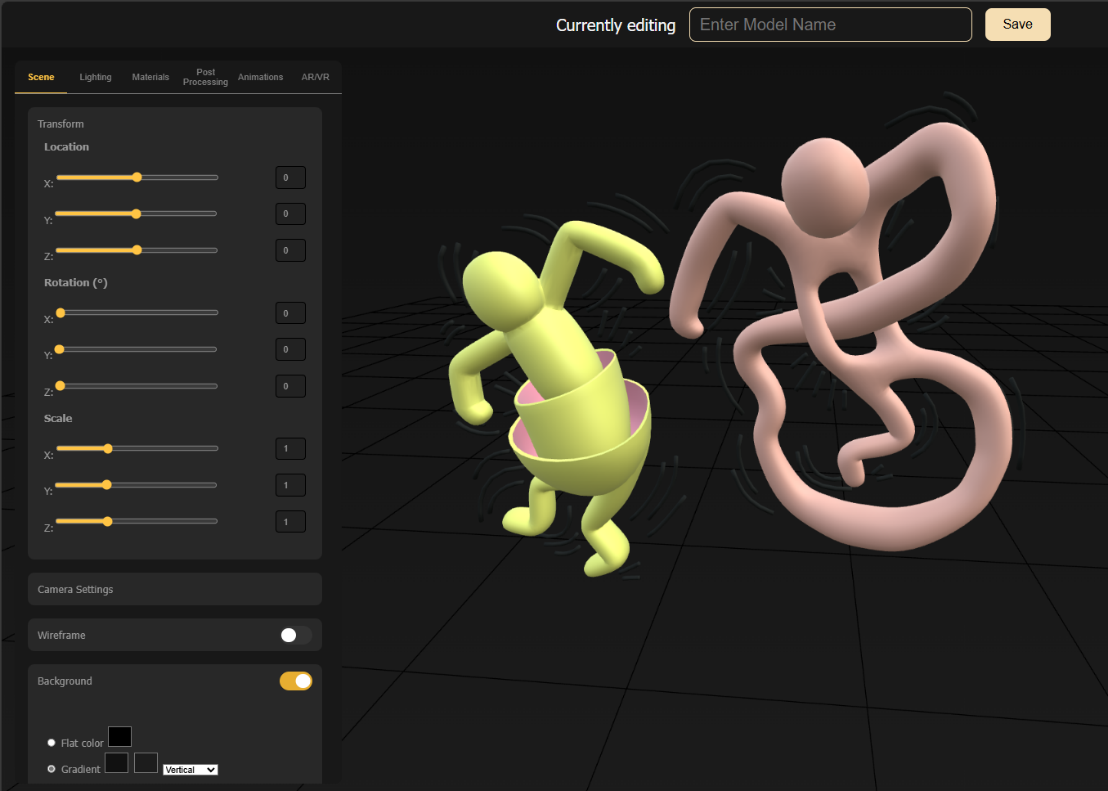
3. Wireframe – отображение и настройка полигональной сетки модели

4. Background – настройка заднего фона(сплошной цвет, градиент, hdri)

5. Grid – вспомогательный элемент, сетка, расположенная на полу.

6. HDRI(Lighting) – выбор и настройка HDRI карты освещения(рис.28)

7. Настройка материалов – выбор между PBR, Matcap и Primal. По умолчанию модель должна загружаться с уже настроенными материалами, но если этого не происходит, их можно загрузить вручную



Изображение выглядит как снимок экрана, мультфильм, Анимация, Фигурка животного

Автоматически созданное описаниеРисунок 27. Интерфейс редактора

Рисунок 28. Выбор HDRI

После завершения настройки модели внутри сцены, пользователю предлагается ввести название модели в поле наверху страницы. В случае если пользователь попытается нажать кнопку Save(сохранить) оставив поле пустым, сохранения не произойдет, и пустое поле мигнёт красным цветом. При успешном сохранении, в нижней части экрана появится «снимок» модели, это превью, которое будет использоваться для отображения модели на сайте. При желании пользователь может переделать снимок, изменив ракурс камеры. Рис.29

Изображение выглядит как снимок экрана, Мультимедийное программное обеспечение, программное обеспечение, Графическое программное обеспечение

Автоматически созданное описание

Рисунок 29. Демонстрация превью модели

Как только превью устроит пользователя, он может нажать на кнопку Publish(опубликовать), и ассет будет сохранен в базе данных и будет отображаться в галерее.

### 2.2.2 Разработка viewer моделей

Viewer должен будет подгружать файлы из базы данных, и создавать сцену с параметрами, указанными в json файле. Так же должна быть реализована функция генерации embed-кода для встраивания на другие сайты в качестве виджета.

