Содержание

[Введение 2](#_Toc194674030)

[1 Общая часть 4](#_Toc194674031)

[1.1 Постановка задачи 4](#_Toc194674032)

[1.2 Требования к программе 5](#_Toc194674033)

[1.3 Состав и параметры технических средств 6](#_Toc194674034)

[1.4 Выбор обоснование языка программирования 7](#_Toc194674035)

[2 Специальная часть 11](#_Toc194674036)

[2.1 Описание алгоритма 11](#_Toc194674037)

[2.2 Описание программы 18](#_Toc194674038)

[2.3 Тестирование и отладка 29](#_Toc194674039)

[3 Методический раздел 32](#_Toc194674040)

[Заключение 36](#_Toc194674041)

[Информационные источники 37](#_Toc194674042)

[Приложение А 38](#_Toc194674043)

# Введение

Тема курсового проекта – разработка обучающей программы Электронный учебник «Разработка игр на Unreal Engine 5 с использованием Blueprint».

В современном мире разработки игр платформа Unreal Engine 5 (UE5) приобретает все большее значение. UE5 предоставляет широчайшие возможности для создания высококачественных, визуально впечатляющих и интерактивных игровых проектов. Благодаря своей гибкости, мощному инструментарию и ориентированности на современные технологии, UE5 является идеальным выбором для разработки игр различных жанров и масштабов.

Одним из ключевых аспектов UE5 является использование системы визуального скриптинга Blueprint. Blueprint представляет собой графический интерфейс программирования, который позволяет разработчикам, даже без глубоких знаний традиционных языков программирования, создавать игровую логику, взаимодействовать с элементами сцены и управлять поведением персонажей.

Применение UE5 и Blueprint в разработке игр приводит к многочисленным преимуществам, включая ускорение процесса разработки, снижение порога вхождения для начинающих разработчиков и возможность быстрого прототипирования игровых механик. Таким образом, изучение и использование UE5 и Blueprint становится критически важным для разработчиков игр, стремящихся создавать инновационные и увлекательные игровые проекты.

Конечно, обучающая программа не может заменить человека-преподавателя, но она может дополнить и облегчить деятельность учителя. В некоторых областях, электронные учебник помогают ученику развить самостоятельность и творческое мышление, что поможет ему стать более професиональным специалистом.

Таким образом, проблема создания обучающих программ или курсов является актуальной.

Электронный учебник – это современная образовательная среда, предоставляющая структурированный курс знаний, базирующийся на передовых научных достижениях. Помимо теоретического материала, он содержит практические задания и тесты для самоконтроля. Ключевым элементом является стимулирование самостоятельного поиска информации и ее интеграции в практические проекты, что позволяет закрепить полученные знания и развить навыки.

Курсовой проект включает в себя программу «Электронный учебник «Разработка игр на Unreal Engine 5 с помощью blueprint» на флеш-носителе и документацию к ней в виде пояснительной записки и графической части.

Пояснительная записка курсового проекта содержит три раздела: «Общая часть», «Специальная часть», «Методическая часть».

Раздел «Общая часть» включает: постановку задачи, требования к программе, состав и параметры технических средств, а также выбор и описание языка программирования, на котором реализована данная программа.

Раздел «Специальная часть» включает описание алгоритма задачи, описание программы, тестирование и отладку.

Методический раздел включает краткую содержательную часть теоретической информации, лабораторные работы, тестирование с перечнем вопросов и вариантами ответов с правильным решением.

Графическая часть представлена в виде двух схем:

* схемы взаимодействия программ: ВЛГК.090207.КП‑106 С1 отображает путь активации программы и взаимодействие с соответствующими данными.

# Общая часть

## 1.1 Постановка задачи

Тема курсового проекта – Электронный учебник «Разработка игр на Unreal Engine 5 с помощью blueprint». Заказчиком является государственное автономное профессиональное образовательное учреждение Владимирской области "Высшая государственная инженерно-техническая школа (колледж) им. Д.К. Советкина отделение информационных технологий (ИТ).

Целями разработки являются:

* повышение результативности обучения;
* уменьшение временных затрат на обучение;
* уменьшение временных затрат на поиск информации на тему «Разработка игр на Unreal Engine 5 с помощью blueprint»
* возможность самостоятельного изучения темы «Разработка игр»;
* возможность подготовки к выполнению лабораторных работ по теме «Разработка игр на Unreal Engine 5 с помощью blueprint»;
* наглядная форма обучения.

Исходные данные для разработки:

* теоретический раздел – включает в себя материал «Разработка игр на Unreal Engine 5 с помощью blueprint» [1];
* практический раздел – включает методический материал ( [3], [5], [7]);
* тестирование основывается на теоретическом материале, составляется

самостоятельно и согласовывается с преподавателем.

Выходными данными является материал, изложенный в теоретической и практической частях с возможностью их сохранения и вывода на печать, а также результат тестирования, выраженный в процентах.

При изучении темы курсового проекта «Разработка игр на Unreal Engine 5 с помощью blueprint» был выполнен мониторинг существующих аналогов программы. В открытом доступе аналогов программы не найдено.

В задании курсового проекта Электронный учебник «Разработка игр на Unreal Engine 5 с помощью blueprint» входит разработка небольшой части «Подсистемы управления ресурсами. Управление процессами и потоками. Сетевое программирование и обмен данными». Программа должна быть построена по модульно – иерархическому принципу и включать части:

* теоретическая;
* практическая;
* тестирование;
* справка.

## 1.2 Требования к программе

Электронный учебник « Разработка игр на Unreal Engine 5 с помощью blueprint» должен корректно запускаться и стабильно функционировать как с флеш-накопителя, на котором он будет поставляться, так и из любого другого места жесткого диска, куда будет скопирована программа.

Программа должна обладать удобным интерфейсом, позволяющим гибко переключаться между разделами.

Теоретический материал по теме Электронный учебник «Разработка игр на Unreal Engine 5 с помощью blueprint» должен быть выполнен в виде файлов формата txt, помещенных в папку theory, содержать лаконичную и понятную информацию, разбитую по темам.

Практический материал по теме «Разработка игр на Unreal Engine 5 с помощью blueprint» должен быть выполнен в виде файлов формата txt, помещенных в папку practicate, содержать лаконичную и понятную информацию, разбитую по темам.

Электронный учебник «Разработка игр на Unreal Engine 5 с помощью blueprint» должен содержать раздел тестирования, ориентированного на самопроверку изученной темы, при этом предусмотреть вывод результата о количестве правильных ответов и оценку в процентном соотношении.

Все формы программы должны быть:

* выполнены в едином цветовом решении;
* выполнены в едином стиле;
* отцентрированы.

В Электронный учебник «Разработка игр на Unreal Engine 5 с помощью blueprint» должен быть реализован диалоговый интерфейс пользователя: в разделе тестирования с выдачей информации и предупреждающих сообщений.

Электронный учебник « Разработка игр на Unreal Engine 5 с помощью blueprint» должен быть записан в папку Program на флеш- накопителе. Документация должна быть помещена на этот оптический источник в папку Documentation, куда помещаются: пояснительная записка, графическая часть, титульный лист.

## Состав и параметры технических средств

Программное средство «Электронный учебник «Разработка игр на Unreal Engine 5 с помощью blueprint» предназначен для работы в операционной системе (ОС) Windows 10. Компьютер с набором минимальных технических средств представлен в таблице 1.1.

Таблица 1.1 – Минимальный состав технических средств

| Наименование | Конфигурация |
| --- | --- |
| Процессор | Intel Pentium |
| Оперативная память | 10 ГБ и выше |
| Видеокарта | PCI-E 512Mb GeForce Gt210  GigaByte GV-N210OC-512I |
| Монитор | Монитор Samsung SyncMaster  943NW, 1440x900, 75 Гц, TN |
| Клавиатура | Gembird KB-8320U-Ru\_Lat-BL  Black USB |
| Манипулятор «мышь» | Defender Dot MB-986 |
| Принтер | HP LaserJet P1005, ч/б, А4 |

В таблице 1.2 представлены программные средства для для работы и разработки курсового проекта Электронный учебник «Разработка игр на Unreal Engine 5 с помощью blueprint»

Таблица 1.2 – Программные требования

|  |  |  |  |
| --- | --- | --- | --- |
| Группа требований | Фактический стандарт | Примечания | Объем памяти на жестком диске |
| Операционная система | Microsoft Windows 10 | Для работы | 20 Гб |
| Среда разработки | IntelliJ IDEA Community Edition | Для разработки 1 | 10 Гб |
| Графический редактор | GIMP | Для разработки | 1 Гб |
| Текстовый редактор | Microsoft Office Word | Для оформления докуметации | 4 Гб |
| Графический проектировщик | Microsoft Office Visio | Для разработки графической части | 4 Гб |

Исходя из таблицы 1.2, системные и программные средства требуют 39 Гб памяти на жестком диске.

## Выбор обоснование языка программирования

Для программирования задачи был выбрал язык Python.

Причинами выбора данного языка программирования послужили: большая область применения; Легкость в освоении. Поддержка обработки http запросов; полная интеграция с вёрсткой страниц (HTML+CSS).

Python – это простой язык программирования. Язык легок в освоении его синтаксис понятен и прост в запоминании. Имеет поддержку большого количества библиотек, что раскрывает его возможности на максимум.

Основное преимущество Python – универсальность, этот язык применяется практически во всех основных областях, включая веб-разработку, науку о данных, автоматизацию, Интернет вещей и многое другое.

Общие характеристики: Python – высокоуровневый язык программирования, имеющий динамическую типизацию данных и автоматическое управление памятью, что помогает программисту быстрее разрабатывать код. Язык является полностью объектно-ориентированным что значит, что все является объектами. Для повышения читабельности кода, Python имеет уникальную необходимость выделять блоки кода отступами.

Python – мультипарадигменный язык программирования, что означает он поддерживает императивное, процедурное, структурное, функциональное, асинхронное, объектно-ориентированное программирование и метапрограммирование.

Области применения Python: веб-разработка, анализ данных, машинное обучение, анимация и графика, мобильная разработка. В области веб-разработки Python выступает в качестве серверной обработки данных, которые не видны пользователю. Например для сохранения паролей, проверки доступа к странице, отправки HTML разметки страницы пользователю.

Основные элементы HTML разметки:

* <!--...--> - используется для добавления комментариев;
* <body> - представляет содержимое HTML-документа. В документе может быть только один такой элемент;
* <!DOCTYPE> - объявляет тип документа и предоставляет основную информацию для браузера — его язык и версия;
* <a> - создаёт гипертекстовые ссылки;
* <ul> - устанавливает список;
* <script> - для описания скриптов и может содержать ссылку на программу на определенном языке;
* <li> - отдельный элемент списка;
* <ins> - выделяет текст подчеркиванием. Применяется для выделения изменений, вносимых в документ;
* <p> -  представляет абзац. Абзацы обычно представляются в визуальных носителях в виде блоков текста, отделенных от соседних блоков пустыми линиями и/или отступами первой строки, но HTML-абзацы могут представлять собой любую структурную группировку связанного содержимого, такого как изображения или поля формы;
* <style> - подключает встраиваемые таблицы стилей;
* <title> - заголовок HTML-документа, отображаемый в верхней части строки заголовка браузера. Также может отображаться в результатах поиска, поэтому это следует принимать во внимание предоставление названия;
* <button> - определяет нажатую кнопку;
* <head> - определяет сведения о документе;
* <base> - содержит машиночитаемую информацию о документе, такую как его заголовок, сценарии и таблицы стилей;
* <link> - Задает связи между текущим документом и внешним ресурсом. Этот элемент чаще всего используется для ссылки на CSS, но также используется для создания значков сайта;
* <meta> - представляет метаданные, которые не могут быть представлены другими элементами, связанными с метаданными HTML, такими как <base>, <link>, <script>, <style> и <title>;
* <section> - представляет универсальный автономный раздел документа, который не имеет более конкретного семантического элемента для его представления. Разделы всегда должны иметь заголовок, за очень немногими исключениями;
* <div> - универсальный контейнер для содержимого потока. Он не влияет на содержимое или макет до тех пор, пока не будет стилизован каким-либо образом с использованием CSS;
* <hr> - представляет тематический разрыв между элементами уровня абзаца: например, изменение сцены в материале или сдвиг темы в разделе.

