

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования

ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ  
ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Институт информационных технологий и анализа данных  
наименование института

Допускаю к защите

Руководитель

  
подпись

А.В. Жуков  
И.О. Фамилия

Разработка web-приложения для тестирования ПДД с 3D-моделированием  
наименование темы

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА  
к курсовому проекту  
по дисциплине

«Web-программирование»

1.003.00.00 - ПЗ

обозначение документа

Выполнил студент

АСУ6-20-2

шифр

  
подпись

А.В. Арбакова

И.О. Фамилия

Нормоконтроль

  
подпись

А.В. Жуков

И.О. Фамилия

Курсовой проект защищен с оценкой

отлично

Иркутск 2023 г.

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования

ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ  
ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

**ЗАДАНИЕ**  
НА КУРСОВОЕ ПРОЕКТИРОВАНИЕ

По курсу Web-программирование

Студенту Арбаковой Анастасии Вячеславовне  
(фамилия, инициалы)

Тема проекта: Разработка web-приложения для тестирования ПДД  
с 3D-моделированием

Исходные данные: Разработка web-приложения для тестирования ПДД  
с 3D-моделированием с использованием технологий Three.js и GSAP.

Рекомендуемая литература:


1. Документация по Three.js. URL: <https://threejs.org/docs/>
2. Learning Three.js: The JavaScript 3D Library for WebGL. Published by Packt Publishing Ltd, ISBN 978-1-78439-221-5, 2015 г.
3. Документация по GSAP. URL: <https://greensock.com/docs/>

Графическая часть на 12 листах.

Дата выдачи задания "10" марта 2023 г.

Задание получил  Арбакова А.В.  
подпись И.О.Фамилия

Дата представления проекта (работы) руководителю "15" июня 2023 г.

Руководитель курсового проектирования (курсовой работы)  Жуков А.В.  
Подпись И.О.Фамилия

## Содержание

Введение .....	4
1 Анализ предметной области .....	5
2 Определение концепции разрабатываемого приложения .....	6
3 Описание используемых технологий Three.js, GSAP и Blender .....	9
4 Проектирование приложения .....	11
5 Тестирование разработанного приложения .....	15
6 Инструкция пользователя .....	19
7 Анализ перспектив развития проекта .....	22
Заключение .....	23
Список использованных источников .....	24

## **Введение**

Развитие персональных компьютеров привело к появлению компьютерных сетей, которые обеспечивают обмен данными между устройствами. Объединение локальных сетей в глобальную сеть повлекло за собой появление – Интернета.

Теперь же практически во всех сферах человеческой деятельности применяются информационные технологии, неотъемлемой частью которой является разработка Интернет-ресурсов. С развитием Интернета возникло такое понятие, как web-программирование. Web-программирование – раздел программирования, ориентированный на разработку веб-приложений. Создание web-сайтов является одной из важнейших технологий разработки ресурсов в Интернете, где сайты могут быть разной тематики. Сайты содержат полезную информацию для конкретной целевой аудитории, или выполняют конкретные узконаправленные задачи, или используются для коммерческой фирмы, частного лица или образовательного учреждения.

В данной работе описывается процесс разработки web-приложения для тестирования ПДД с использованием 3D-моделирования.

Список задач курсовой работы:

1. Анализ предметной области
2. Определение актуальности и цели работы
3. Определение концепции разрабатываемого приложения
4. Описание используемых технологий Three.js, GSAP и Blender
5. Проектирование приложения
6. Тестирование разработанного приложения
7. Анализ перспектив развития проекта

## **1 Анализ предметной области**

Получение водительских прав является одним из основных шагов к взрослой жизни для большинства людей. Водительские права дают человеку возможность большую свободу выбора в перемещении по стране, ведь он может ехать куда и когда захочет самостоятельно.

Однако, водительские права выдаются государством только обученным водителям, тем, кто закончил автошколу и сдал государственный экзамен. Люди, управляющие транспортными средствами, должны быть обучены вождению, чтобы избежать дорожно-транспортных происшествий. Но для многих этап сдачи экзамена становится самым сложным, и без должной подготовки шанс успешной сдачи является практически минимальным.

Разрабатываемое web-приложение должно представлять собой систему проверки знаний пользователя для подготовки его к сдаче экзамена на водительские права, посредством визуализации вопросов экзамена с помощью отображения анимированной трехмерной компьютерной графики. Для корректной разработки web-приложения следует ознакомиться и изучить правила дорожного движения (ПДД) РФ, используемые при сдаче экзамена на водительские права. Правила дорожного движения являются правовым документом, они устанавливают нормы общественного поведения участников дорожного движения и призваны обеспечить безопасность.

Автошколы рекомендуют выполнять тесты ПДД, ведь данные тесты дают возможность проверить знания в области правил дорожного движения и выявить пробелы в знаниях. Тесты повторяют вопросы, которые утверждены в качестве экзаменационных для проверки знаний ПДД в ГИБДД.

Актуальность данной работы заключается в проверке своих знаний пользователем при подготовке к сдаче экзамена на водительские права.

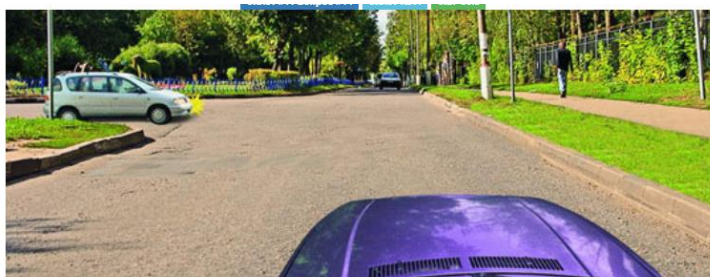
Целью работы является создание веб-приложения для сдачи тестирования ПДД пользователем с использованием 3D-моделирования ситуаций на дороге.

## 2 Определение концепции разрабатываемого приложения

Концепцией разрабатываемого приложения является подготовка к сдаче экзамена на водительские права, с использованием трехмерного моделирования и его визуальной составляющей для конкретных ситуаций из имеющихся вопросов на экзамене.

Для разработки web-приложения будут использоваться такие средства, как: язык разметки HTML5 – для создания и отображения веб-приложения, язык декорирования CSS – для описания внешнего вида веб-приложения, язык программирования JavaScript – для создания веб-приложения, Three.js – для отображения анимированной компьютерной 3D графики, GSAP – для создания анимации на основе временной шкалы, и Blender – для создания трехмерной компьютерной графики.

Вопросы, выбранные из экзамена на водительские права и подлежащие трехмерному моделированию, представлены на рисунках 1-6.



В каком случае Вы имеете право проехать перекресток первым?

1. Только при движении прямо.
2. При движении прямо и налево.
3. При движении прямо, налево и в обратном направлении.

Далее [Вопрос 1.2.3 - выбор](#)

При движении прямо или налево данный перекресток равнозначных дорог Вы имеете право проехать первым, поскольку водитель легкового автомобиля, находящегося слева, должен уступить Вам дорогу п. 13.11. При развороте это ТС становится для Вас «помехой справа». Поэтому преимущество в движении переходит к нему.

Рисунок 1 – Вопрос 1



При повороте налево Вы:

1. Имеете преимущество.
2. Должны уступить дорогу только автобусу.
3. Должны уступить дорогу легковому автомобилю и автобусу.

Далее [Вопрос 1.2.3 - выбор](#)


Вы и водитель автобуса находитесь на главной дороге (знак 2.1  "Главная дорога"), поэтому при повороте налево Вы должны уступить дорогу автобусу п. 13.12. Перед легковым автомобилем Вы имеете преимущество, так как он находится на второстепенной дороге п. 13.9

Рисунок 2 – Вопрос 2





Кому Вы обязаны уступить дорогу при повороте налево?

1. Обоим транспортным средствам.
2. Только автобусу.
3. Только легковому автомобилю.
4. Никому.

Далее - [Вопрос 1.2.3](#) - [выбор](#)



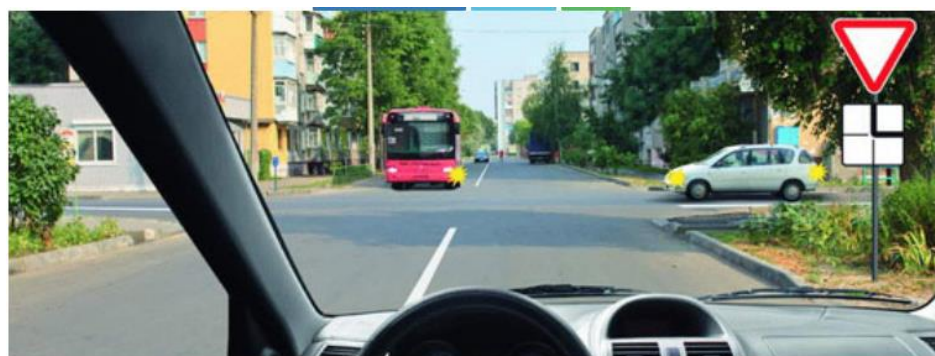
В данном случае при проезде перекрестка неравнозначных дорог следует уступить дорогу только автобусу, который, так же как и Вы, находится на главной дороге (знаки 2.1  «Главная дорога» и 8.13  «Направление главной дороги») и является для Вас «помехой справа» [п. 13.10](#) и [п. 13.11](#). Легковой автомобиль движется по второстепенной дороге и поэтому уступает дорогу Вам [п. 13.9](#).

Рисунок 3 – Вопрос 3



Кому Вы обязаны уступить дорогу при движении в прямом направлении?

1. Только легковому автомобилю.
2. Только автобусу.
3. Обоим транспортным средствам.

Далее - [Вопрос 1.2.3](#) - [выбор](#)



К перекрестку неравнозначных дорог, где главная дорога меняет направление, Вы подъезжаете по второстепенной дороге (знаки 2.4  «Уступите дорогу» и 8.13  «Направление главной дороги»), поэтому должны уступить дорогу обоим ТС, находящимся на главной дороге, независимо от направления их движения через перекресток [п. 13.9](#).

Рисунок 4 – Вопрос 4



Кому Вы обязаны уступить дорогу при повороте налево?

1. Только автобусу.
2. Только легковому автомобилю.
3. Никому.

Далее: [Вопрос 1.2.8 - выбор](#)



Проезжая данный перекресток неравнозначных дорог по направлению главной дороги (знаки 2.1  «Главная дорога» и 8.13  «Направление главной дороги»), вы никому не должны уступать дорогу, так как пользуетесь преимуществом как перед находящимся на главной дороге автобусом, для которого вы являетесь «помехой справа» [п. 13.10](#) и [п. 13.11](#), так и перед легковым автомобилем, движущимся по второстепенной дороге [п. 13.9](#).

Рисунок 5 – Вопрос 5



Кому Вы обязаны уступить дорогу при повороте налево?

1. Только автобусу.
2. Только легковому автомобилю.
3. Никому.

Далее: [Вопрос 1.2.3 - выбор](#)



При проезде данного перекрестка неравнозначных дорог по направлению главной дороги (знаки 2.1  «Главная дорога» и 8.13  «Направление главной дороги») Вам нет необходимости уступать дорогу ни автобусу, который движется по второстепенной дороге [п. 13.9](#), ни легковому автомобилю, с которым Вы разъезжаетесь по правилам проезда перекрестков равнозначных дорог [п. 13.10](#) и [п. 13.11](#), поскольку он находится слева от Вас.