### 2.2.3 Разработка галереи моделей

Изображение выглядит как текст, транспортное средство, Наземный транспорт, машина

Автоматически созданное описаниеГлавная страница будет выполнять обязанности лендинг-страницы, а также галереи моделей. Для ПК версии был разработан макет сайта на рис.30

Рисунок 30. Главная страница

На главной странице расположен интерактивный виджет модели, встроенный туда с помощью embed-кода, кнопка Upload, открывающая окно загрузки файла, с последующим переходом в редактор, а также кнопка Browse, которая листает страницу до галереи моделей.

Так же была разработана адаптивность сайта. В мобильном режиме нет возможности загрузить 3D-модель, поэтому кнопка Upload скрыта, а галерея сделана в один столбик. Рис.31 и Рис.32.

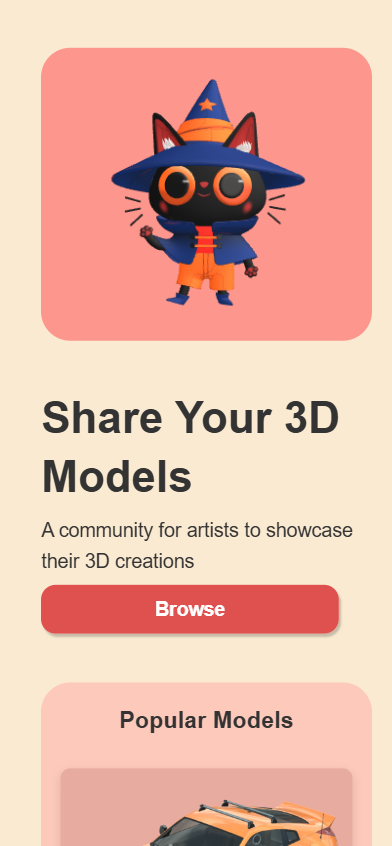


Рисунок 31. Адаптивный дизайн

Изображение выглядит как шина, текст, автокомпонент, транспортное средство

Автоматически созданное описание

*Рисунок 32. Адаптивный дизайн галереи*

## 2.3 Тестирование сайта

Тестирование – это не просто этап разработки, а гарантия того, что приложение будет работать корректно, безопасно и эффективно. Оно помогает миними1зировать риски, снизить затраты на исправление ошибок и повысить удовлетворенность пользователей. Без тестирования невозможно создать качественный продукт, который соответствует требованиям заказчиков и конечных пользователей. Именно поэтому тестирование является одним из ключевых аспектов успешной разработки программного обеспечения.

Таблица 1 – Тестирование, проведенное разработчиками

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Подробное описание** | **Шаг к воспроизведению** | **Ранжирование ошибки** | **Возможность обойти** | **Решение** |
| 1 | После загрузки файла редактор выдает ошибку | Нажать «Upload», загрузить «битый» .glb файл | Высокая | Нет | Добавить обработку ошибок, проверку формата и размера файла |
| 2 | Материалы отображаются неправильно или отсутствуют | Загрузить модель с PBR-материалами | Средняя | Частично | Добавить обработку PBR-материалов |
| 3 | После обновления страницы настройки камеры/света сбрасываются | Настроить сцену  Обновить страницу | Низкая | Да | Добавить авто-сохранение |
| 4 | Интерфейс «разьезжается», элементы перекрывают друг друга на мобильных устройствах | Открыть сайт на смартфоне  Перейти в галерею | Средняя | Частично | Доработать адаптивную верстку, протестировать на разных устройствах |
| 5 | Долгая загрузка больших моделей(>50МБ) | Загрузить большую модель | Средняя | Частично | Изменить способ загрузки, оптимизировать процесс |

Таблица 2 – Тестирование, проведенное независимым пользователем

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Подробное описание** | **Шаг к воспроизведению** | **Ранжирование ошибки** | **Возможность обойти** | **Решение** |
| 1 | Не отображается превью загруженных моделей | Загрузить новую модель  Перейти в галерею | Средняя | Частично | Проверить генерацию превью, добавить повторную генерацию по кнопке |
| 2 | Превью отображается криво, модель захвачена не с правильного ракурса | Загрузить новую модель  Перейти в галерею | Низкая | Частично | Добавить предпросмотр превью в редакторе при сохранении |
| 3 | Сайт долго загружается на старых устройствах | Перейти на главную страницу | Низкая | Нет | Оптимизировать загрузку моделей |
| 4 | Не работает drag-and-drop для загрузки | Перетащить 3D-файл на область загрузки | Средняя | Да | Реализовать обработку drag-and-drop, добавить визуальное выделение области при наведении |

## 2.4 Информационная безопасность

Информационная безопасность веб-платформы для работы с 3D-моделями включает комплекс мер по защите пользовательских данных, моделей и сервисной инфраструктуры от внешних и внутренних угроз. Особое внимание уделяется защите авторских прав на 3D-контент, предотвращению несанкционированного доступа, утечек данных и обеспечению устойчивости платформы к кибератакам.