# 2 Специальная часть

## Описание алгоритма

* + 1. Вызов и загрузка

Дистрибутив программы находится на флеш-накопителе. Программа находится в папке ..\Program\. Чтобы запустить программу в демонстрационном режиме, необходимо использовать флеш-накопитель. Для полноценной работы программы ее необходимо скопировать в любое место жесткого диска и запустить файлом main.exe и в браузере в адресной строке ввести адрес http://localhost:5000.

* + 1. Интерфейс пользователя

Интерфейс пользователя должен быть выполнен в единых цветовых решениях на всех страницах. Интерфейс программы построен на основе главного меню, расположенного в шапке каждой страницы.

Структура меню программы, изображенная на рисунке 2.1, содержит разделы  Главная,  Теория,  Практика,  Тестирование, смена темы страницы.



Рисунок 2.1 – Структура меню программы

* + 1. Входные и выходные данные

Входные данные подготавливаются в процессе разработки структуры интерфейса программы и представляют собой .html, .css, .js, .py, .exe файлы, описанные в таблице 2.1.

Таблица 2.1 – Структура файлов программы

| Наименование | Описание | Место расположения |
| --- | --- | --- |
| main.exe | Упакованный главный скрипт | Program/ |
| main.py | Главный скрипт обработки данных | Program/ |
| main.html | Главная страница | Program/templates |
| admin\_panel.html | Страница администрирования | Program/templates |
| 404.html | Страница ошибки 404 | Program/templates |
| theory.html | Страница с лекциями | Program/templates |
| practicate.html | Страница с практической работами | Program/templates |
| tests.html | Страница с тестами для самопроверки | Program/templates |
| login.html | Страница авторизации | Program/templates |
| MainStyle.css | Файл каскадных таблиц стилей для главной страницы | Program/static/css |
| AdminPanelStyle.css | Файл каскадных таблиц стилей для страницы администрирования | Program/static/css |
| 404.css | Файл каскадных таблиц стилей для 404 страницы | Program/static/css |
| TheoryStyle.css | Файл каскадных таблиц стилей для страницы с лекциями | Program/static/css |
| PracticateStyle.css | Файл каскадных таблиц стилей для страницы с практическими работами | Program/static/css |
| TestsStyle.css | Файл каскадных таблиц стилей для страницы с тестами для самопроверки | Program/static/css |
| LoginStyle.css | Файл каскадных таблиц стилей для страницы авторизации | Program/static/css |
| Theme-change.js | Скрипт для смены и сохранения выбранной темы страницы у пользователя |  |

Структура файлов и описание теории представлены в таблице 2.2.

Таблица 2.2 – Структура файлов теории

| Наименование | Содержание файла | Описание файла |
| --- | --- | --- |
| theory1.txt | Лекция №1 | Введение в инструмент Unreal Engine 5 |
| theory2.txt | Лекция №2 | Обзор редактора инструмента Unreal Engine 5 |
| theory3.txt | Лекция №3 | Использование и наследования технологии Blueprint’s |
| theory4.txt | Лекция №4 | Основы работы с игровыми мирами в инструменте Unreal Engine 5 |
| theory5.txt | Лекция №5 | Использование графики и освещения в инструменте Unreal Engine 5 |

Структура и описание файлов практики представлены в таблице 2.3.

Таблица 2.3 – Структура файлов практики

| Наименование | Содержание файла | Описание файла |
| --- | --- | --- |
| practicate1.txt | Практическая работа №1 | Создание простой локации или игровой зоны в Unreal Engine 5 |
| practicate2.txt | Практическая работа №2 | Создание базового игрового персонажа с использованием возможностей Unreal Engine 5 |
| practicate3.txt | Практическая работа №3 | Создание мира с игровой логикой и функциональностью используя Blueprint’s |

Структура и описание файлов практики представлены в таблице 2.3.

Таблица 2.4 – Структура файлов практики

| Наименование | Содержание файла | Описание файла |
| --- | --- | --- |
| test1.txt | Тестовая работа №1 | Тестовая работа по лекции 1 |
| test2.txt | Тестовая работа №2 | Тестовая работа по лекции 2 |
| test3.txt | Тестовая работа №3 | Тестовая работа по лекции 3 |

* + 1. Логическая структура

Логическая структура приведена на рисунке 2.2

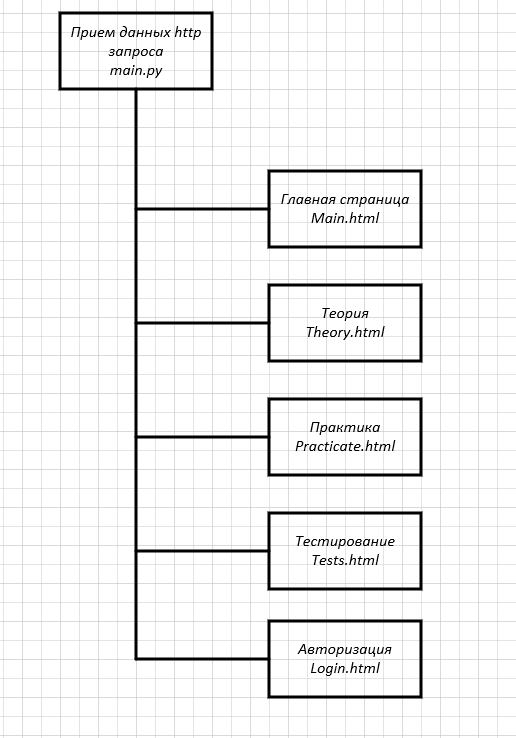


Рисунок 2.2 – Логическая структура программы

В таблице 2.4 приводится наименование модуля, его функциональное назначение, откуда он вызывается и что вызывает.

Таблица 2.5 – Модули программы

| Наименование модуля | Функциональное назначение |
| --- | --- |
| index.html | Главная страница |
| theory1.html | Страница с лекцией №1 |
| theory2.html | Страница с лекцией №2 |
| theory3.html | Страница с лекцией №3 |
| theory4.html | Страница с лекцией №4 |
| theory5.html | Страница с лекцией №5 |
| theory6.html | Страница с лекцией №6 |
| theory7.html | Страница с лекцией №7 |
| theory8.html | Страница с лекцией №8 |
| theory9.html | Страница с лекцией №9 |
| theory10.html | Страница с лекцией №10 |
| practice1.html | Страница с практической работой №1 |
| practice2.html | Страница с практической работой №2 |
| practice3.html | Страница с практической работой №3 |
| testing1.html | Страница тестирования (первый тест) |
| testing2.html | Страница тестирования (второй тест) |
| testing3.html | Страница тестирования (третий тест) |
| reference.html | Страница справки |
| styles.css | Файл каскадных таблиц стилей |
| test1.js | Файл на языке программирования JavaScript |
| test2.js | Файл на языке программирования JavaScript |
| test2.js | Файл на языке программирования JavaScript |

## 2.2 Описание программы

Для корректной работы программы её рекомендуется предварительно скопировать в любое место жесткого диска. Программа запускается файлом main.exe. После запуска программы открывается лог-консоль сервера, который запускается на порте 5000, чтобы открыть сайт с компьютера на котором запустился сервер нужно в адресной строке браузера ввести <http://127.0.0.1:5000>. Чтобы запустить с другого компьютера в локальной сети требуется ввести локальный адрес компьютера-сервера. После перехода на сайт откроется Главная страница с которой можно перейти на страницу Тестов, Лекций, Лабораторных работ или войти в аккаунт, Рисунок 2.1.

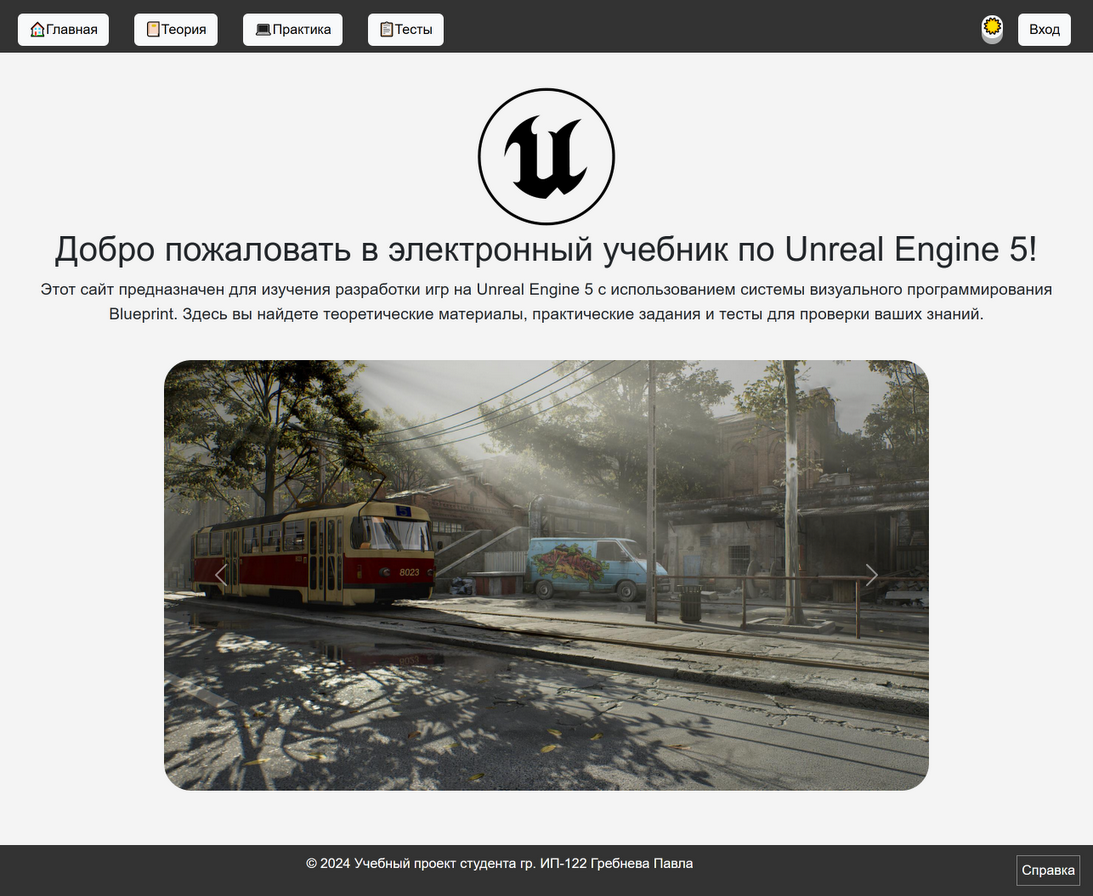


Рисунок 2.1 – Главная страница

Перейдя на страницу Теория можно увидеть раскрывающийся раздел с лекционным материалом, изображено на рисунке 2.2.