Рисунок 6 – Вопрос 6



### 3 Описание используемых технологий Three.js, GSAP и Blender

Для трехмерного моделирования конкретных ситуаций из имеющихся вопросов на экзамене на водительские права были использованы такие технологии, как Three.js, GSAP и Blender.

Three.js – это библиотека JavaScript, содержащая набор готовых классов для создания и отображения интерактивной 3D графики в браузерах. Официальный сайт Three.js представлен на рисунке 7, на главной странице которого представлены примеры работ, созданные при помощи Three.js.

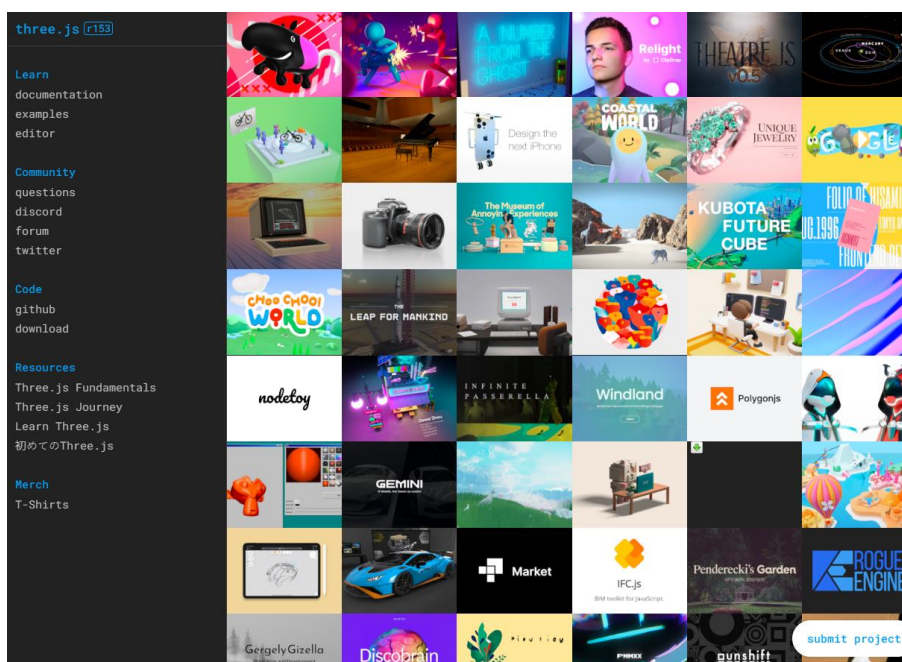


Рисунок 7 – Официальный сайт Three.js

GSAP – это набор инструментов для реализации анимации любого уровня сложности с помощью JavaScript. Официальный сайт GSAP представлен на рисунке 8.

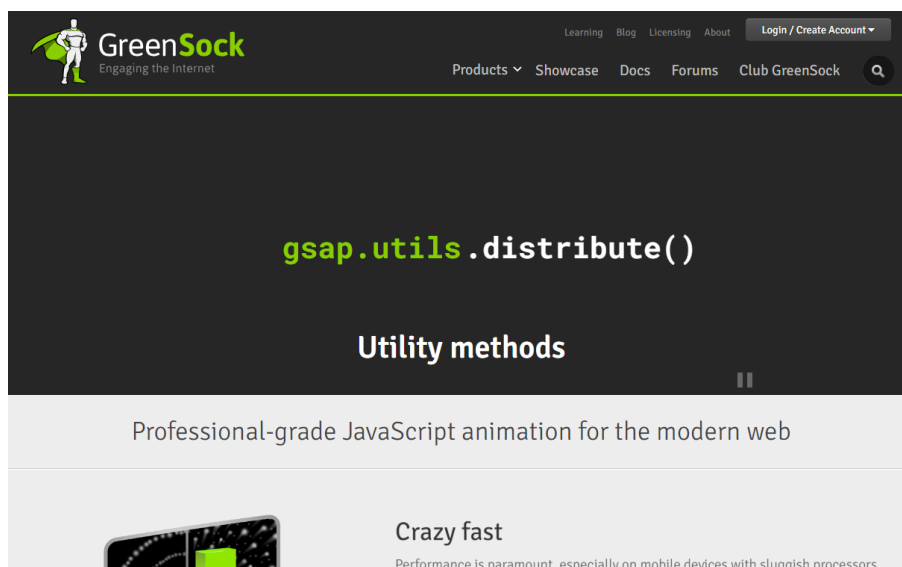


Рисунок 8 – Официальный сайт GSAP

Blender – профессиональное свободное и открытое программное обеспечение для создания трёхмерной компьютерной графики, включающее в себя средства моделирования, скульптинга, анимации, симуляции, рендеринга, постобработки и монтажа видео со звуком. На сегодняшний день является самым популярным бесплатным редактором в своей среде. На рисунке 9 представлен официальный сайт Blender.

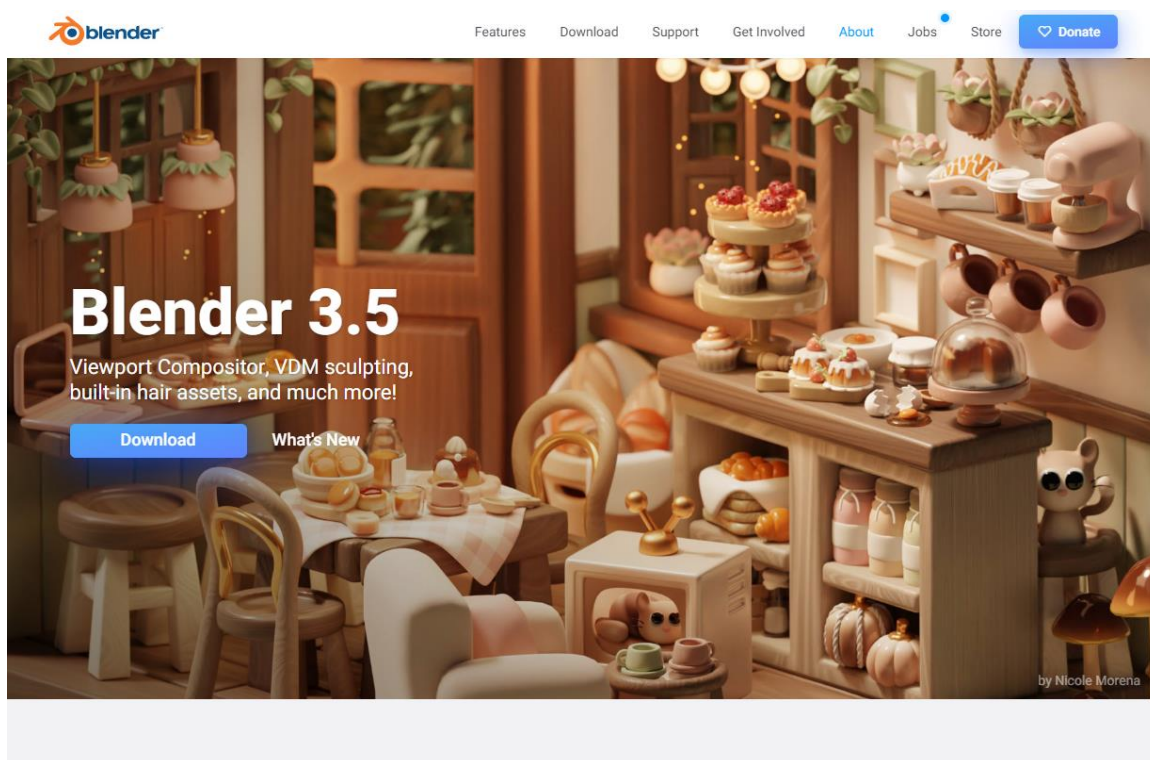


Рисунок 9 – Официальный сайт Blender

#### 4 Проектирование приложения

Для начала работы над разработкой web-приложения следует открыть Visual Code – текстовый редактор для кроссплатформенной разработки веб-приложений, и открыть

Начало работы над проектом начинается с установки менеджера пакетов, который управляет модулями и зависимостями проекта npm, командой: `npm install`. Также производится установка пакетов Yuka и GSAP, командами: `npm install yuka`; `npm install gsap`. Далее запускаем командой: `prx parcel ./src/index.html`.

В файле `index.html` находится вёрстка приложения, в котором имеются строка загрузки (см. рисунок 10), форма с количеством правильных ответов (см. рисунок 11), заголовок с кнопкой «Начать» (см. рисунок 12), блок с вопросом и ответами (см. рисунок 13) и блок с пояснением вопроса и кнопками «Завершить» и «Следующий вопрос» (см. рисунок 14).

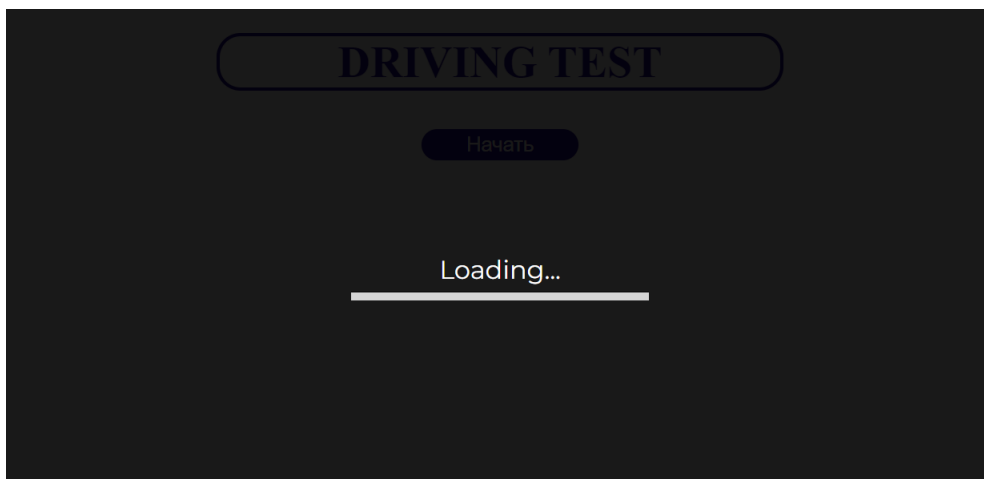


Рисунок 10 – Строка загрузки

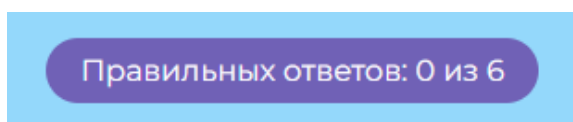


Рисунок 11 – Количество правильных ответов

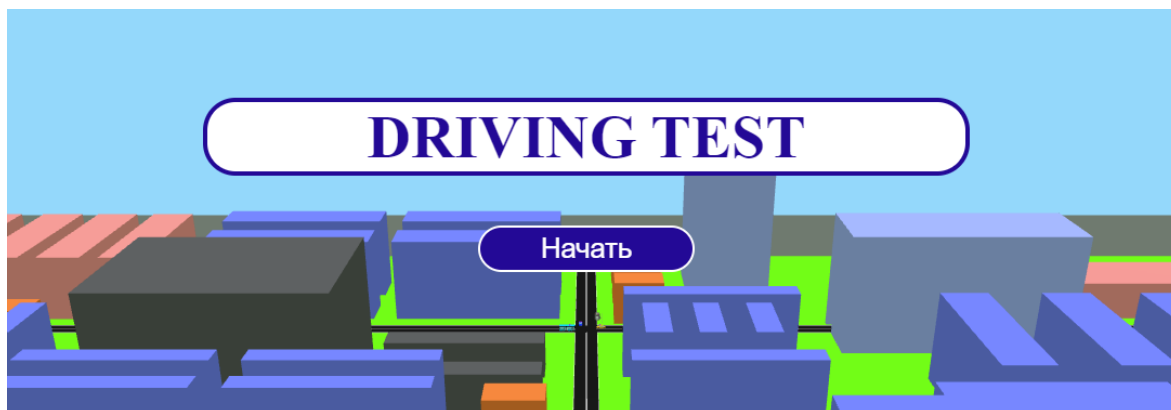


Рисунок 12 – Заголовок и кнопка «Начать»