Таблица 3 – Риски

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **Риск** | **Последствия** | **Вероятность** | **Решение** |
| 1 | Несанкционированный доступ к данным | Кража аккаунтов, компрометация моделей | Средняя | Внедрение аутентификации, ограничение попыток входа, аудит активности |
| 2 | Загрузка вредоносных файлов | Вредоносное ПО на сервере, возможные атаки на пользователей | Средняя | Проверка расширений, антивирусная проверка |
| 3 | Кража или пиратство 3D-моделей | Незаконное распространение и коммерческое использование чухого контента | Высокая | Водяные знаки, ограничение скачивания, защита embed-кода |
| 4 | DDoS-атаки и перегрузка сервера | Недоступность сайта для пользователей, потеря прибыли и репутации | Средняя | Использование CDN,ограничение частоты запросов, защита на уровне сервера |
| 5 | Потеря данных из-за сбоев или атак | Удаление, повреждение или полная утрата пользовательских и сервисных данных | Низкая | Регулярное резервное копирование, хранение бэкапов на независимых носителях |

# 3.Экономическая часть

## 3.1 Организация и экономика производства

В данном разделе рассматривается структура организации работ по разработке веб-платформы для работы с 3D-моделями, а также расчет основных экономических показателей проекта.

На разработку требуются три ключевые роли: разработчик, тестировщик и руководитель проекта. Разработчик создает программный код, тестировщик проверяет функционал приложения. Расчет оплаты труда предоставлен в таблице 4.

Таблица 4 – оплата труда



Материальные затраты включают покупку оборудования, лицензий на ПО и другие необходимые ресурсы. Подробные данные приведены в таблице 5.

Таблица 5 – Материальные затраты



Оборудование имеет ограниченный срок службы, что требует учета амортизации. Расчет ежегодных отчислений приведен в таблице 6.

Таблица 6 – Амортизационные отчисления



Расходные материалы включают офисные принадлежности, электроэнергию и интернет. Пример в таблице 7.

Таблица 7 – Расходные материалы



Прочие расходы включают маркетинговые расходы, обучение команды и услуги внешних специалистов показаны в таблице 8.

Таблица 8 – Прочие материалы



Себестоимость проекта рассчитывается из итоговых значений предыдущих таблице приведена в таблице 9.

Таблица 9 – Себестоимость проекта



Затраты на проект также были приведены на диаграмме 1.

Себестоимость проекта выходит на сумму 1291741 руб.

## 3.2 Экономическая эффективность продукта

Экономическая эффективность веб-платформы для работы с 3D-моделями основана на сочетании низких эксплуатационных затрат и гибкой системы монетизации. Проект рассчитан на окупаемость в течение 12-18 месяцев при условии стабильного роста аудитории

Модели монетизации указаны в таблице 10.

Таблица 10 – Модели монетизации

|  |  |  |  |
| --- | --- | --- | --- |
| **Модель** | **Описание** | **Целевая аудитория** | **Средний чек** |
| Комиссия с продаж моделей | 15% от стоимости каждой модели, проданной через платформу | 3D-художники, студии | 300руб./транзакция |
| Премиум-подписка | Платные embed-коды, расширенное хранилище, продвижение | Профессионалы, e-commerce | 5000руб./мес. |
| Продвижение работ | Выделение моделей в топах поиска, баннеры на главной странице | Все пользователи | 200руб./модель |
| Платный embed-код | Лицензия на встраивание моделей на коммерческие сайты | e-Commerce, веб-дизайнеры | 200руб./модель |

Большая часть доходов расчитана на веб-дизайнеров и e-Commerce, которым нужно встроить интерактивную модель продукта на свои сайты. Для художников и гостей площадки обязательных затрат не предусмотрено.