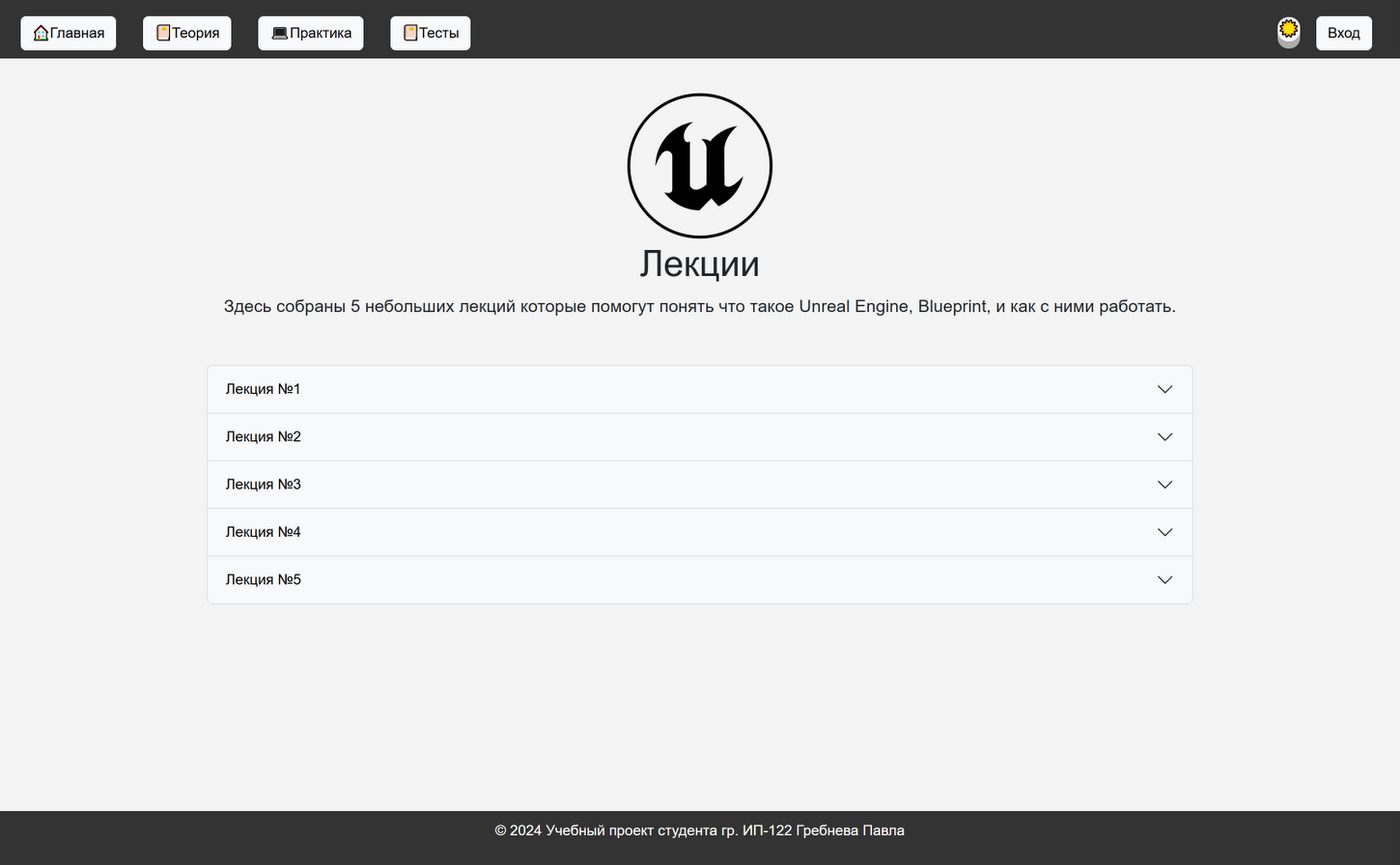


Рисунок 2.2 – Теория

После выбор нужно лекционного материала раскроется текст этого лекционного материала, при повторном нажатии или открытии другой лекции, текущая закроется, пример изображен на рисунке 2.3.

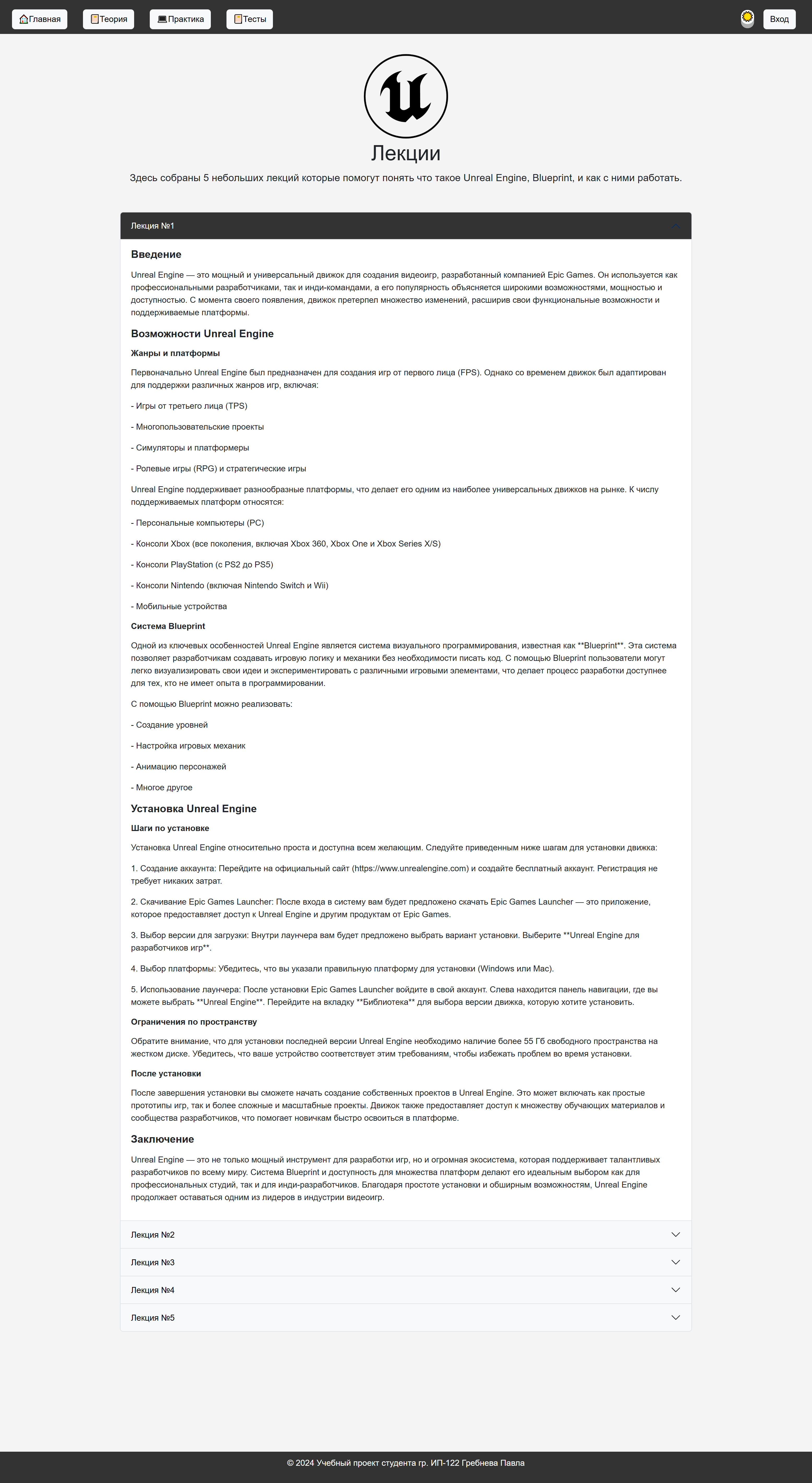


Рисунок 2.3 – Пример страницы теоретического материала

Перейдя на страницу Практика можно увидеть раскрывающийся раздел с практическими работами. Все действия для прочтения соответствуют с действиями которые проводились на странице с лекционным материалом (Рисунок 2.4).

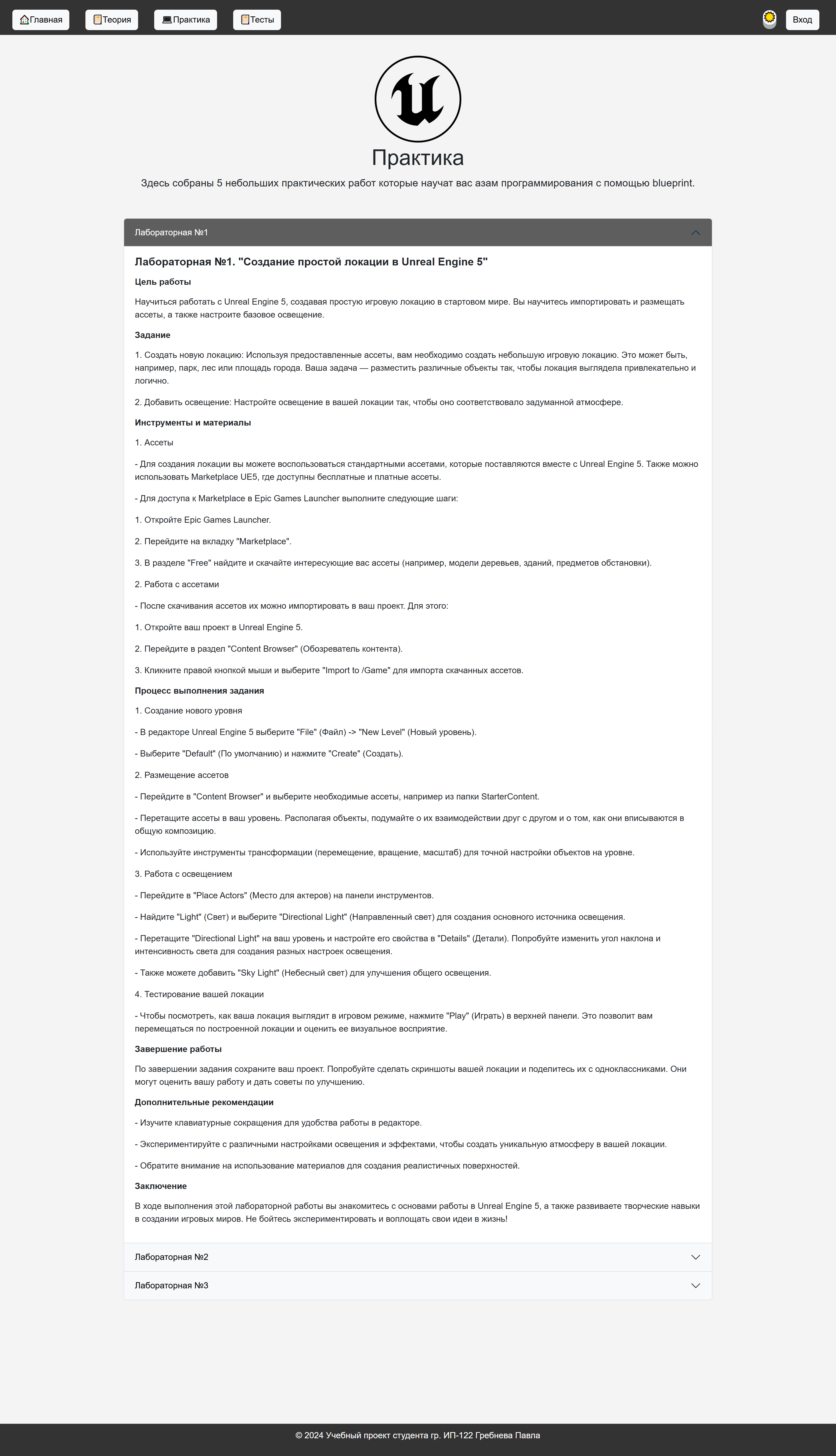


Рисунок 2.4 – Страница «Практика» с практической работой №1.