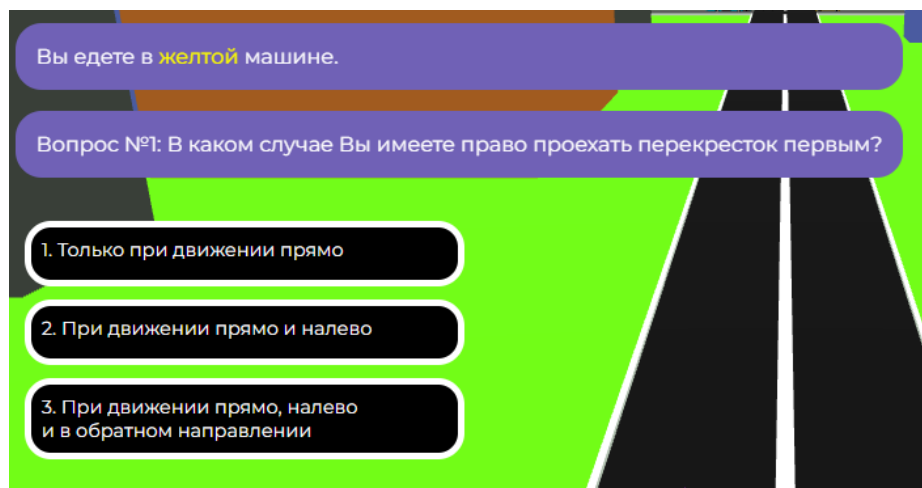


Рисунок 13 – Вопрос и ответами

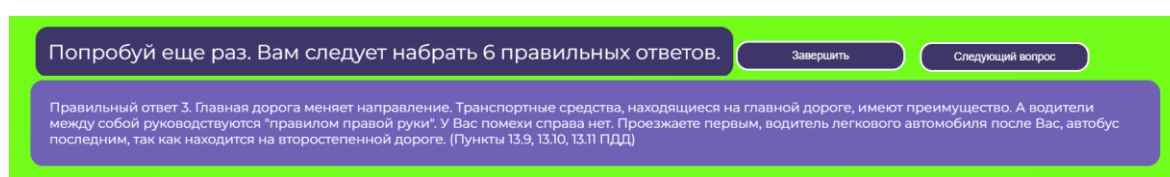


Рисунок 14 – Пояснением вопроса и кнопки «Завершить», «Следующий вопрос»

Файл index.html связан с файлами style.css и script.js. В файле style.css задаются визуальные параметры объектов, используемые в index.html.

Файл script.js связан с файлом constants.js, хранящий в себе модели машин и их координаты. Модели были созданы с помощью программы Blender (см. рисунок 15-18)



Рисунок 15 – Модель желтой машины

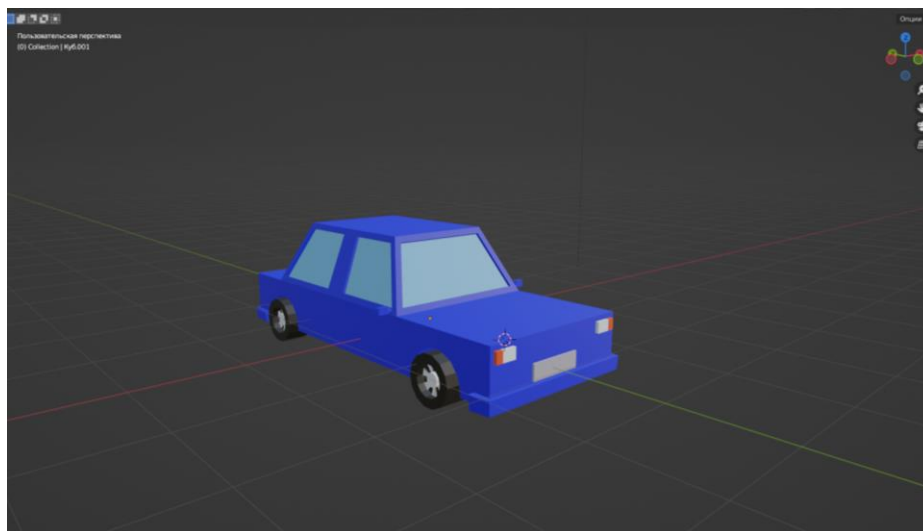


Рисунок 16 – Модель синей машины



Рисунок 17 – Модель автобуса

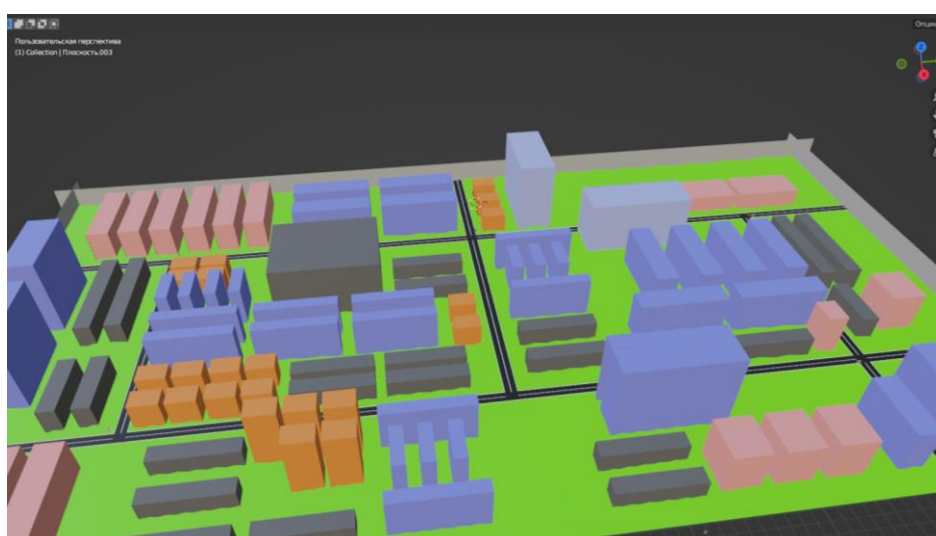


Рисунок 18 – Модель карты города и дорог



После в файле `script.js` загружаем карту города и импортируем готовые модели на эту карту, вычисляем их координаты и создаем свет на сцене (см. рисунок 19).

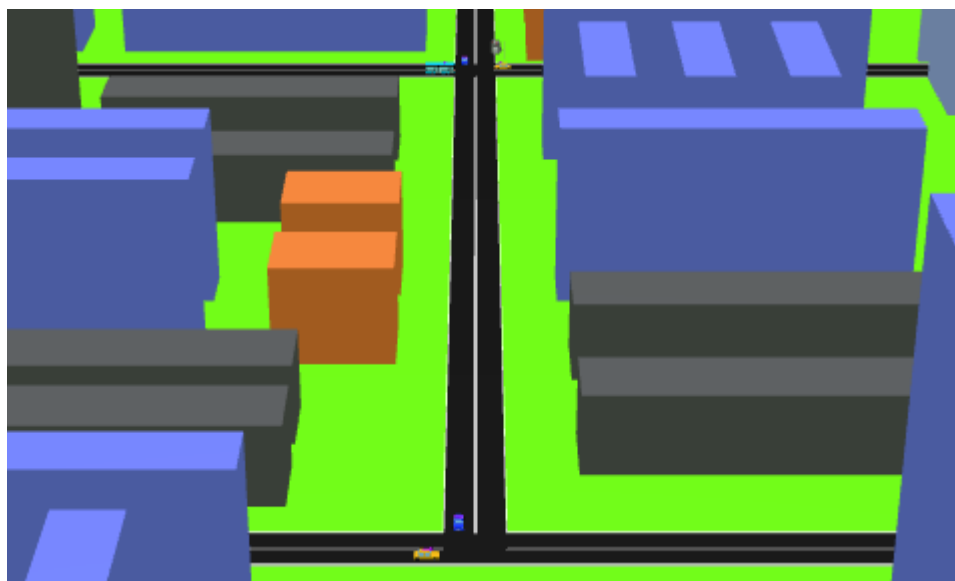


Рисунок 19 – Модель карты города и машин на ней

Следующим шагом, создаем движение камеры с помощью временной шкалы `gsap`. Траектория движения камеры начинается с первого перекрестка и движется по очереди к следующему перекрестку. Далее, при выборе ответа на вопрос проигрывается анимация движения машин на заданном перекресте дорог (см. рисунок 20).

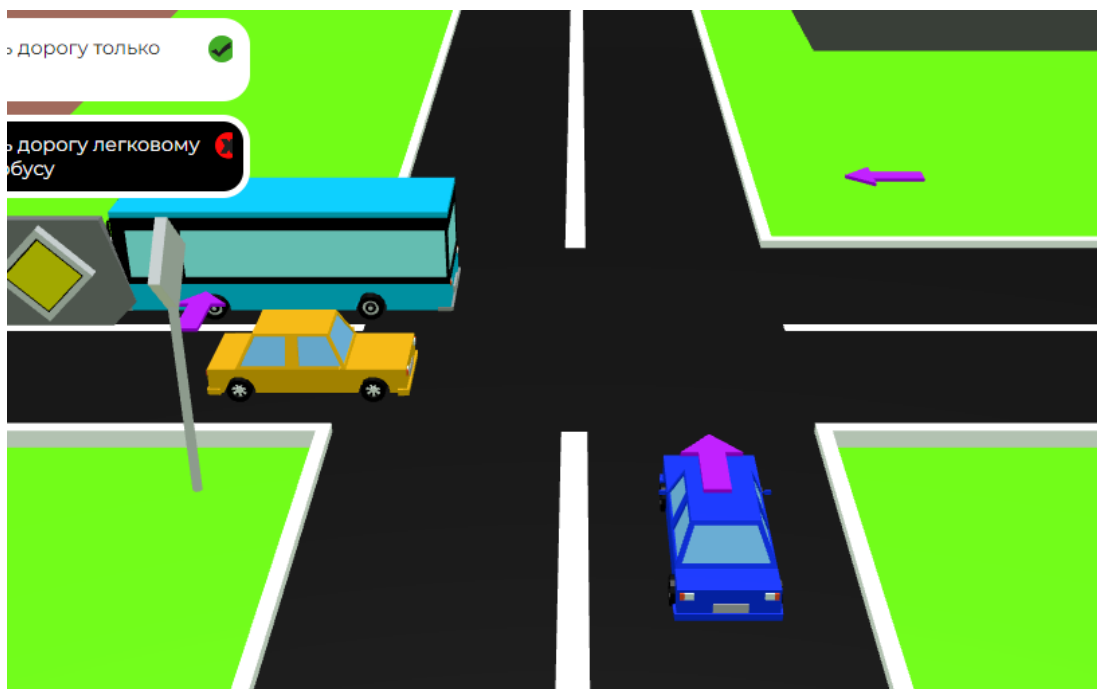


Рисунок 20 – Анимация движения машин

## 5 Тестирование разработанного приложения

Для тестирования запустим web-приложение, в стартовом окне нажмем кнопку «Начать» (см. рисунок 21).