Таблица 11 – прогноз доходов (на 12 месяцев)

|  |  |  |
| --- | --- | --- |
| **Параметр** | **Значение** | **Расчет дохода(руб./мес.)** |
| Ежемесячная аудитория | 5 000 пользователей | - |
| Конверсия в продажи | 8% (400 транзакций/мес.) | 400 \* 300 = 120 000 |
| Премиум-подписчики | 5% (250 пользователей) | 250 \* 1500 = 375 000 |
| Платные embed-коды | 400 моделей/мес. | 400 \* 200 = 80 000 |
| Продвижение работ | 200 моделей/мес. | 200\*200 = 40 000 |
| Суммарных доход/мес. |  | 615 000 |

Таблица 12 – затраты и окупаемость

|  |  |
| --- | --- |
| **Показатель** | **Сумма (руб.)** |
| Себестоимость | 1 291 741 |
| Ежемесячные расходы | 40 000 ~ |
| Годовой доход | 7 380 000 |
| Годовые расходы | 1 771 741(480 000 + 1 291 741) |
| Чистая прибыль/год | 5 608 259 |

Срок окупаемости: 1.5 месяца после выхода на плановую аудиторию

Факторы роста эффективности:

* 1. Разработка мобильного приложения
  2. Уникальность сервиса на территории стран СНГ
  3. Расширение функционала для корпоративных клиентов (групповые аккаунты, API)
  4. Активное продвижение в профессиональных сообществах

Риски:

1. Конкуренция с крупными платформами (Sketchfab)
2. Изменение спроса на 3D-контент
3. Технические сбои при масштабировании

# Заключение

В ходе выполнения дипломной работы была разработана современная веб-платформа для загрузки, хранения, настройки и просмотра 3D-моделей, ориентированная на широкий круг пользователей — от профессиональных 3D-художников до представителей бизнеса и образовательных учреждений. Реализация проекта позволила решить ряд актуальных задач, связанных с интеграцией интерактивной 3D-графики в веб-пространство, а также сделать этот процесс доступным и удобным для пользователей с разным уровнем технической подготовки.

В результате создан веб-сервис, который позволяет пользователям не только быстро загружать и настраивать 3D-модели, но и легко встраивать их на внешние сайты с помощью генерации embed-кода. Реализованы возможности кастомизации сцены, управления материалами, освещением и камерой, что делает платформу универсальным инструментом для самых разных задач — от создания портфолио до демонстрации товаров в интернет-магазинах.

Экономическая часть работы показала, что внедрение подобного сервиса является не только технологически актуальным, но и экономически эффективным решением. Благодаря гибкой системе монетизации, низким эксплуатационным затратам и высокой востребованности интерактивных 3D-виджетов на рынке, проект обладает значительным потенциалом для коммерческого развития и быстрой окупаемости.

Таким образом, поставленная в начале работы цель — создание доступного, производительного и функционального веб-сервиса для интеграции 3D-графики в онлайн-платформы — была успешно достигнута. Разработанный продукт способен повысить качество визуального представления товаров и услуг, улучшить пользовательский опыт и открыть новые возможности для бизнеса и образования в условиях цифровой экономики.

Дальнейшее развитие проекта может включать расширение функционала, интеграцию с внешними сервисами, внедрение новых форматов 3D-моделей и инструментов для совместной работы, что позволит ещё больше повысить ценность платформы для различных категорий пользователей.

# Список используемой литературы

1. Интернет статья [Электронный ресурс] URL: <https://www.artstation.com/blogs/luismesquita/jGXd/everything-about-pbr-textures-and-a-little-more-part-2>
2. Интернет статья [Электронный ресурс] URL: <https://www.computerhope.com/jargon/p/pixel-shader.html>
3. Интернет статья [Электронный ресурс] URL: <https://www.artstation.com/blogs/samslover/QwNE/ue4-optimization-performance-pt2-lods>
4. Интернет статья [Электронный ресурс] URL: <https://www.researchgate.net/figure/Mesh-decimation-takes-the-original-data-left-and-computes-a-coarse-approximation-with_fig1_234782118?__cf_chl_tk=YBkIRy3C0Sm84hFdrFRKmEOD29iwzvQOmeymVN5Nq9U-1741857406-1.0.1.1-bBvplB2PFFs3IWBIss5SI_3gP3mx67QB4YIzgYQWh3g>
5. Интернет статья [Электронный ресурс] URL: <https://marmoset.co/posts/create-clearcoat-materials-in-toolbag-4/>
6. Интернет статья [Электронный ресурс] URL: <https://open.gl/drawing>
7. Интернет статья [Электронный ресурс] URL: <https://catlikecoding.com/unity/tutorials/basics/compute-shaders/>
8. Интернет статья [Электронный ресурс] URL: https://www.researchgate.net/figure/Morph-Target-Models\_fig4\_221573077
9. Интернет статья [Электронный ресурс] URL: <https://eccarrilloe.github.io/2017/09/24/Rasterization-Z-buffer-Algorithm/>
10. Интернет статья [Электронный ресурс] URL: <https://docs.vulkan.org/samples/latest/samples/performance/msaa/README.html>
11. Интернет статья [Электронный ресурс] URL: <https://docs.unity3d.com/Packages/com.unity.render-pipelines.high-definition@14.0/manual/Forward-And-Deferred-Rendering.html>