На странице тестирования есть три кнопки, которые открывают доступные тесты (рисунок 2.5). Если тест в данный момент не доступен появляется предупреждение в котором говорится что данный тест сейчас не доступен (рисунок 2.6).

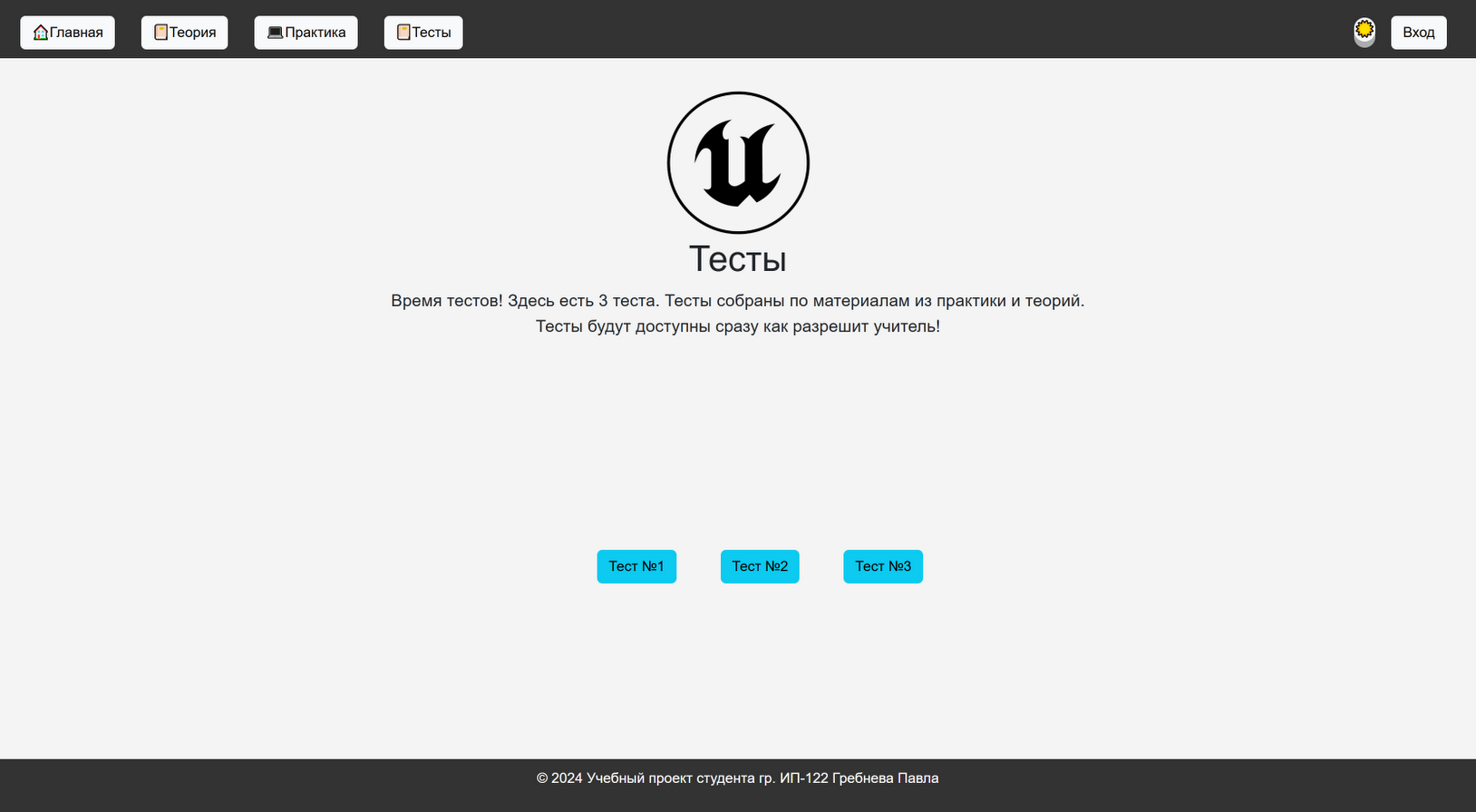


Рисунок 2.5 – Страница Тесты

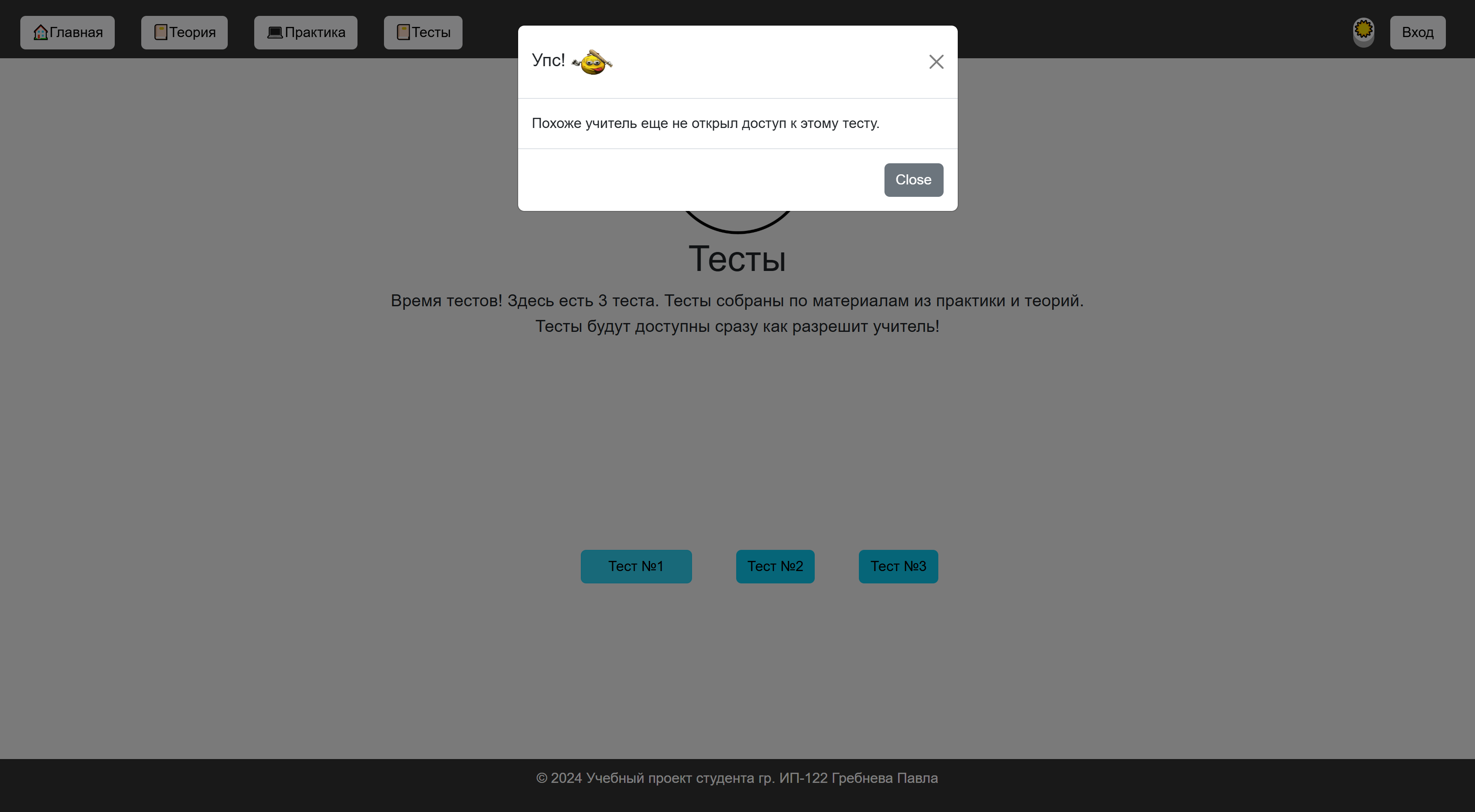


Рисунок 2.6 – Предупреждение о недоступности теста

Если нужный тест доступен, ниже кнопок открывается окно тестирования, в котором можно выбирать с помощью радио-кнопок варианты ответа (рисунок 2.7).

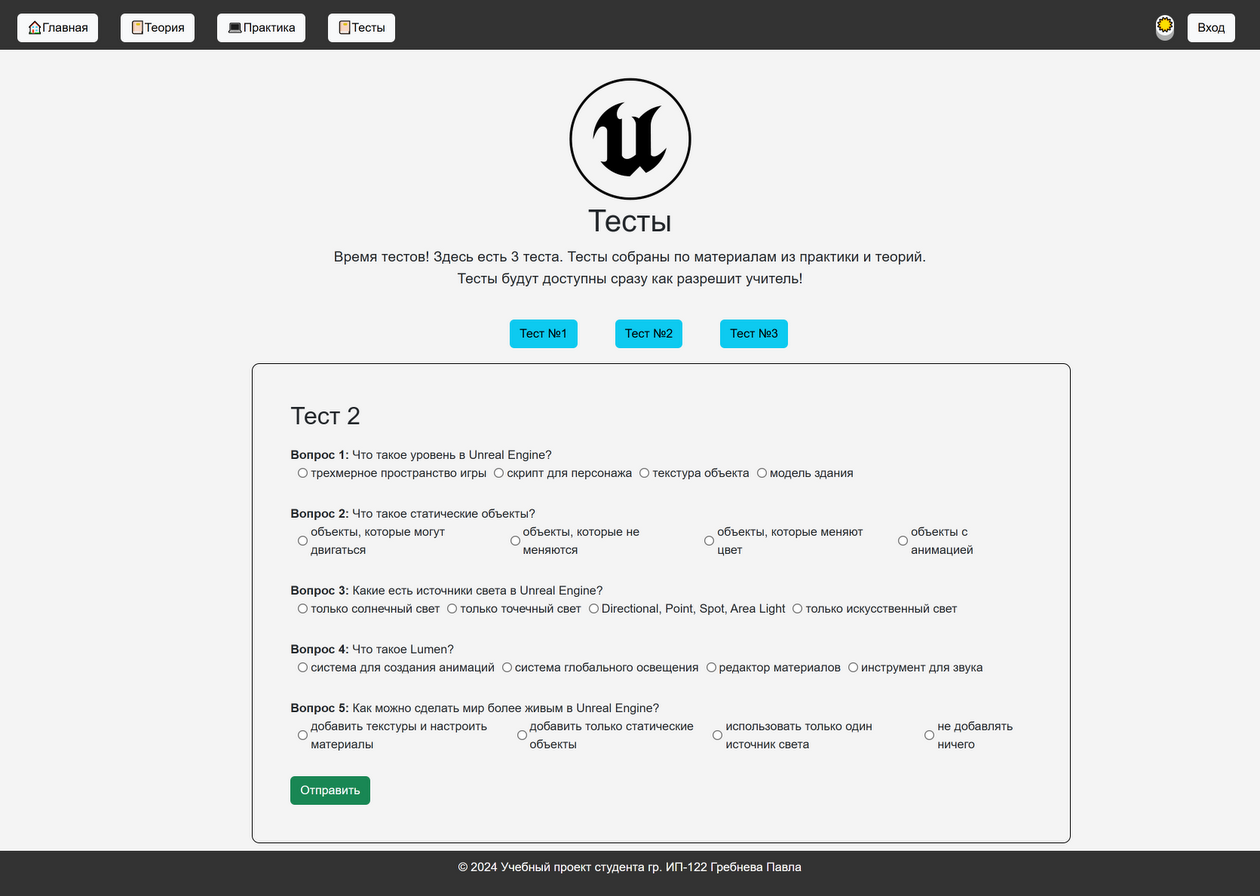


Рисунок 2.7 – Страница Тесты с окном тестирования

После нажатия на кнопку отправке, появляется окно с результатами тестирования (рисунок 2.8).