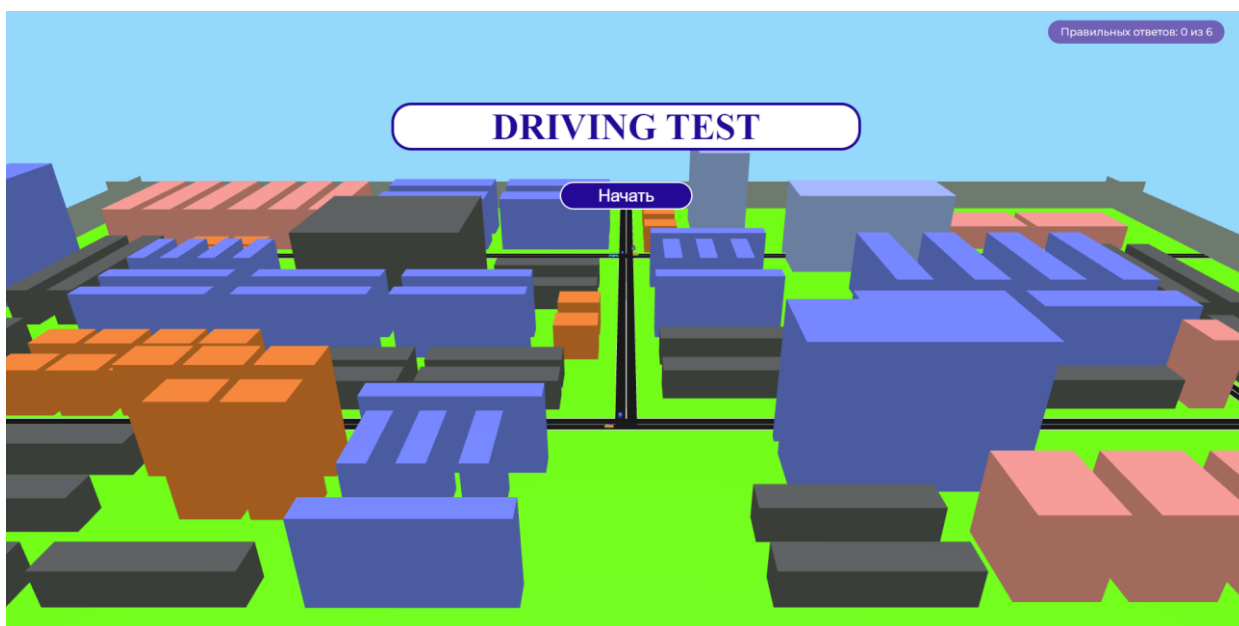


Рисунок 21 – Стартовое окно приложения

После этого камера спустится для демонстрации перекрестка, описывающий 1 вопрос. В 1 вопросе у пользователя есть 3 варианта ответа (см. рисунок 22):

1. «Только при движении прямо» – Неверный, следовательно, количество правильных ответов не изменится.
2. «При движении прямо и налево» – Верный, следовательно, количество правильных ответов изменится на 1.
3. «При движении прямо, налево и в обратном направлении» – Неверный, следовательно, количество правильных ответов не изменится.

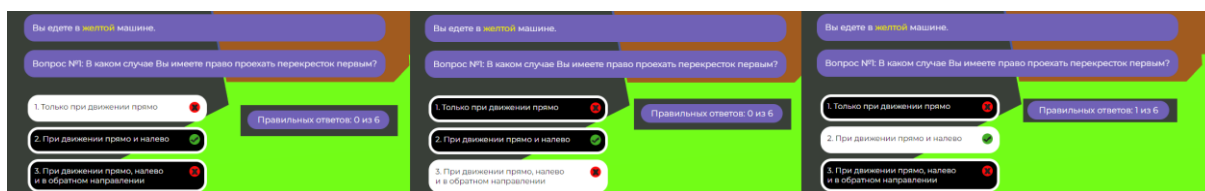


Рисунок 22 – Варианты ответов на вопрос 1

Далее камера движется для демонстрации перекрестка, описывающий вопрос 2. Во 2 вопросе у пользователя 3 варианта ответа (см. рисунок 23):

1. «Имеете преимущество» – Неверный, следовательно, количество правильных ответов не изменится.
2. «Должны уступить дорогу только автобусу» – Верный, следовательно, количество правильных ответов изменится на 1.

3. «Должны уступить дорогу легковому автомобилю и автобусу» – Неверный, следовательно, количество правильных ответов не изменится.

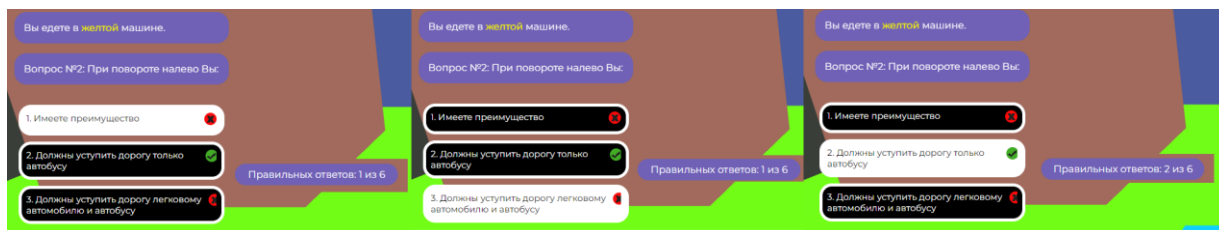


Рисунок 23 – Варианты ответов на вопрос 2

Камера движется для демонстрации перекрестка, описывающий вопрос 3. Во 3 вопросе у пользователя 3 варианта ответа (см. рисунок 24):

1. «Обоим транспортным средствам» – Неверный, следовательно, количество правильных ответов не изменится.
2. «Только автобусу» – Верный, следовательно, количество правильных ответов изменится на 1.
3. «Только легковому автомобилю» – Неверный, следовательно, количество правильных ответов не изменится.

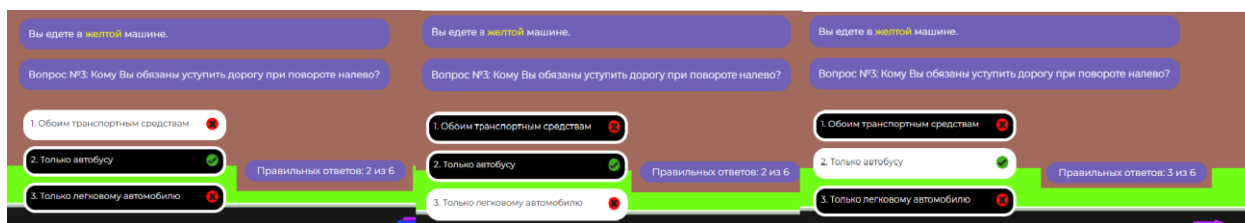


Рисунок 24 – Варианты ответов на вопрос 3

Камера движется для демонстрации перекрестка, описывающий вопрос 4. Во 4 вопросе у пользователя 3 варианта ответа (см. рисунок 25):

1. «Только легковому автомобилю» – Неверный, следовательно, количество правильных ответов не изменится.
2. «Только автобусу» – Неверный, следовательно, количество правильных ответов не изменится.
3. «Обоим транспортным средствам» – Верный, следовательно, количество правильных ответов изменится на 1.

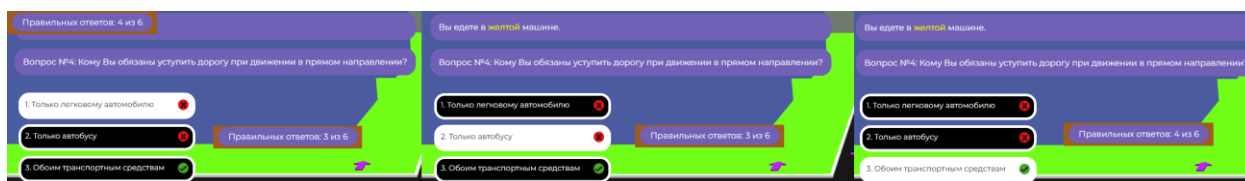


Рисунок 25 – Варианты ответов на вопрос 4

Камера движется для демонстрации перекрестка, описывающий вопрос 5. Во 5 вопросе у пользователя 3 варианта ответа (см. рисунок 26):

1. «Только автобусу» – Неверный, следовательно, количество правильных ответов не изменится.
2. «Только легковому автомобилю» – Неверный, следовательно, количество правильных ответов не изменится.
3. «Никому» – Верный, следовательно, количество правильных ответов изменится на 1.

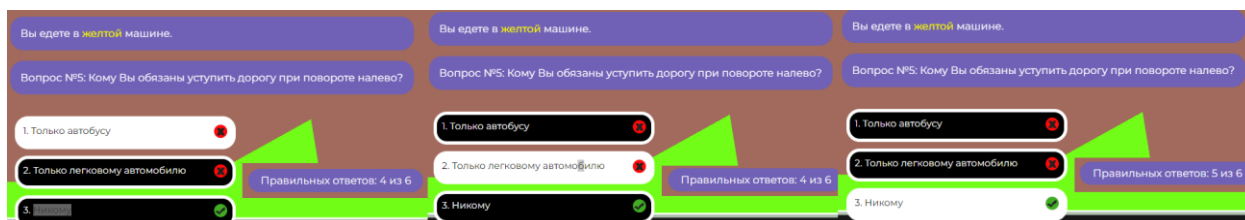


Рисунок 26 – Варианты ответов на вопрос 5

Камера движется для демонстрации перекрестка, описывающий вопрос 6. Во 6 вопросе у пользователя 3 варианта ответа (см. рисунок 27):

1. «Только автобусу» – Неверный, следовательно, количество правильных ответов не изменится.
2. «Только легковому автомобилю» – Неверный, следовательно, количество правильных ответов не изменится.
3. «Никому» – Верный, следовательно, количество правильных ответов изменится на 1.

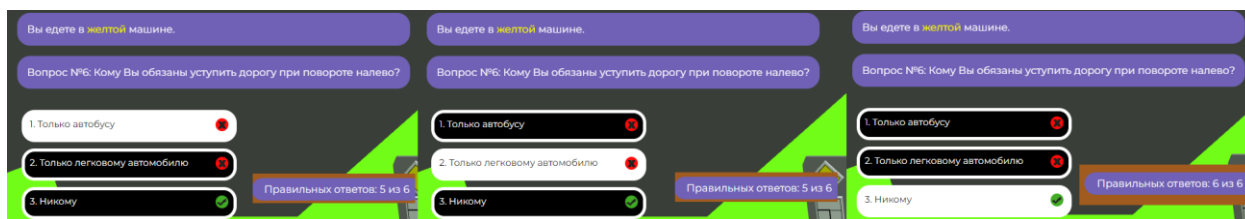


Рисунок 27 – Варианты ответов на вопрос 6

При успешном выполнении теста с итоговым количеством баллов, равным 6, появится форма с надписью (см. рисунок 28).



Рисунок 28 – Успешного выполнение теста

Если при выполнении теста не было набрано нужное количество баллов, то появится форма с надписью (см. рисунок 29).

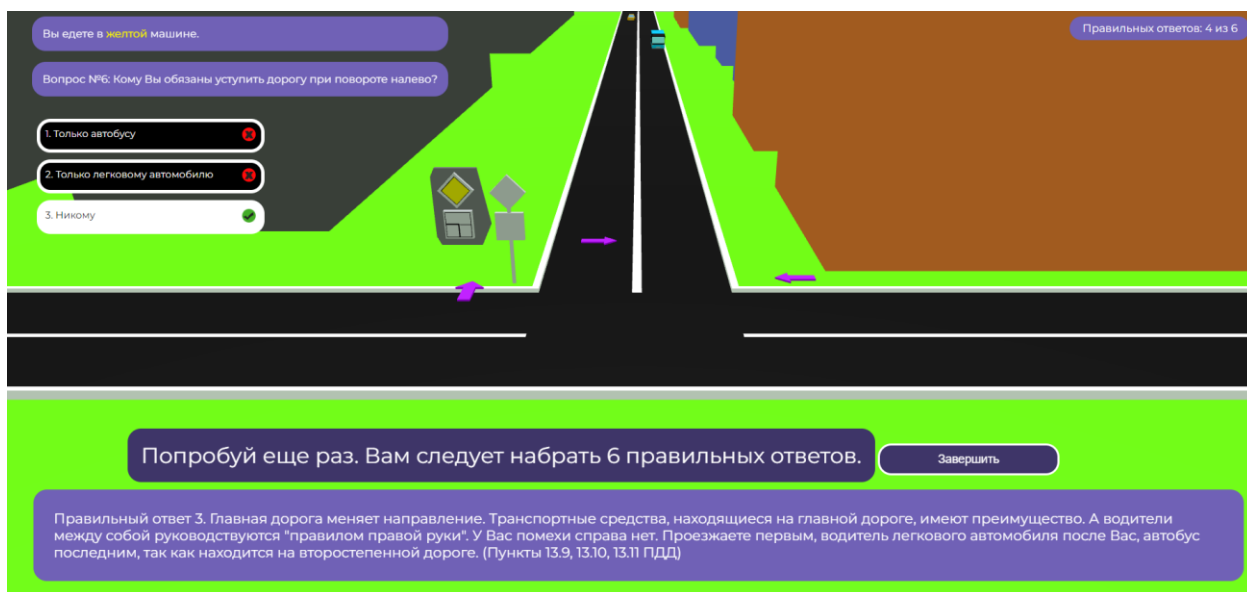


Рисунок 29 – Пример окончание теста с наличием ошибок



## 6 Инструкция пользователя

Для того чтобы начать следует нажать на кнопку «Начать», расположенную посередине экрана под заголовком (см. рисунок 30). При наведении мышки на кнопку, её цвет изменится на белый и сменится курсор.



Рисунок 30 – Главный экран web-приложения

Далее увеличится карта и появятся всплывающие формы, содержащие вопрос и выбор ответов. Выбор ответов выполняется от лица желтой машины. Стрелки над моделями машин указывают на их направление дальнейшего движения. При выборе варианта ответа, он окрасится в белый цвет, если ответ оказался правильным, то в верхнем правом углу количество правильных ответов изменится. Следом после анимации проезжающих машин, появится пояснение к правильному ответу. Чтобы перейти на следующий вопрос нажмите кнопку «Следующий вопрос» (см. рисунок 31).

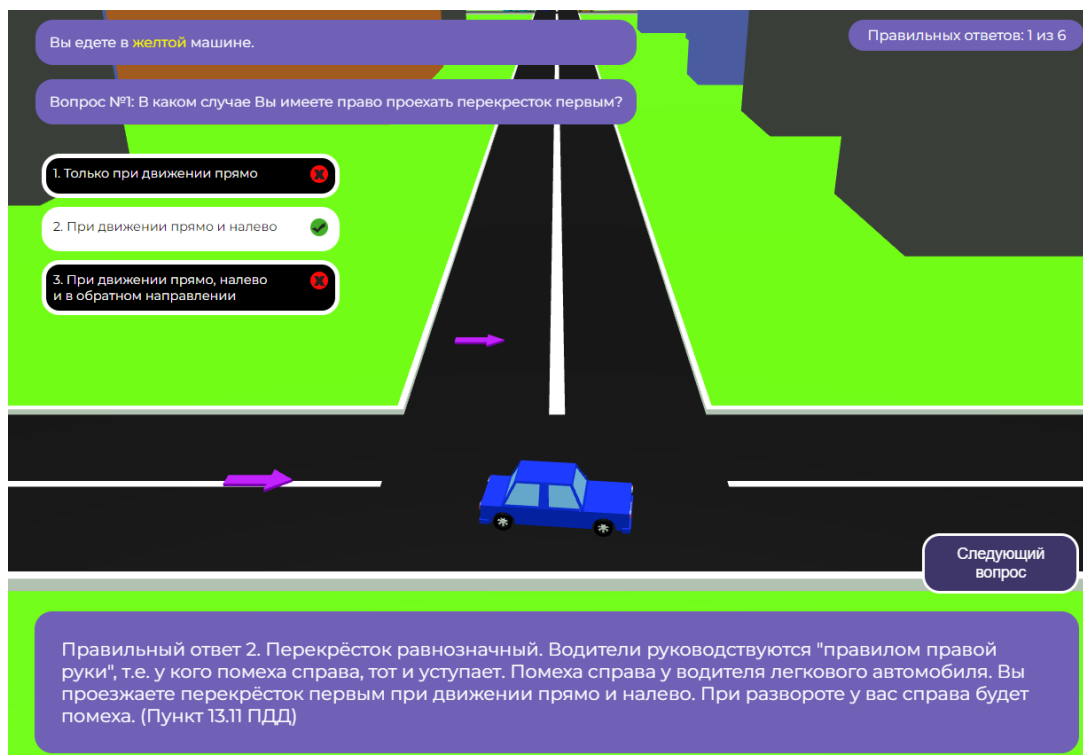


Рисунок 31 – Пример выбора ответа для вопроса 1

После нажатия кнопки «Следующий вопрос» видимая часть экрана переместится на следующий перекресток. Аналогичными действиями, как и в вопросе 1, выполняем все 6 вопросов. После ответа на 6 вопрос появится кнопку «Завершить» (см. рисунок 32).



Рисунок 32 – Пример окончания теста, после выполнения вопроса 6

При успешном выполнении теста появится форма с надписью (см. рисунок 33).

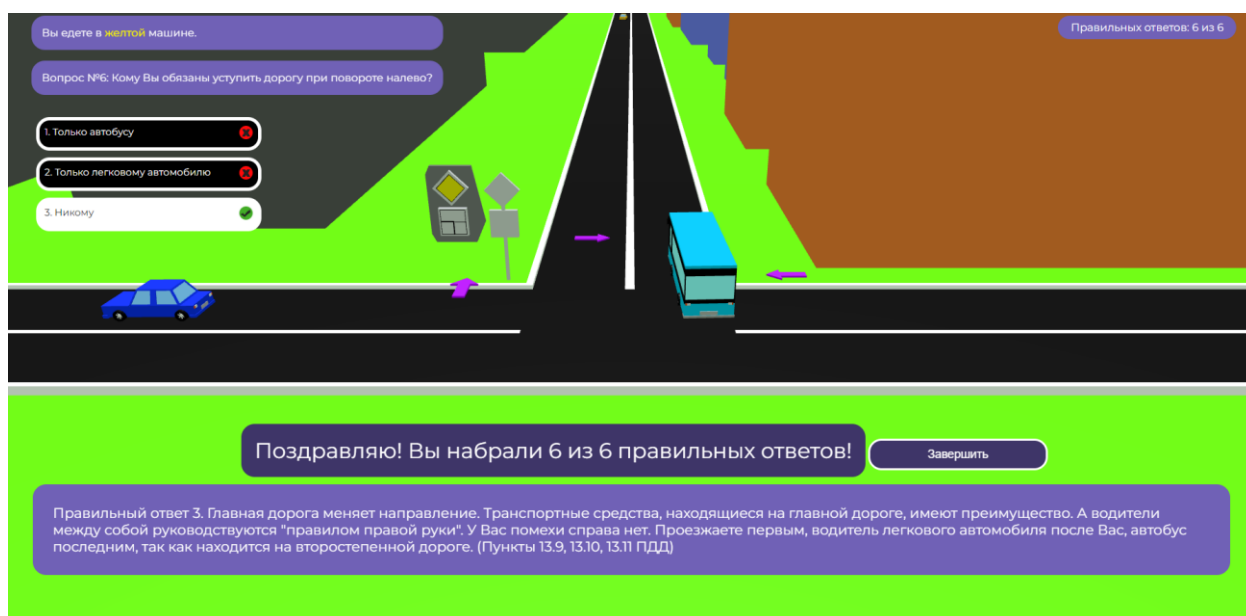


Рисунок 33 – Пример успешного окончания теста

Если при выполнении теста не было набрано нужное количество баллов, то появится форма с надписью (см. рисунок 34).

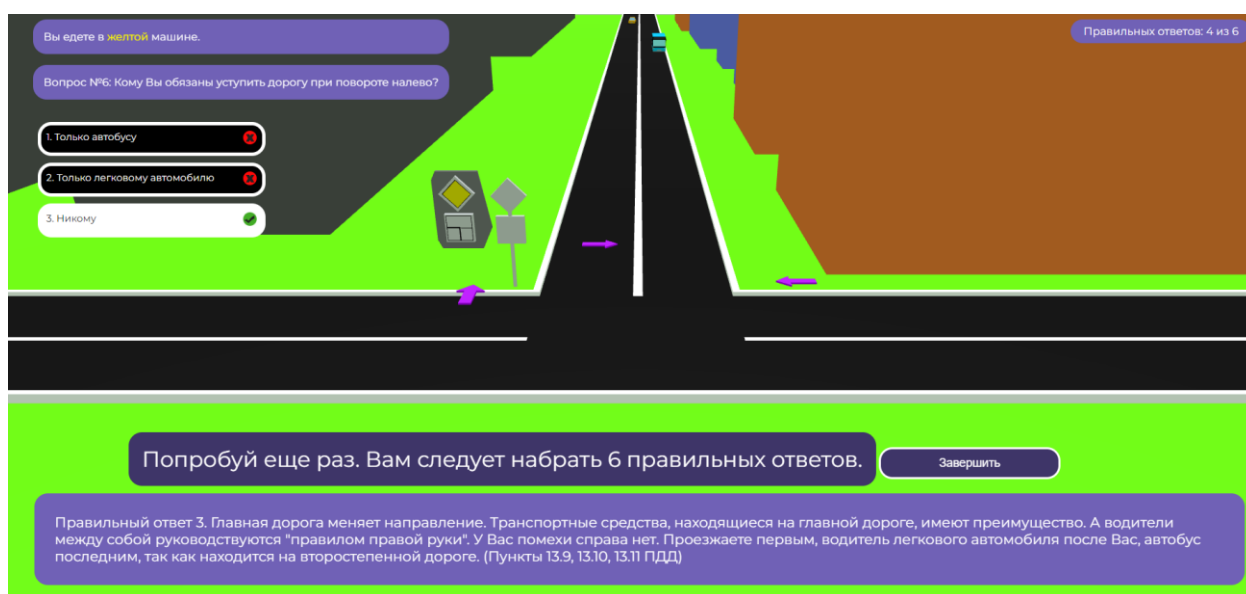


Рисунок 34 – Пример окончания теста с наличием ошибок

## **7 Анализ перспектив развития проекта**

Дальнейшими перспектива развития данного проекта является расширение пула вопросов для подготовки к сдаче экзамена на водительские права. На сегодняшний день в официальном сайте ГИБДД приводится 40 билетов по 20 вопросов, на каждый приводится от двух до пяти вариантов ответов, один из которых правильный. Теория считается не сданной, если кандидат в водители сделал две ошибки в одном тематическом блоке вопросов или более трех ошибок в разных темах. Поэтому увеличение вопросов является главной перспективой.