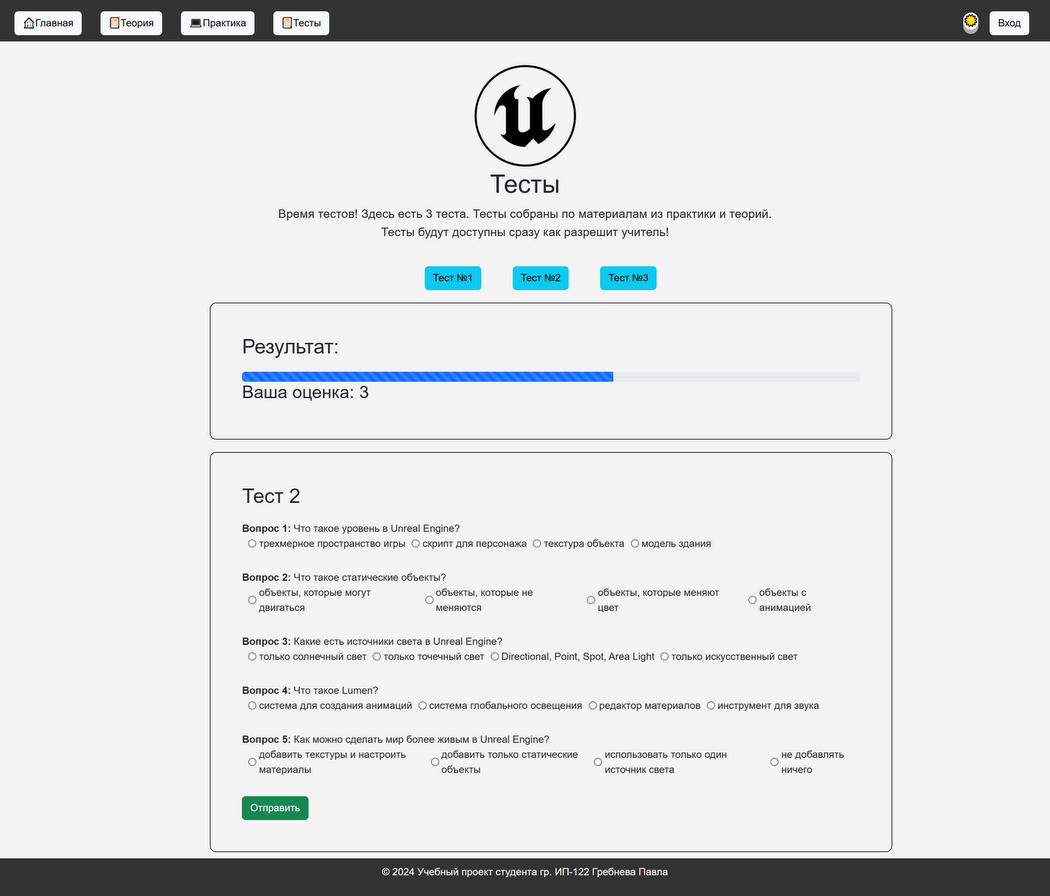


Рисунок 2.8 – Окно с результатами тестирования

В панели управления сверху сайта также имеется кнопка переключения темы, для удобства просмотра, при нажатии на которую сайт становится более темным и в ночное время глаза не будет слепить. Пример главной страницы с темной темой изображен на рисунок 2.9.



Рисунок 2.9 – Темная тема на главной странице

На сайте присутствует страница авторизации, в ней может зайти как ученик так и учитель.

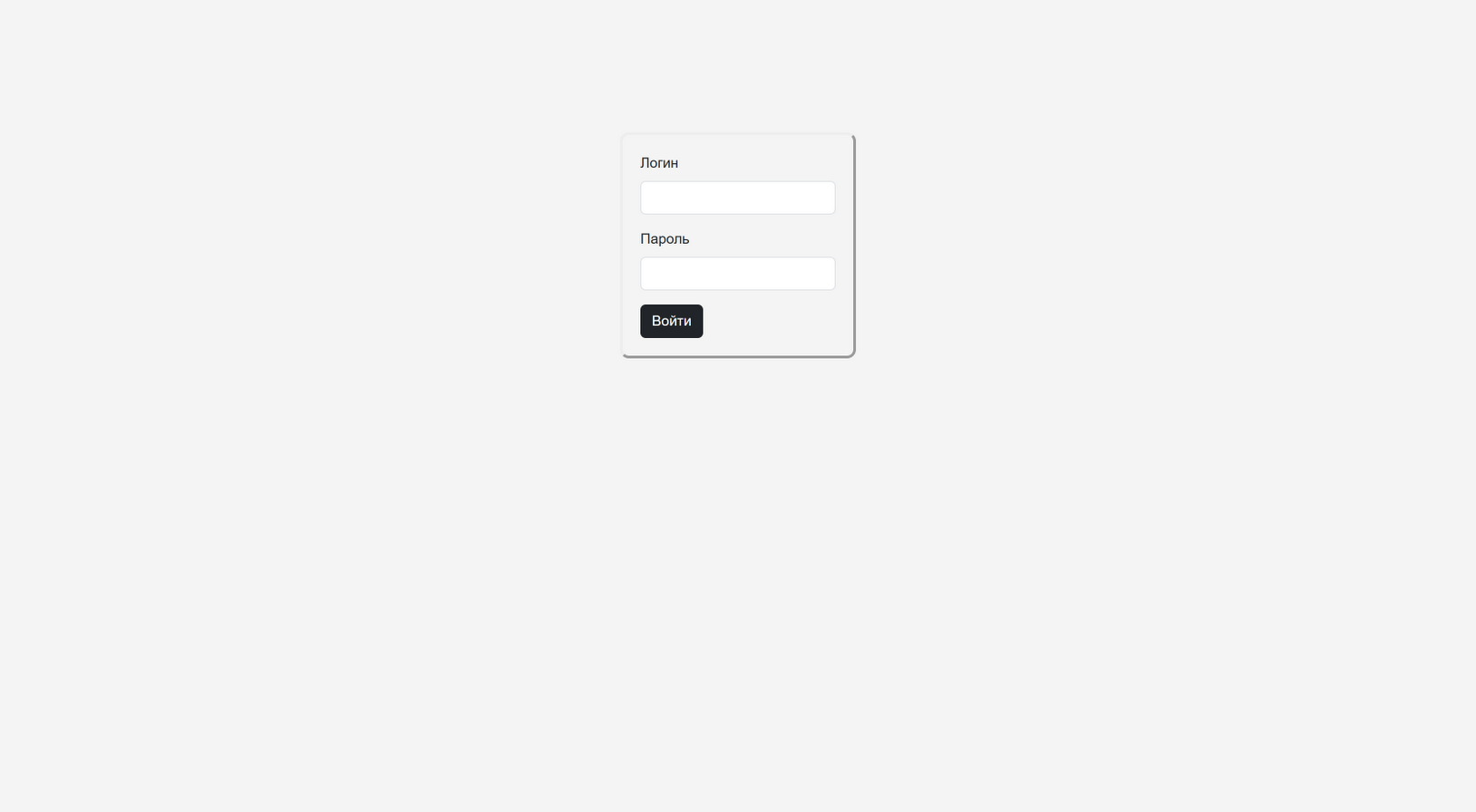


Рисунок 2.10 – Страница авторизации

Войдя как учитель можно заметить что у учителя есть доступ к администрационой панели (рисунок 2.12).

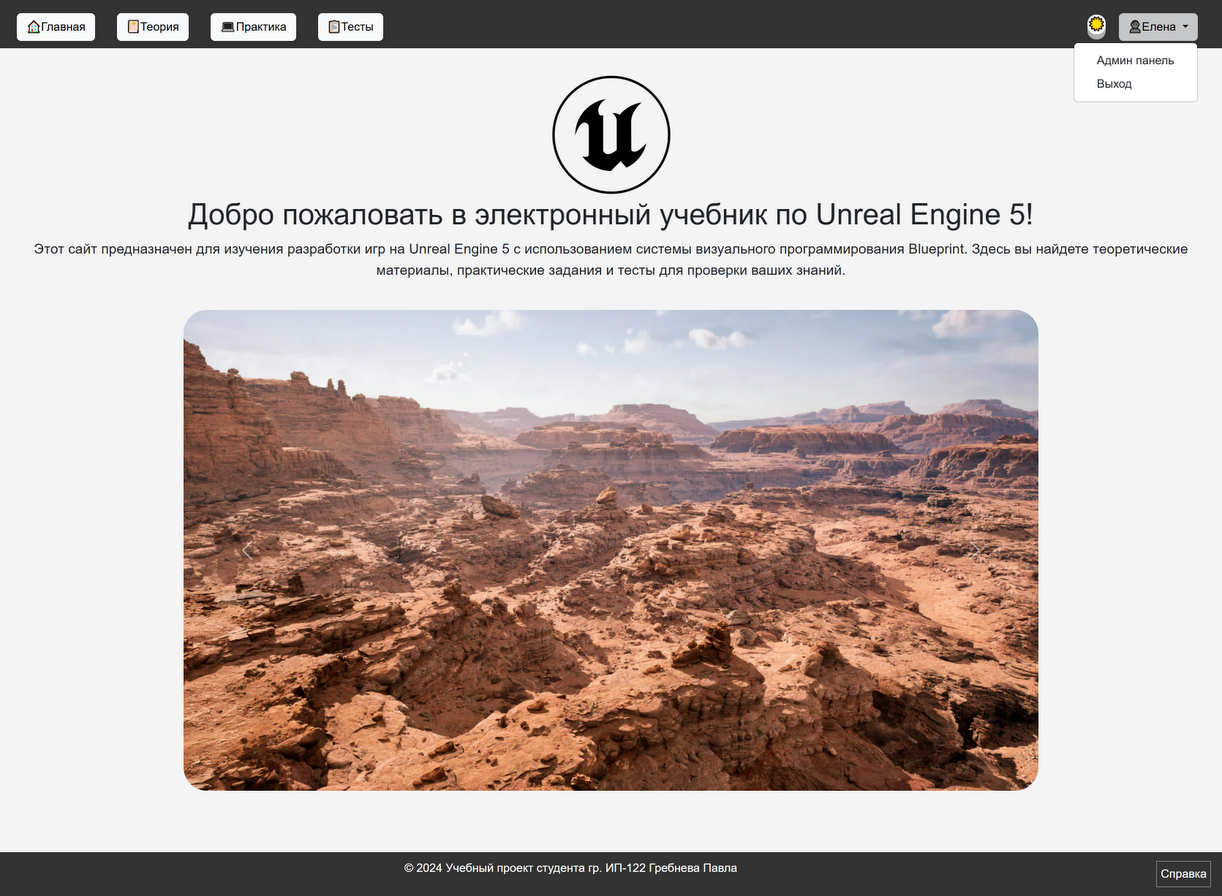


Рисунок 2.11 – Доступ учителя к админ-панели

В администрационой-панели можно добавлять и редактировать существующие аккаунты учеников и учителей. Возможно изменять тексты теорий, практических работ и тестов, в которых можно добавлять новые вопросы. (рисунки 2.13, 2.14, 2.15)

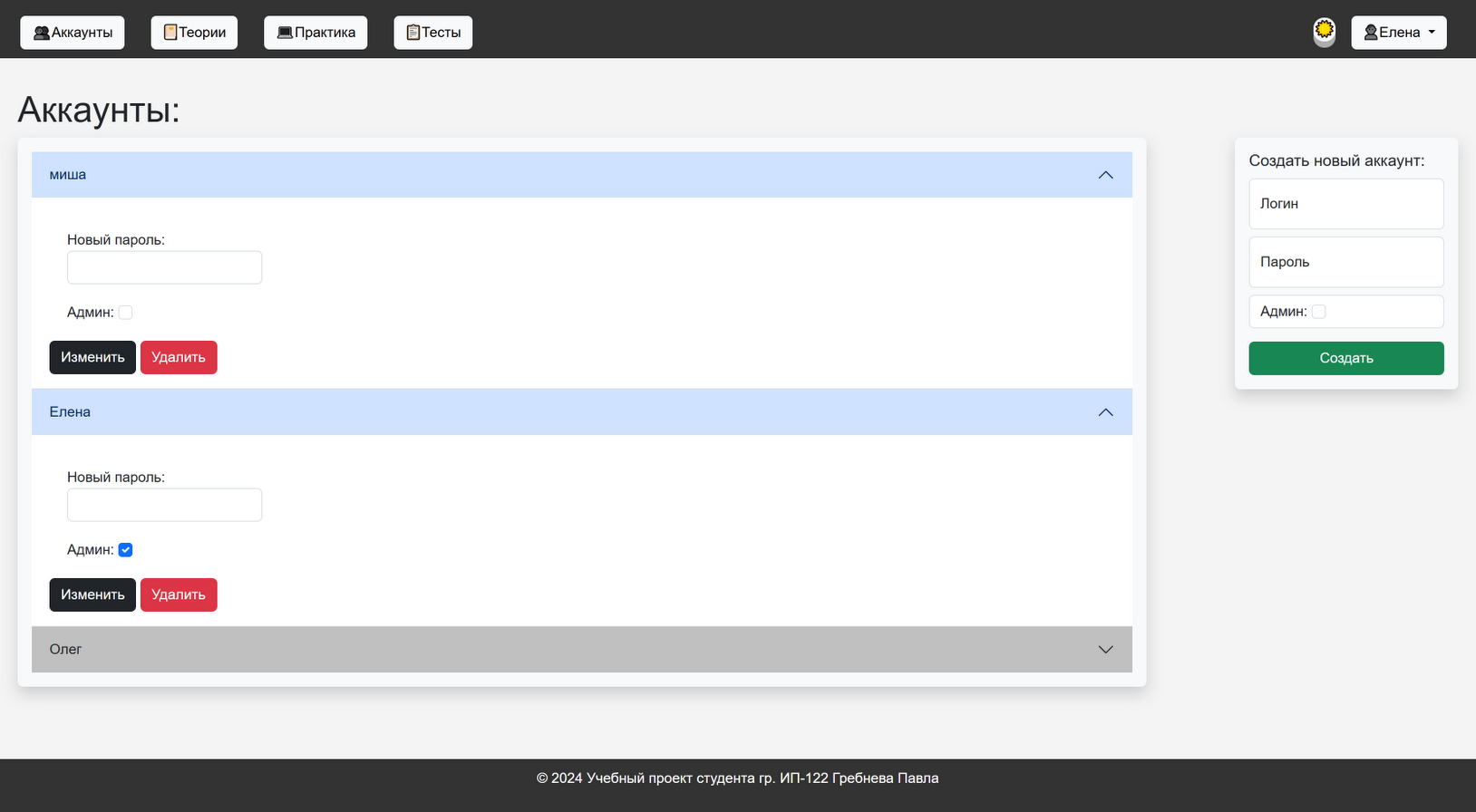


Рисунок 2.12 – Страница с редактированием аккаунтов

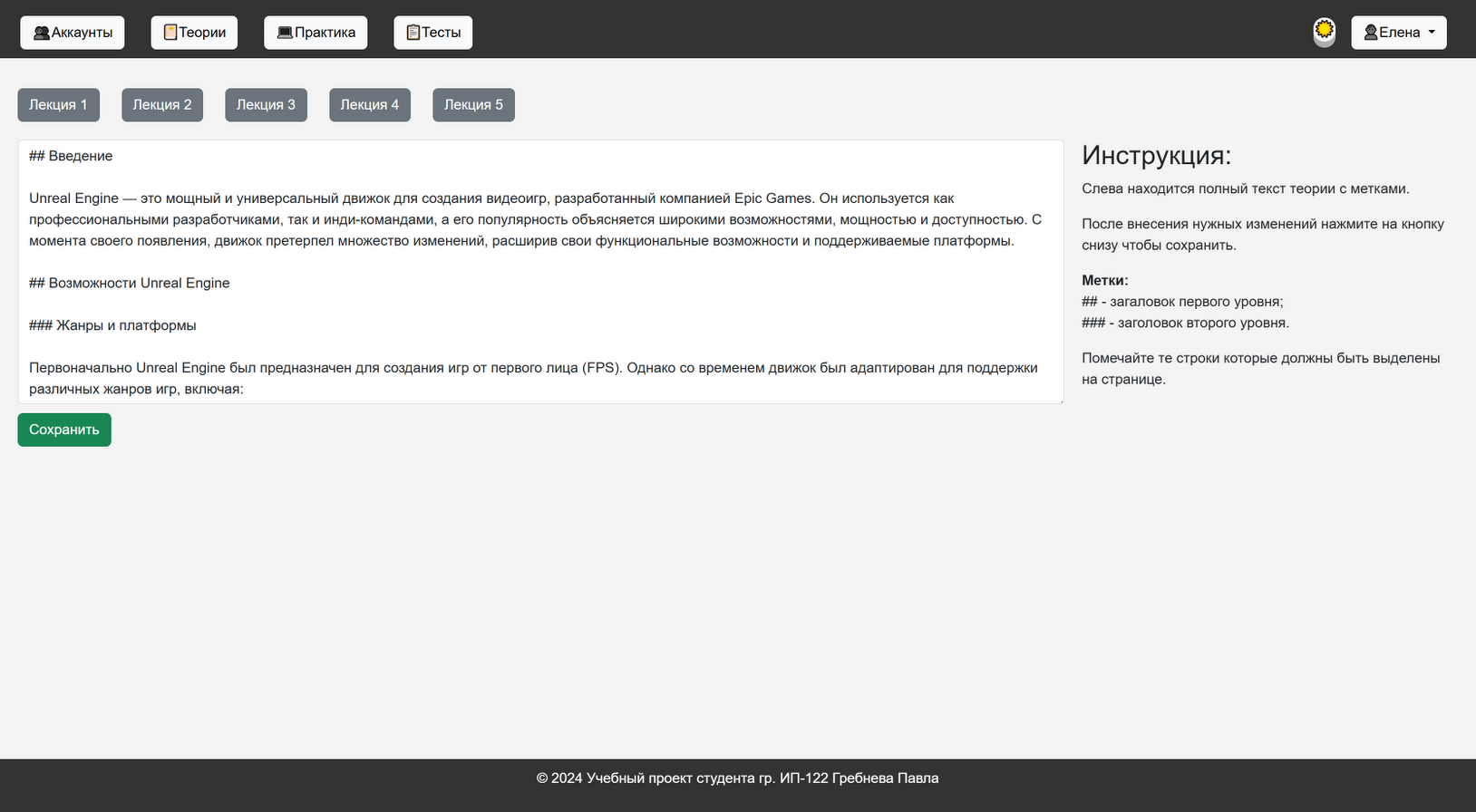


Рисунок 2.13 – Страница с редактированием текста лекций

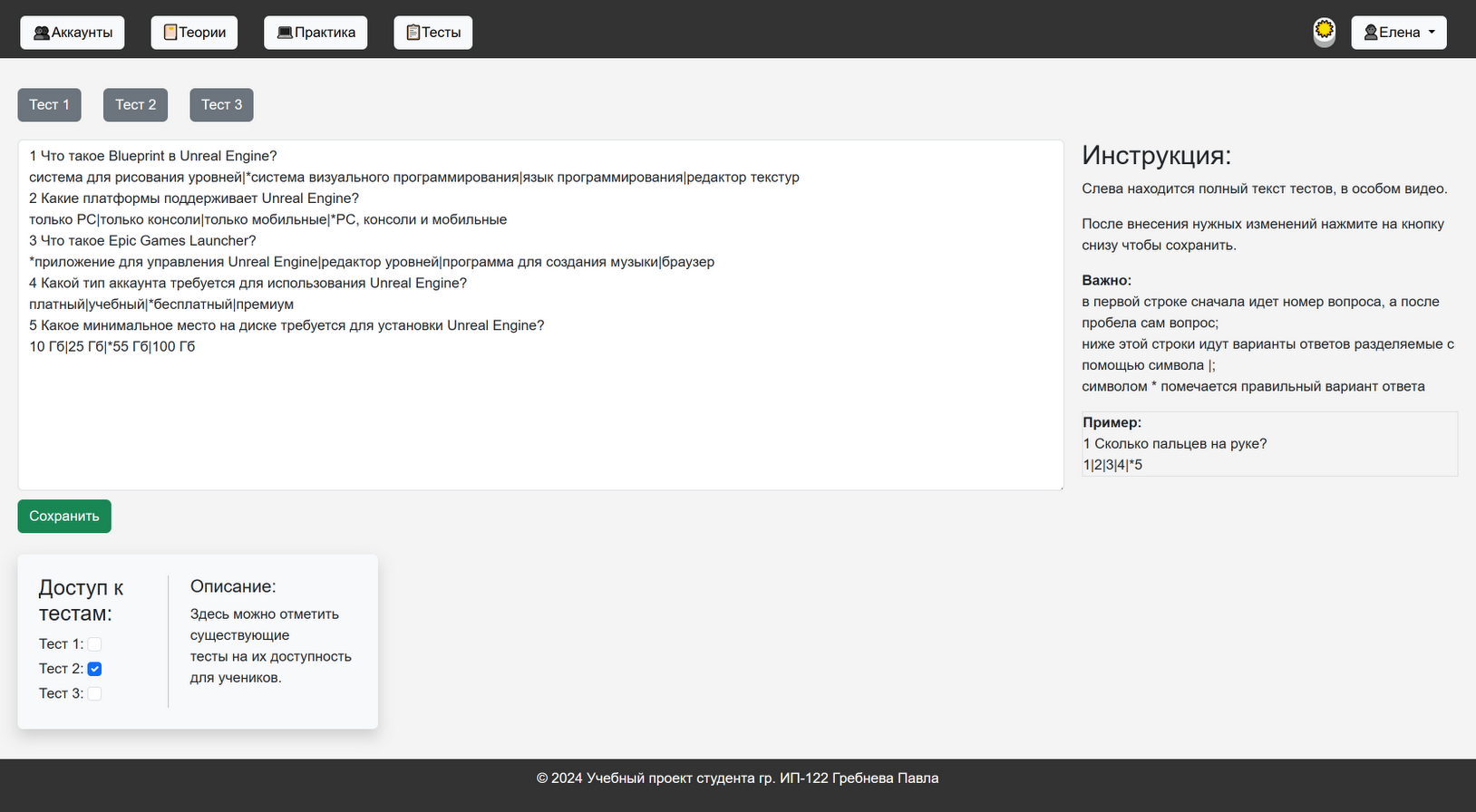


Рисунок 2.14 – Страница с редактированием тестов.

Справка находится в самом низу главной страницы. (рисунок 2.16)



Рисунок 2.15 – Главная страница с справкой

## 2.3 Тестирование и отладка

2.3.1 Выбор стратегии тестирования

Стратегия тестирования - система методов отбора и создания тестов для тестового набора.

Систематические методы тестирования делятся на методы, в которых программы рассматриваются как «черный ящик», и методы, в которых программа рассматривается как «белый ящик».

2.3.2 Метод тестирования «черного ящика»

Тестирование «черного ящика» — это метод тестирования функционального поведения объекта с точки зрения внешнего мира. При тестировании методом «черного ящика» программа рассматривается как объект, внутренняя структура которого неизвестна. Тестировщик вводит данные и анализирует результат, но он не знает, как именно работает программа.

Достоинства:

* простота: облегчает тестирование проектов высокого уровня и сложных приложений;
* экономия ресурсов: тестеры сосредоточены на функциональности программного обеспечения;
* тестовые случаи: Сосредоточение внимания на функциональности программного обеспечения для облегчения быстрой разработки тестовых случаев;
* обеспечивает гибкость: специальные знания программирования не требуются.

Недостатки:

* невозможно найти взаимоуничтожающиеся ошибки ;
* некоторые ошибки возникают достаточно редко (ошибки работы с памятью) и потому их трудно найти и воспроизвести.

Таблица 2.6 – Тестирование методом «черного ящика»

| Тест | Ожидаемый результат | Фактический результат |
| --- | --- | --- |
| Запуск программы | Открытие Главной страницы в браузере | Открытие Главной страницы в браузере |
| Прохождение тестирования на оценку 3 | Оценка 3 | Оценка 3 |

2.3.3 Метод тестирования «белого ящика»

Метод «белого ящика» позволяет исследовать внутреннюю структуру программы. Тестирование по принципу «белого ящика» ориентировано на проверку прохождения всех путей программного кода. Тестирование «белого ящика» применяется на уровне модулей путем выбора тестовых ситуаций и подготовки данных.

Достоинства:

* оптимизация кода путем нахождения скрытых ошибок;
* доступность структуры кода позволяет выбрать тип входных данных, необходимых для эффективного тестирования;
* возможность автоматизирования тест-кейсов.

Недостатки:

* поскольку знание кода и внутренней структуры является необходимым условием, для проведения такого тестирования требуется квалифицированный тестировщик, что увеличивает стоимость;
* почти невозможно изучить каждый кусок кода, чтобы обнаружить скрытые ошибки, что может создать проблемы, приводящие к сбою приложения.

В таблице 2.6 представлены ошибки, которые были допущены при разработке программы.

Таблица 2.7 – Ошибки, допущенные при разработке программы

| Ошибка | Причина ошибки | Меры устранения ошибки |
| --- | --- | --- |
| Не отображается главная страница | Разметка главной страницы не находится в папке templates | Изменение пути в коде программы |
| Все лекции имеют одинаковый текст | Указан неверный путь к файлу лекции theory1.txt Неверно построен алгоритм программы | Изменение пути к файлу theory1.txt в программном коде. Переработка алгоритма получения текста |
| При изменении темы страницы, тема на другой странице не сохраняется | Не сохранение состояния страницы | Добавление кода на JavaScript который локально сохраняет состояние |
| Все пользователи имеют доступ к тестам | Не задана логика запрещающая логика программы | Добавлена логика сохранения и изменения доступности тестов |

# Методический раздел

Программа Электронный учебник «Разработка игр на Unreal Engine 5 с помощью blueprint» состоит из разделов: теория, практика, тестирование и справка. В теоретический раздел входит материал [1].

В практический раздел входит материал [2] и предусматривает подготовку к выполнению практических работ по темам: «Создание простой локации в Unreal Engine 5», «Управляемый шар», «Интерактивный уровень».

В разделе тестирование содержатся вопросы по пройденному материалу.