Следовательно, при увеличении количества вопросов нужно будет увеличить также количество трехмерных моделей и карты дорог, для более точной передачи, визуальной составляющей вопросов.

Следующим действием можно ввести аналитику и отчетность о решенных или нерешенных вопросах, о времени выполнения, о наиболее частых ошибках и т.д.

Также возможной перспективой развития является адаптации интерфейса под мобильные устройства

В завершении планируется улучшение дизайна пользовательского интерфейса веб-приложения.

## Заключение

В результате выполнения курсового проекта было создано веб-приложение для тестирования ПДД с использованием 3D-моделирования. Были изучены такие темы, как:

1. Изучение и использование библиотеки Three.js для отображения анимированной компьютерной 3D графики при разработке веб-приложения.
2. Изучение и использование библиотеки GSAP (GreenSock Animation Platform) для создания анимации на основе временной шкалы при разработке веб-приложения
3. Изучение и использование программы Blender для создания трехмерной компьютерной графики.
4. Работа с языком разметки HTML5 для создания и отображения веб-приложения.
5. Работа с языком декорирования CSS для описания внешнего вида веб-приложения.
6. Работа с языком программирования JavaScript для создания веб-приложения.
7. Изучение правил дорожного движения (ПДД) РФ и экзаменационных вопросов ГИБДД для корректного составления вопросов теста при разработке приложения.

В качестве дальнейшего развития проекта следует увеличить объём экзаменационных вопросов ГИБДД, сформировав пул хранящейся информации.

Были выполнены работы по анализу предметной области, определению актуальности и цели работы, определению концепции разрабатываемого приложения, описанию используемых технологий Three.js, GSAP и Blender, проектированию приложения, тестированию разработанного приложения и анализу перспектив развития проекта.



### Список использованных источников

1. Документация по Three.js. // URL: <https://threejs.org/docs/> (дата обращения 29.04.2023)
2. Learning Three.js: The JavaScript 3D Library for WebGL. Published by Packt Publishing Ltd, ISBN 978-1-78439-221-5, 2015 г.
3. Документация по GSAP. // URL: <https://greensock.com/docs/> (дата обращения 29.04.2023)
4. Официальный сайт Three.js. // URL: <https://threejs.org> (дата обращения 28.04.2023)
5. Официальный сайт GSAP. // URL: <https://greensock.com/gsap/> (дата обращения 28.04.2023)
6. Официальный сайт Blender. // URL: <https://www.blender.org> (дата обращения 28.04.2023)
7. Документация по Blender. // URL: <https://docs.blender.org> (дата обращения 29.04.2023)
8. HTML5, CSS3 и JavaScript. Исчерпывающее руководство / Дженнифер Роббинс; [пер. с англ. М. А. Райтман]. — 4-е издание. — М. : Эксмо, 2014. — 528 с.
9. Документация по языку программирования JavaScript // URL: <https://developer.mozilla.org/ru/docs/Web/JavaScript> (дата обращения 10.05.2023)
10. ПДД 2023 России (Правила дорожного движения). // URL: <https://www.pdd24.com> (дата обращения 30.04.2023)
11. Экзаменационные билеты ГИБДД. // URL: <https://гибдд.пф/mens/avtovladeltsam/abm> (дата обращения 30.04.2023)

## Листинг

### Файл index.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Arbakova WEB-6 semestr KP</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <section class="progress-bar-container">
    <label for="progress-bar">Loading...</label>
    <progress id="progress-bar" value="0" max="100"></progress>
  </section>
  <p class="score">Правильных ответов: <span>0</span> из 6</p>

  <section class="header">
    <h1>DRIVING TEST</h1>
    <button> Начать </button>
  </section>
  <section class="questions">
    <div>Вы едете в <span>желтой</span> машине.</div>
    <p>Вопрос №1: В каком случае Вы имеете право проехать перекресток первым?</p> <!--(билет 11, вопрос 14)-->
    <ul>
      <li id="option1"><span id="a1-text">1. Только при движении прямо</span> <span id="a1-symbol"></span></li>
      <li id="option2"><span id="a2-text">2. При движении прямо и налево</span> <span id="a2-symbol"></span></li>
      <li id="option3"><span id="a3-text">3. При движении прямо, налево<br> и в обратном направлении</span> <span id="a3-symbol"></span></li>
    </ul>
  </section>
  <section class="explanation">
    <span id="congratulation">Congratulations!</span>
    <button id="2">Завершить</button>
    <button id="1">Следующий вопрос</button>
    <div class="text-wrapper" id="text-wrapper">
      <p>
        Правильный ответ 2.
        Перекрёсток равнозначный. Водители руководствуются "правилом правой руки",
        т.е. у кого помеха справа, тот и уступает. Помеха справа у водителя легкового автомобиля.
        Вы проезжаете перекрёсток первым при движении прямо и налево. При развороте у вас справа будет помеха.
        (Пункт 13.11 ПДД)
      </p>
    </div>
  </section>
  <script src="scripts.js" type="module"></script>
</body>
</html>
```

### Файл style.css:

```
@import url('https://fonts.googleapis.com/css2?family=Montserrat&display=swap');
body {
  margin: 0;
  font-family: 'Montserrat', sans-serif;
}
.progress-bar-container {
  position: absolute;
  left: 50%;
  top: 50%;
  transform: translate(-50%, -50%);
  width: 100%;
  height: 100%;
  background-color: rgba(0, 0, 0, 0.9);
  display: flex;
  flex-direction: column;
  justify-content: center;
  align-items: center;
  z-index: 1000;
}
#progress-bar {
  width: 30%;
  height: 2%;
  margin-top: 0.5%;
}
label {
  color: white;
```

```

    font-size: 2rem;
}
.header {
    position: absolute;
    left: 50%;
    top: 10%;
    transform: translateX(-50%);
    text-align: center;
    z-index: 100;
}
.header h1 {
    font-family: 'Montserrat Extrabold';
    font-size: 3.5rem;
    color: rgb(35, 9, 150);
    background-color: rgb(255, 255, 255);
    border-radius: 30px;
    width: 700px;
    border: 4px solid rgb(35, 9, 150);
}
.header button {
    font-size: 1.6rem;
    background-color: rgb(35, 9, 150);
    color: white;
    padding: 0.3rem;
    cursor: pointer;
    border-radius: 30px;
    width: 200px;
    border: 2px solid rgb(255, 255, 255);
}
.header button:hover {
    background-color: white;
    color: rgb(35, 9, 150);
}
.score {
    position: absolute;
    right: 2%;
    font-size: 1rem;
    background-color: rgb(112, 97, 182);
    color: white;
    padding: 0.5rem;
    border-radius: 20px;
    width: 250px;
    text-align: center;
}
.questions {
    position: absolute;
    top: 1%;
    left: 2%;
}
.questions p, ul, div {
    color: rgb(255, 255, 255);
    font-size: 1rem;
}
.questions p {
    padding: 1rem;
    background-color: rgb(112, 97, 182);
    opacity: 0;
    border-radius: 20px;
}
.questions div {
    padding: 1rem;
    background-color: rgb(112, 97, 182);
    opacity: 0;
    border-radius: 20px;
}
.questions div span {
    color: yellow;
}
.questions, ul {
    list-style: none;
    padding: 7;
    margin-top: 0;
    display: inline-block;
    font-size: 0.9rem;
}
.questions ul li {
    cursor: pointer;
    padding: 0.5rem;
}

```

```

background-color: rgb(0, 0, 0);
transform: rotateX(90deg);
border-radius: 20px;
margin: 10px 0;
display: flex;
border: 5px solid rgb(255, 255, 255);
margin-right: 5%;
width: 310px;
}
.questions ul li span:nth-child(2) {
background-size: cover;
width: 20px;
height: 20px;
display: inline-block;
margin-bottom: 4px;
margin-left: auto;
margin-right: 0%;
}
.explanation {
position: absolute;
left: 50%;
transform: translateX(-50%);
bottom: 1%;
width: 95%;
text-align: right;
opacity: 0;
}
.explanation button {
font-size: 1.1rem;
background-color: rgb(62, 53, 104);
border: 3px solid rgb(255, 255, 255);
color: rgb(255, 255, 255);
border-radius: 20px;
padding: 10px;
cursor: pointer;
width: 15%;
}
.explanation button:hover {
background-color: rgb(255, 255, 255);
border: 3px solid rgb(62, 53, 104);
color: rgb(62, 53, 104);
}
.explanation .text-wrapper p {
font-size: 1.3rem;
background-color: rgb(112, 97, 182);
color: white;
padding: 0.8rem;
text-align: left;
border-radius: 20px;
padding: 30px;
}
.explanation span {
display: center;
font-size: 2rem;
background-color: rgb(62, 53, 104);
color: white;
border-radius: 20px;
text-align: center;
padding: 20px;
}

```

## Файл script.js:

```

import * as THREE from 'three';
import * as YUKA from 'yuka';
import gsap from 'gsap';
import { GLTFLoader } from 'three/examples/jsm/loaders/GLTFLoader';
import { DRACOLoader } from 'three/examples/jsm/loaders/DRACOLoader';
import * as SkeletonUtils from 'three/examples/jsm/utils/SkeletonUtils';
import {
YELLOWVEHICLESPATHS,
BLUEVEHICLESPATHS,
BUSVEHICLESPATHS,
ANSWERSTEXT
} from './constants';
const entityManager = new YUKA.EntityManager();
const progressBar = document.getElementById('progress-bar');
const progressBarContainer = document.querySelector('.progress-bar-container');
const loadingManager = new THREE.LoadingManager();

```

```

const startButton = document.querySelector('.header button');
const title = document.querySelector('.header h1');
const explanation = document.querySelector('.explanation');
const nextQuestionBtn = document.getElementById('1');
const finish = document.getElementById('2');
finish.style.visibility = 'hidden';
const congratulation = document.getElementById('congratulation');
congratulation.style.visibility = 'hidden';
const question = document.querySelector('.questions p');
const question1 = document.querySelector('.questions div');
const option1 = document.getElementById('option1');
const option2 = document.getElementById('option2');
const option3 = document.getElementById('option3');
const option1Symbol = document.getElementById('a1-symbol');
const option2Symbol = document.getElementById('a2-symbol');
const option3Symbol = document.getElementById('a3-symbol');
const option1Text = document.getElementById('a1-text');
const option2Text = document.getElementById('a2-text');
const option3Text = document.getElementById('a3-text');
const explanation1 = document.querySelector('.explanation .text-wrapper p');

let clicked = false;
let questionNumber = 1;
let cameraX = 140;
let cameraY = 15;
let cameraZ = 0;

const yellowCars = [];
const blueCars = [];
const busCars = [];
let carToAnimate = 0;
const score = document.querySelector('.score span');
let scoreValue = 0;

const renderer = new THREE.WebGLRenderer({ antialias: true });
renderer.setSize(window.innerWidth, window.innerHeight);
document.body.appendChild(renderer.domElement);

// Sets the color of the background
renderer.setClearColor(0x94D8FB);

const scene = new THREE.Scene();
const camera = new THREE.PerspectiveCamera(
    45,
    window.innerWidth / window.innerHeight,
    0.1,
    1000
);
// Camera positioning
camera.position.set(420, 200, 0);
camera.lookAt(scene.position);

const ambientLight = new THREE.AmbientLight(0xE1E1E1, 0.3);
scene.add(ambientLight);

const hemisphereLight = new THREE.HemisphereLight(0x94D8FB, 0x9CFF2E, 0.3);
scene.add(hemisphereLight);

const directionalLight = new THREE.DirectionalLight(0xFFFFFF, 0.7);
scene.add(directionalLight);

renderer.outputEncoding = THREE.sRGBEncoding;

loadingManager.onProgress = function(url, loaded, total) {
    progressBar.value = (loaded/total) * 100;
}
loadingManager.onLoad = function() {
    progressBarContainer.style.display = 'none';
}

const loader = new GLTFLoader(loadingManager);
const dLoader = new DRACOLoader();
dLoader.setDecoderPath('http://www.gstatic.com/draco/versioned/decoders/1.5.6/');
dLoader.setDecoderConfig({ type: 'js' });
loader.setDRACOLoader(dLoader);

loader.load('./assets/road2.glb', function(glb) {
    const model = glb.scene;

```



```

    scene.add(model);
  });

function sync(entity, renderComponent){
  renderComponent.matrix.copy(entity.worldMatrix);
}

function createCarV(model, path, entityManager, yRotation){
  const group = new THREE.Group();
  scene.add(group);
  group.matrixAutoUpdate = false;

  const car = SkeletonUtils.clone(model);
  group.add(car);

  const v = new YUKA.Vehicle();
  v.setRenderComponent(group, sync);

  entityManager.add(v);

  const followPathBehavior = new YUKA.FollowPathBehavior(path, 2);
  const onPathBehavior = new YUKA.OnPathBehavior(path);
  onPathBehavior.radius = 0.1;

  v.position.copy(path.current());
  v.maxSpeed = 5;
  v.steering.add(onPathBehavior);
  v.steering.add(followPathBehavior);

  followPathBehavior.active = false;

  v.rotation.fromEuler(0, yRotation, 0);

  const vehicleAll = { vehicle: v, modelGroup: car };
  return vehicleAll;
}

loader.load('./assets/car1.glb', function(glb) {
  const model = glb.scene;
  const v1 = createCarV(model, YELLOWVEHICLESPATHS[0], entityManager, Math.PI);
  const v2 = createCarV(model, YELLOWVEHICLESPATHS[1], entityManager, Math.PI);
  const v3 = createCarV(model, YELLOWVEHICLESPATHS[2], entityManager, Math.PI*1.5);
  const v4 = createCarV(model, YELLOWVEHICLESPATHS[3], entityManager, 0);
  const v5 = createCarV(model, YELLOWVEHICLESPATHS[4], entityManager, 0);
  const v6 = createCarV(model, YELLOWVEHICLESPATHS[5], entityManager, Math.PI/2);
  yellowCars.push(v1, v2, v3, v4, v5, v6);
});

loader.load('./assets/car2.glb', function(glb) {
  const model = glb.scene;
  const v1 = createCarV(model, BLUEVEHICLESPATHS[0], entityManager, Math.PI/2);
  const v2 = createCarV(model, BLUEVEHICLESPATHS[1], entityManager, Math.PI*1.5);
  const v3 = createCarV(model, BLUEVEHICLESPATHS[2], entityManager, Math.PI);
  const v4 = createCarV(model, BLUEVEHICLESPATHS[3], entityManager, Math.PI/2);
  const v5 = createCarV(model, BLUEVEHICLESPATHS[4], entityManager, Math.PI);
  const v6 = createCarV(model, BLUEVEHICLESPATHS[5], entityManager, 0);
  blueCars.push(v1, v2, v3, v4, v5, v6);
});

loader.load('./assets/bus.glb', function(glb) {
  const model = glb.scene;
  const v1 = createCarV(model, BUSVEHICLESPATHS[0], entityManager, 0);
  const v2 = createCarV(model, BUSVEHICLESPATHS[1], entityManager, 0);
  const v3 = createCarV(model, BUSVEHICLESPATHS[2], entityManager, Math.PI);
  const v4 = createCarV(model, BUSVEHICLESPATHS[3], entityManager, Math.PI*1.5);
  const v5 = createCarV(model, BUSVEHICLESPATHS[4], entityManager, Math.PI);
  busCars.push(v1, v2, v3, v4, v5);
});

startButton.addEventListener('mousedown', function(){
  const t1 = gsap.timeline();
  t1.to(startButton, {
    autoAlpha: 0,
    y: '-=20',
    duration: 2
  })
  .to(title, {
    autoAlpha: 0,

```

```

        y: '-20',
        duration: 2
    }, 0)
    .to(camera.position, {
        x: 140,
        y: 15,
        z: 0,
        duration: 4,
    }, 0)
    .to(question1, {
        autoAlpha: 1,
        duration: 0.2
    }, '+=0.5')
    .to(question, {
        autoAlpha: 1,
        duration: 0.2
    }, '+=0.5')
    .to(option1, {
        rotateX: 0,
        duration: 0.2
    }, '+=0.5')
    .to(option2, {
        rotateX: 0,
        duration: 0.2
    }, '+=0')
    .to(option3, {
        rotateX: 0,
        duration: 0.2
    }, '+=0')
    });

loader.load('./assets/arrow.glb', function(glb) {
    const model = glb.scene;

    function createArrow(position, yRotation = 0) {
        const arrow = SkeletonUtils.clone(model);
        arrow.position.copy(position);
        arrow.rotation.y = yRotation;
        scene.add(arrow);
    }

    createArrow(new THREE.Vector3(118, 2, 8), Math.PI);
    createArrow(new THREE.Vector3(108, 2, 2.8), Math.PI);

    createArrow(new THREE.Vector3(118, 2, -307), Math.PI*1.5);
    createArrow(new THREE.Vector3(124, 2, -317.3), Math.PI*1.5);
    createArrow(new THREE.Vector3(114, 3.5, -322), 0);

    createArrow(new THREE.Vector3(-105, 2, -317.5), 0);
    createArrow(new THREE.Vector3(-112, 2, -307), Math.PI);
    createArrow(new THREE.Vector3(-117, 3.5, -323), Math.PI/2);

    createArrow(new THREE.Vector3(-116, 2, -7), 0);
    createArrow(new THREE.Vector3(-120, 2, 3), Math.PI);
    createArrow(new THREE.Vector3(-112, 3.5, 8), Math.PI*1.5);

    createArrow(new THREE.Vector3(-116, 2, 307), Math.PI/2);
    createArrow(new THREE.Vector3(-112, 2, 322), Math.PI);
    createArrow(new THREE.Vector3(-102, 3.5, 312.2), Math.PI);

    createArrow(new THREE.Vector3(108, 2, 317), Math.PI);
    createArrow(new THREE.Vector3(113, 2, 308), 0);
    createArrow(new THREE.Vector3(118, 3.5, 321.5), Math.PI*1.5);
    });

function showAnswerSymbol(opt1, opt2, opt3) {
    option1Symbol.style.backgroundImage = `url('./assets/symbols/${opt1}.png')`;
    option2Symbol.style.backgroundImage = `url('./assets/symbols/${opt2}.png')`;
    option3Symbol.style.backgroundImage = `url('./assets/symbols/${opt3}.png')`;
}

function animateCar(delay, car, wheels, last) {
    setTimeout(function() {
        car.vehicle.steering.behaviors[1].active = true;

        if (last)
            carToAnimate++;
    }, delay);
}

```

```

}

function chooseAnswer(option) {
  if (!clicked) {
    switch (carToAnimate) {
      case 0:
        showAnswerSymbol('incorrect', 'correct', 'incorrect');
        animateCar(0, yellowCars[carToAnimate], null);
        animateCar(3000, blueCars[carToAnimate], null, true);
        if (option.id === 'option2') {
          scoreValue++;
          score.innerText = scoreValue;
        }
        break;
      case 1:
        showAnswerSymbol('incorrect', 'correct', 'incorrect');
        animateCar(3000, yellowCars[carToAnimate], null);
        animateCar(7000, blueCars[carToAnimate], null, true);
        animateCar(0, busCars[carToAnimate-1], null);
        if (option.id === 'option2') {
          scoreValue++;
          score.innerText = scoreValue;
        }
        break;
      case 2:
        showAnswerSymbol('incorrect', 'correct', 'incorrect');
        animateCar(3000, yellowCars[carToAnimate], null);
        animateCar(7000, blueCars[carToAnimate], null, true);
        animateCar(0, busCars[carToAnimate-1], null);
        if (option.id === 'option2') {
          scoreValue++;
          score.innerText = scoreValue;
        }
        break;
      case 3:
        showAnswerSymbol('incorrect', 'incorrect', 'correct');
        animateCar(7000, yellowCars[carToAnimate], null, true);
        animateCar(4000, blueCars[carToAnimate], null);
        animateCar(0, busCars[carToAnimate-1], null);
        if (option.id === 'option3') {
          scoreValue++;
          score.innerText = scoreValue;
        }
        break;
      case 4:
        showAnswerSymbol('incorrect', 'incorrect', 'correct');
        animateCar(0, yellowCars[carToAnimate], null);
        animateCar(7000, blueCars[carToAnimate], null, true);
        animateCar(3000, busCars[carToAnimate-1], null);
        if (option.id === 'option3') {
          scoreValue++;
          score.innerText = scoreValue;
        }
        break;
      case 5:
        showAnswerSymbol('incorrect', 'incorrect', 'correct');
        animateCar(0, yellowCars[carToAnimate], null);
        animateCar(3000, blueCars[carToAnimate], null);
        animateCar(5000, busCars[carToAnimate-1], null, true);
        if (option.id === 'option3') {
          scoreValue++;
          score.innerText = scoreValue;
        }
        break;
      default:
        break;
    }

    option.style.backgroundColor = 'white';
    option.style.color = 'black';
    gsap.to(explanation, {
      autoAlpha: 1,
      y: '-=10',
      duration: 0.5
    }, '+=5')
    clicked = true;
  }
}

```