Вопросы с вариантами ответов теста приведены в таблице 3.1

Таблица 3.1 – Перечень вопросов с вариантами ответов

| Вопрос | Варианты ответа | Правильный ответ |
| --- | --- | --- |
| Что такое Blueprint в Unreal Engine? | 1 система для рисования уровней  2 система визуального программирования  3 язык программирования  4 редактор текстур | 2 |
| Какие платформы поддерживает Unreal Engine? | 1 только PC  2 только консоли  3 только мобильные  4 PC, консоли и мобильные | 4 |
| Что такое Epic Games Launcher? | 1 приложение для управления Unreal Engine  2 редактор уровней  3 программа для создания музыки  4 браузер | 1 |
| Какой тип аккаунта требуется для использования Unreal Engine? | 1 Платный  2 учебный  3 бесплатный  4 премиум | 3 |
| Какое минимальное место на диске требуется для установки Unreal Engine? | 1 10 Гб  2 25 Гб  3 55 Гб  4 100 Гб | 3 |
| Что такое уровень в Unreal Engine? | 1 трехмерное пространство игры  2 скрипт для персонажа  3 текстура объекта  4 модель здания | 1 |
| Что такое статические объекты? | 1 объекты, которые могут двигаться  2 объекты, которые не меняются  3 объекты, которые меняют цвет  4 объекты с анимацией | 2 |
| Какие есть источники света в Unreal Engine? | 1 только солнечный свет  2 только точечный свет  3 Directional, Point, Spot, Area Light  4 только искусственный свет | 3 |
| Что такое Lumen? | 1 система для создания анимаций  2 система глобального освещения  3 редактор материалов  4 инструмент для звука | 2 |
| Как можно сделать мир более живым в Unreal Engine? | 1 добавить текстуры и настроить материалы  2 добавить только статические объекты  3 использовать только один источник света  4 не добавлять ничего | 1 |
| Что такое наследование в контексте классов? | 1 добавление новых функций  2 удаление старых функций  3 когда дочерний класс получает свойства родительского  4 переименование классов | 3 |
| Какой класс является базовым для всех объектов в Unreal Engine? | 1 Actor  2 Pawn  3 StaticMesh  4 Light | 1 |
| Что такое Blueprint? | 1 текстовый редактор  2 консольный терминал  3 визуальная схема для создания игровых объектов  4 звуковой редактор | 3 |
| Зачем нужно использовать триггеры? | 1 для изменения текстур  2 для создания новых объектов  3 для запуска событий при взаимодействии с игроком  4 для управления освещением | 3 |
| Какие возможности дают блюпринты? | 1 добавление только статических объектов  2 настройка только освещения  3 создание нескольких экземпляров объектов  4 управление только звуком | 3 |

# Заключение

В ходе курсового проектирования был разработан Электронный учебник «Разработка игр на Unreal Engine 5 с помощью blueprint».

Программа Электронный учебник «Разработка игр на Unreal Engine 5 с помощью blueprint» выполняет функции:

* теоретического материала;
* практического материала;
* тестирования

Проектное решение реализовано в среде программирования «Intellij IDEA Community Edition». Для полученной программы в дальнейшем возможна ее модернизация. Для этого потребуются следующие программные средства: Intellij IDEA, Google Chrome.

В итоге полученный программный продукт может применяться для самостоятельного   изучения  темы «Разработка игр на Unreal Engine 5 с помощью blueprint» теоретического раздела, подготовки к выполнению практических работ и проверке знаний теоретического и практического материалов.

# Информационные источники

HTML/CSS/JS:

* - HTML & CSS Tutorial and Projects Course (freeCodeCamp) (https://www.freecodecamp.org/learn/2022/responsive-web-design/) — бесплатный курс.
* - PDF: "Eloquent JavaScript" (Marijn Haverbeke) (https://eloquentjavascript.net/Eloquent\_JavaScript.pdf) — книга по JS, открытый доступ.

Python/Flask:

* - Official Flask Documentation (https://flask.palletsprojects.com/en/3.0.x/) — бесплатная дока, можно указать как источник.
* - PDF: "Python Crash Course" (Eric Matthes) (https://www.nostarch.com/sites/default/files/pcc2e\_chapter1.pdf) — глава из книги, бесплатная.

Unreal Engine:

* - Unreal Engine Official Documentation (https://docs.unrealengine.com/5.3/en-US/) — бесплатная официальная дока.
* - Unreal Engine Blueprint Tutorial (Epic Games) (https://dev.epicgames.com/community/learning/courses/VV/unreal-engine-blueprints-quickstart) — бесплатный курс от Epic.