```

option1.addEventListener('click', chooseAnswer.bind(null, option1));
option2.addEventListener('click', chooseAnswer.bind(null, option2));
option3.addEventListener('click', chooseAnswer.bind(null, option3));
function changeColors() {
    option1.style.backgroundColor = 'black';
    option1.style.color = 'white';
    option2.style.backgroundColor = 'black';
    option2.style.color = 'white';
    option3.style.backgroundColor = 'black';
    option3.style.color = 'white';

    option1Symbol.style.backgroundImage = "";
    option2Symbol.style.backgroundImage = "";
    option3Symbol.style.backgroundImage = "";
}
function changeOptionsText(qtion, opt1, opt2, opt3, exp) {
    question.textContent = qtion;
    option1Text.textContent = opt1;
    option2Text.textContent = opt2;
    option3Text.textContent = opt3;
    explanation1.textContent = exp;
}
nextQuestionBtn.addEventListener('click', function() {
    questionNumber++;
    const t1 = gsap.timeline();
    switch (questionNumber) {
        case 2:
            cameraX = 145;
            cameraZ = -315;
            break;
        case 3:
            cameraX = -75;
            cameraZ = -315;
            break;
        case 4:
            cameraX = -75;
            cameraZ = 0;
            break;
        case 5:
            cameraX = -75;
            cameraZ = 315;
            break;
        case 6:
            cameraX = 145;
            cameraZ = 315;
            nextQuestionBtn.disabled = true;
            nextQuestionBtn.style.visibility = 'hidden';
            finish.style.visibility = 'visible';
            break;
        default:
            break;
    }
    t1.to(camera.position, {
        x: cameraX,
        z: cameraZ,
        duration: 2
    }, 0)
    .to(question, {
        autoAlpha: 0,
        duration: 0.2
    }, 0)
    .to(explanation, {
        autoAlpha: 0,
        duration: 0.5
    }, 0)
    .to(option1, {
        rotateX: 90,
        duration: 0.2
    }, 0)
    .to(option2, {
        rotateX: 90,
        duration: 0.2
    }, 0)
    .to(option3, {
        rotateX: 90,
        duration: 0.2,
        onComplete: function() {
            changeColors();

```

```

        changeOptionsText(
            ANSWERSTEXT[questionNumber - 1].question,
            ANSWERSTEXT[questionNumber - 1].answer1,
            ANSWERSTEXT[questionNumber - 1].answer2,
            ANSWERSTEXT[questionNumber - 1].answer3,
            ANSWERSTEXT[questionNumber - 1].explanation,
        );
    }
}, 0)
.to(question, {
    autoAlpha: 1,
    duration: 0.2
}, '+=0.5')
.to(option1, {
    rotateX: 0,
    duration: 0.2
}, '+=0.5')
.to(option2, {
    rotateX: 0,
    duration: 0.2
}, '+=0')
.to(option3, {
    rotateX: 0,
    duration: 0.2,
    onComplete: function() {
        clicked = false;
    }
}, '+=0')
});
finish.addEventListener('click', function(){
    congratulation.style.visibility = 'visible';
    if (scoreValue === 6)
        congratulation.textContent = 'Поздравляю! Вы набрали 6 из 6 правильных ответов!';
    else
        congratulation.textContent = 'Попробуй еще раз. Вам следует набрать 6 правильных ответов.';
});
const time = new YUKA.Time();
function animate() {
    const delta = time.update().getDelta();
    entityManager.update(delta);
    renderer.render(scene, camera);
}
renderer.setAnimationLoop(animate);

window.addEventListener('resize', function() {
    camera.aspect = window.innerWidth / window.innerHeight;
    camera.updateProjectionMatrix();
    renderer.setSize(window.innerWidth, window.innerHeight);
});

```

## Файл constants.js:

```

import {Path, Vector3} from 'yuka';
const YELLOWVEHICLESPATHS = [];
const BLUEVEHICLESPATHS = [];
const BUSVEHICLESPATHS = [];
const yellowV1 = new Path();
yellowV1.add(new Vector3(118, 0.7, 8));
yellowV1.add(new Vector3(118, 0.7, -200));
YELLOWVEHICLESPATHS.push(yellowV1);
const yellowV2 = new Path();
yellowV2.add(new Vector3(118, 0.7, -307));
yellowV2.add(new Vector3(118, 0.7, -317.5));
yellowV2.add(new Vector3(-50, 0.7, -317.5));
YELLOWVEHICLESPATHS.push(yellowV2);
const yellowV3 = new Path();
yellowV3.add(new Vector3(-105, 0.7, -317.5));
yellowV3.add(new Vector3(-116, 0.7, -317.5));
yellowV3.add(new Vector3(-116, 0.7, -200));
YELLOWVEHICLESPATHS.push(yellowV3);
const yellowV4 = new Path();
yellowV4.add(new Vector3(-116, 0.7, -7));
yellowV4.add(new Vector3(-116, 0.7, 50));
YELLOWVEHICLESPATHS.push(yellowV4);
const yellowV5 = new Path();
yellowV5.add(new Vector3(-116, 0.7, 307));
yellowV5.add(new Vector3(-116, 0.7, 317));
yellowV5.add(new Vector3(-70, 0.7, 317));
YELLOWVEHICLESPATHS.push(yellowV5);

```

```

const yellowV6 = new Path();
yellowV6.add(new Vector3(108, 0.7, 317));
yellowV6.add(new Vector3(118, 0.7, 317));
yellowV6.add(new Vector3(118, 0.7, 100));
YELLOWVEHICLESPATHS.push(yellowV6);

const blueV1 = new Path();
blueV1.add(new Vector3(108, 0.7, 3));
blueV1.add(new Vector3(116.5, 0.7, 3));
blueV1.add(new Vector3(118, 0.7, -195));
BLUEVEHICLESPATHS.push(blueV1);
const blueV2 = new Path();
blueV2.add(new Vector3(124, 0.7, -317.5));
blueV2.add(new Vector3(-45, 0.7, -317.5));
BLUEVEHICLESPATHS.push(blueV2);
const blueV3 = new Path();
blueV3.add(new Vector3(-112, 0.7, -307));
blueV3.add(new Vector3(-112, 0.7, -390));
BLUEVEHICLESPATHS.push(blueV3);
const blueV4 = new Path();
blueV4.add(new Vector3(-120, 0.7, 3));
blueV4.add(new Vector3(-112, 0.7, 3));
blueV4.add(new Vector3(-112, 0.7, -50));
BLUEVEHICLESPATHS.push(blueV4);
const blueV5 = new Path();
blueV5.add(new Vector3(-112, 0.7, 322));
blueV5.add(new Vector3(-112, 0.7, 200));
BLUEVEHICLESPATHS.push(blueV5);

const blueV6 = new Path();
blueV6.add(new Vector3(113, 0.7, 308));
blueV6.add(new Vector3(113, 0.3, 390));
BLUEVEHICLESPATHS.push(blueV6);

const busV1 = new Path();
busV1.add(new Vector3(113.5, 0.7, -323));
busV1.add(new Vector3(113.5, 0.7, -200));
BUSVEHICLESPATHS.push(busV1);
const busV2 = new Path();
busV2.add(new Vector3(-117, 0.7, -323));
busV2.add(new Vector3(-117, 0.7, -312));
busV2.add(new Vector3(-70, 0.7, -312));
BUSVEHICLESPATHS.push(busV2);
const busV3 = new Path();
busV3.add(new Vector3(-112, 0.7, 8));
busV3.add(new Vector3(-112, 0.7, -2));
busV3.add(new Vector3(-190, 0.7, -3));
BUSVEHICLESPATHS.push(busV3);
const busV4 = new Path();
busV4.add(new Vector3(-102, 0.7, 312));
busV4.add(new Vector3(-111, 0.7, 312));
busV4.add(new Vector3(-112, 0.7, 200));
BUSVEHICLESPATHS.push(busV4);
const busV5 = new Path();
busV5.add(new Vector3(118, 0.7, 323));
busV5.add(new Vector3(117, 0.7, 312));
busV5.add(new Vector3(0, 0.7, 312));
BUSVEHICLESPATHS.push(busV5);
const ANSWERSTEXT = [
  {
    question: 'Вопрос №1: В каком случае Вы имеете право проехать перекресток первым?', //(билет 11, вопрос 14)
    answer1: '1. Только при движении прямо',
    answer2: '2. При движении прямо и налево', //ответ
    answer3: '3. При движении прямо, налево и в обратном направлении',
    explanation: 'Правильный ответ 2. Перекрёсток равнозначный. Водители руководствуются "правилом правой руки", т.е. у кого помеха справа, тот и уступает. Помеха справа у водителя легкового автомобиля. Вы проезжаете перекрёсток первым при движении прямо и налево. При развороте у вас справа будет помеха. (Пункт 13.11 ПДД)',
  },
  {
    question: 'Вопрос №2: При повороте налево Вы:', //(билет 14, вопрос 15)
    answer1: '1. Имеете преимущество',
    answer2: '2. Должны уступить дорогу только автобусу', //ответ
    answer3: '3. Должны уступить дорогу легковому автомобилю и автобусу',
    explanation: 'Правильный ответ 2. Перекрёсток неравнозначный. Преимущество имеют транспортные средства, находящиеся на главной дороге. При повороте налево следует уступить дорогу транспортным средствам, движущимся прямо со встречного направления. Вы должны уступить только автобусу. (Пункты 13.9, 13.12 ПДД)',
  },
]

```

```

question: 'Вопрос №3: Кому Вы обязаны уступить дорогу при повороте налево?', //(билет 23, вопрос 15)
answer1: '1. Обоим транспортным средствам',
answer2: '2. Только автобусу', //ответ
answer3: '3. Только легковому автомобилю',
explanation: 'Правильный ответ 2. Перекрёсток неравнозначный. Главная дорога меняет направление. Транспортные средства,
находящиеся на главной дороге, имеют преимущество, между собой руководствуются "правилом правой руки". У Вас помеха справа -
уступаете автобусу. (Пункты 13.9, 13.10, 13.11 ПДД',
},
{
question: 'Вопрос №4: Кому Вы обязаны уступить дорогу при движении в прямом направлении?', //(билет 29, вопрос 15)
answer1: '1. Только легковому автомобилю',
answer2: '2. Только автобусу',
answer3: '3. Обоим транспортным средствам', //ответ
explanation: 'Правильный ответ 3. Перекрёсток неравнозначный. Главная дорога меняет направление. Преимущество имеют
транспортные средства, находящиеся на главной дороге. Между собой они руководствуются "правилом правой руки", т.е. те у кого
помеха справа, тот и уступает. Вы находитесь на второстепенной дороге - уступаете обоим транспортным средствам. ("Дорожные знаки"
2.4, 8.13, пункты 13.9, 13.10 ПДД)',
},
{
question: 'Вопрос №5: Кому Вы обязаны уступить дорогу при повороте налево?', //(билет 1, вопрос 15)
answer1: '1. Только автобусу',
answer2: '2. Только легковому автомобилю',
answer3: '3. Никому', //ответ
explanation: 'Правильный ответ 3. Главная дорога меняет направление. Транспортные средства, находящиеся на главной дороге,
имеют преимущество, а водители между собой руководствуются "правилом правой руки". Никому не уступая, первым проезжаете Вы,
вторым автобус, легковой автомобиль последним, так как он находится на второстепенной дороге. ("Дорожные знаки", пункты 13.9,
13.10, 13.11 ПДД)',
},
{
question: 'Вопрос №6: Кому Вы обязаны уступить дорогу при повороте налево?', //(билет 9, вопрос 15)
answer1: '1. Только автобусу',
answer2: '2. Только легковому автомобилю',
answer3: '3. Никому', //ответ
explanation: 'Правильный ответ 3. Главная дорога меняет направление. Транспортные средства, находящиеся на главной дороге,
имеют преимущество. А водители между собой руководствуются "правилом правой руки". У Вас помехи справа нет. Проезжаете первым,
водитель легкового автомобиля после Вас, автобус последним, так как находится на второстепенной дороге. (Пункты 13.9, 13.10, 13.11
ПДД)',
}
]
export {
YELLOWVEHICLESPATHS,
BLUEVEHICLESPATHS,
BUSVEHICLESPATHS,
ANSWERSTEXT
}

```