# Приложение А

(обязательное)

Текст программы

import hashlib

import re

from flask import Flask, render\_template, request, redirect, url\_for, session, abort, jsonify

from myDb import get\_db\_connection, createUser

app = Flask(\_\_name\_\_)

app.secret\_key = 'myBoY'

PATH\_EDU\_MATERIAL = [

"static/theories/theory",

"static/practicate/practicate",

"static/tests/test"

]

# region Здесь функции для упрощения

def get\_path\_edu\_material(index, indexOfMat):

return PATH\_EDU\_MATERIAL[index] + f"{indexOfMat}.txt"

def read\_edu\_material(index, indexOfMat):

if indexOfMat == '': return ""

path = get\_path\_edu\_material(index, indexOfMat)

return read\_file(path)

def write\_edu\_material(index, indexOfMat, text):

if indexOfMat == '': return ""

path = get\_path\_edu\_material(index, indexOfMat)

write\_file(path, text)

return

def read\_file(path):

with open(path, 'r', encoding="utf-8") as f:

text = f.read()

return text

def write\_file(path, string):

with open(path, 'w', encoding="utf-8") as f:

f.write(string.replace('\r\n', '\n'))

def get\_text\_as\_components(text):

components = []

while len(text) > 0:

# Найти индекс первой новой строки

int\_index = text.find('\n')

# Если новой строки нет, берем остаток текста

if int\_index == -1:

line = text

text = ''

else:

line = text[:int\_index]

text = text[int\_index + 1:] # Убираем обработанную строку

# Определяем тип компонента

if line.startswith('###'):

components.append({'type': 'h2', 'content': line[3:].strip()})

elif line.startswith('##'):

components.append({'type': 'h1', 'content': line[2:].strip()})

else:

components.append({'type': 'p', 'content': line.strip()})

return components

def get\_questions\_as\_components(text):

questions = []

lines = text.strip().split('\n')

i = 0

while i < len(lines):

# Извлекаем номер вопроса и текст вопроса

match = re.match(r"(\d+)(.\*)", lines[i])

if match:

i\_num = int(match.group(1))

question = match.group(2).strip()

else:

i += 1

continue

i += 1

# Извлекаем варианты ответов и правильный ответ

if i < len(lines):

options = lines[i].split('|')

correct\_answer = None

cleaned\_options = []

for option in options:

if option.startswith('\*'):

correct\_answer = option[1:]

cleaned\_options.append(option[1:])

else:

cleaned\_options.append(option)

if correct\_answer is None:

correct\_answer = cleaned\_options[0]

questions.append({

'i': i\_num,

'question': question,

'options': cleaned\_options,

'correct\_answer': correct\_answer

})

i += 1

else:

break

return questions

def get\_users():

conn = get\_db\_connection()

usersDb = conn.execute('SELECT \* FROM users').fetchall() # Получаем все записи

conn.close()

# Обработка данных

users = []

for user in usersDb:

user\_data = {

'id': user[0],

'login': user[1],

# 'password': user[2],

'admin': user[3]

}

users.append(user\_data)

return users

# endregion

# region Events for processing user actions

@app.route('/', methods=["GET", "POST"])

def index():

return render\_template('main.html')

@app.route('/theory', methods=["GET", "POST"])

def theory():

theory1 = get\_text\_as\_components(read\_file(get\_path\_edu\_material(0, 1)))

theory2 = get\_text\_as\_components(read\_file(get\_path\_edu\_material(0, 2)))

theory3 = get\_text\_as\_components(read\_file(get\_path\_edu\_material(0, 3)))

theory4 = get\_text\_as\_components(read\_file(get\_path\_edu\_material(0, 4)))

theory5 = get\_text\_as\_components(read\_file(get\_path\_edu\_material(0, 5)))

return render\_template('theory.html', theory1=theory1, theory2=theory2, theory3=theory3, theory4=theory4,

theory5=theory5)

@app.route('/practicate', methods=["GET", "POST"])

def practicate():

lab1 = get\_text\_as\_components(read\_file(get\_path\_edu\_material(1, 1)))

lab2 = get\_text\_as\_components(read\_file(get\_path\_edu\_material(1, 2)))

lab3 = get\_text\_as\_components(read\_file(get\_path\_edu\_material(1, 3)))

return render\_template('practicate.html', pract1=lab1, pract2=lab2, pract3=lab3)

@app.route('/tests', methods=["GET", "POST"])

def tests():

tab = request.args.get('test', '')

check = get\_test\_availability().get\_json()

if check.get('test'+tab) == 0 and not session.get('admin'):

return redirect('/tests')

test1 = get\_questions\_as\_components(read\_file(get\_path\_edu\_material(2, 1)))

test2 = get\_questions\_as\_components(read\_file(get\_path\_edu\_material(2, 2)))

test3 = get\_questions\_as\_components(read\_file(get\_path\_edu\_material(2, 3)))

if request.method == 'POST':

user\_answers = request.form

score = 0

count\_questions = 0

current\_test = -1

if tab == "1":

current\_test = test1

elif tab == "2":

current\_test = test2

elif tab == "3":

current\_test = test3

if current\_test != -1:

for component in current\_test:

question\_number = component['i']

correct\_answer = component['correct\_answer']

user\_answer = user\_answers.get(f'answer\_{question\_number}')

count\_questions += 1

if user\_answer == correct\_answer:

score += 1

score = (score / count\_questions) \* 100

return render\_template('tests.html', test=tab, score=score, test1=test1, test2=test2, test3=test3)

return render\_template('tests.html', test=tab, test1=test1, test2=test2, test3=test3)

@app.route('/getTest', methods=["GET", "POST"])

def check\_availability\_test():

index\_test = request.args.get("index", "")

if session.get('admin'):

return jsonify({'status': 'success', 'redirect': f'tests?test={index\_test}'})

c = get\_db\_connection()

test\_availability = c.execute('SELECT availability FROM tests WHERE id = ?', (index\_test,)).fetchone()

c.close()

if bool(test\_availability['availability']):

return jsonify({'status': 'success', 'redirect': f'tests?test={index\_test}'})

return jsonify({'status': 'error', 'message': 'Похоже учитель еще не открыл доступ к этому тесту.'})

@app.route('/getTestAvailability', methods=["GET", "POST"])

def get\_test\_availability():

c = get\_db\_connection()

tests = c.execute("SELECT \* FROM tests").fetchall()

return jsonify(

{'test1': tests[0]['availability'], 'test2': tests[1]['availability'], 'test3': tests[2]['availability']})

@app.route('/changeAvailabilityTest', methods=["GET", "POST"])

def change\_test\_availability():

i = request.get\_json()

c = get\_db\_connection()

now\_availability = c.execute("SELECT availability FROM tests WHERE id = ?", (i,)).fetchone()

if now\_availability["availability"] == 1:

now\_availability = 0

else:

now\_availability = 1

c.execute("UPDATE tests SET availability = ? WHERE id = ?", (now\_availability, i,))

c.commit()

c.close()

return jsonify('success')

@app.route('/admin', methods=["GET", "POST"])

def admin\_panel():

if not session.get('admin'):

abort(404)

if request.method == 'POST':

action = request.form.get('form\_id')

if action == 'save\_user':

login = request.form.get('login')

password = request.form['password']

newpassword = hashlib.sha256(password.encode('utf-8')).hexdigest()

admin = request.form.get('admin', '0')

conn = get\_db\_connection()

if password == "":

conn.execute('UPDATE users SET admin = ? WHERE username = ?', (admin, login,)).fetchone()

else:

conn.execute('UPDATE users SET password = ?, admin = ? WHERE username = ?',

(newpassword, admin, login,)).fetchone()

conn.commit()

conn.close()

elif action == 'delete\_user':

login = request.form.get('login')

conn = get\_db\_connection()

conn.execute('DELETE FROM users WHERE username = ?', (login,)).fetchone()

conn.commit()

conn.close()

elif action == 'create\_user':

login = request.form.get('newLogin')

password = request.form.get('newPassword')

admin = request.form.get('newAdmin', '0')

if login == "" or password == "":

return render\_template('admin\_panel.html', values\_error='create\_user', active\_tab='accounts',

accounts=get\_users())

createUser(login, password, admin)

elif action == 'save\_lectures':

index = request.args.get('theory', '')

text = request.form.get('lectures')

write\_edu\_material(0, index, text)

elif action == 'save\_labs':

index = request.args.get('lab', '')

text = request.form.get('labs')

write\_edu\_material(1, index, text)

elif action == 'save\_tests':

index = request.args.get('test', '')

text = request.form.get('test')

write\_edu\_material(2, index, text)

tab = request.args.get('tab', 'intro')

theory = request.args.get('theory', '')

theory = read\_edu\_material(0, theory)

lab = request.args.get('lab', '')

lab = read\_edu\_material(1, lab)

test = request.args.get('test', '')

test = read\_edu\_material(2, test)

return render\_template('admin\_panel.html', active\_tab=tab, theory=theory, lab=lab, test=test, accounts=get\_users())

@app.route('/login', methods=["GET", "POST"])

def login():

if request.method == 'POST':

login = request.form.get('login')

passw = request.form.get('password')

hashed\_password = hashlib.sha256(passw.encode('utf-8')).hexdigest()

conn = get\_db\_connection()

user = conn.execute('SELECT \* FROM users WHERE username = ?', (login,)).fetchone()

conn.close()

if user and user['password'] == hashed\_password:

if user['admin'] == 1:

session['admin'] = True

session['username'] = login

return redirect(url\_for('index'))

else:

return render\_template('login.html', passw=False)

return render\_template('login.html')

@app.route('/logout', methods=["GET", "POST"])

def logout():

session.clear()

return redirect(url\_for('index'))

@app.route('/change-theme', methods=['POST'])

def set\_theme():

data = request.get\_json()

theme = data.get('theme')

session['theme'] = theme

return jsonify({'message': 'Тема успешно сохранена!', 'theme': theme}), 200

@app.route('/change-theme', methods=['GET'])

def get\_theme():

theme = session.get('theme', 'light')

return jsonify({'theme': theme}), 200

@app.errorhandler(404)

def not\_found(e):

return render\_template('404.html'), 404

# endregion

if \_\_name\_\_ == "\_\_main\_\_":

app.run()

import sqlite3

import hashlib

def get\_db\_connection():

conn = sqlite3.connect('database.db')

conn.row\_factory = sqlite3.Row

return conn

def createUser(username, password, admin):

# Хеширование пароля

hashed\_password = hashlib.sha256(password.encode('utf-8')).hexdigest()

# Подключение к нашей базе данных

conn = get\_db\_connection()

c = conn.cursor()

# Добавление нового пользователя

c.execute('INSERT INTO users (username, password, admin) VALUES (?, ?, ?)', (username, hashed\_password, admin))

# Сохранение изменений и закрытие соединения с базой данных

conn.commit()

conn.close()

def createTables():

conn = get\_db\_connection()

c = conn.cursor()

c.execute('''CREATE TABLE IF NOT EXISTS users (

id INTEGER PRIMARY KEY AUTOINCREMENT,

username TEXT UNIQUE,

password TEXT,

admin BOOLEAN)''')

c.execute('''CREATE TABLE IF NOT EXISTS tests (

id INTEGER PRIMARY KEY AUTOINCREMENT,

name TEXT,

availability BOOLEAN) ''')

conn.commit()

conn.close()

get\_db\_connection()

createTables()

#createUser('миша', 'миша', 0)

#createUser('Елена', 'елена', 1)

let currentTheme = localStorage.getItem('theme') || 'light';

document.addEventListener('DOMContentLoaded', function() {

// Применяем сохраненную тему при загрузке страницы

applyTheme();

document.getElementById('theme-change').addEventListener('click', function() {

changeTheme();

// Сохраняем новую тему в localStorage

localStorage.setItem('theme', currentTheme);

// Отправляем новую тему на сервер

fetch('/change-theme', {

method: 'POST',

headers: {

'Content-Type': 'application/json',

},

body: JSON.stringify({ theme: currentTheme }),

})

.then(response => response.json())

.then(data => {

console.log(data.message);

})

.catch(error => console.error('Ошибка:', error));

});

});

function changeTheme() {

currentTheme = (currentTheme === 'light') ? 'dark' : 'light';

applyTheme();

}

function applyTheme() {

if (currentTheme === 'dark') {

document.body.classList.add('dark-mode');

document.getElementById('theme-change-lb').innerText = '🌙';

} else {

document.body.classList.remove('dark-mode');

document.getElementById('theme-change-lb').innerText = '☀️';

}

}