

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  
федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Российский экономический университет им. Г.В. Плеханова»  
Московский приборостроительный техникум

## Курсовой проект

ПМ 02 Осуществление интеграции программных модулей

МДК 02.01 Технология разработки программного обеспечения

Специальность 09.02.07 «Информационные системы и  
программирование»

Квалификация: Программист

Тема: «Разработка информационной системы контроля освоения знаний  
и практических умений обучающихся (на примере ФГБОУ ВО РЭУ им.  
Г.В. Плеханова)»

## Пояснительная записка

Листов: 35

Руководитель

\_\_\_\_\_ / Л.А. Соколова  
«\_\_\_\_» \_\_\_\_\_ 2025 год

Исполнитель

\_\_\_\_\_ / С.А. Куртева  
«\_\_\_\_» \_\_\_\_\_ 2025 год

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение высшего образования  
«Российский экономический университет имени Г.В. Плеханова»  
**МОСКОВСКИЙ ПРИБОРОСТРОИТЕЛЬНЫЙ ТЕХНИКУМ**

«УТВЕРЖДАЮ»  
Заместитель директора по учебной работе  
\_\_\_\_\_ Д.А. Клопов  
«\_\_\_\_» 2025 г.

## ЗАДАНИЕ

на выполнение курсового проекта (курсовой работы)

**Куртевой Светлане Андреевне**

(фамилия, имя, отчество студента — полностью)

студенту группы П50-3-21 специальности 09.02.07 «Информационные системы и программирование» по МДК 02.01 «Технология разработки программного обеспечения»

1. Исходные данные к проекту (работе):

1.1. Тема: «Разработка информационной системы контроля освоения знаний и практических умений обучающихся (на примере ФГБОУ ВО РЭУ им. Г.В. Плеханова)»

1.2. Состав курсового проекта:

1.2.1. Задание КП

1.2.2. Пояснительная записка

1.2.3. Программа (исходные данные) на электронном носителе

1.2.4. Презентация и инсталляционный пакет программы на электронном носителе

1.3. Содержание пояснительной записи:

### ВВЕДЕНИЕ

#### 1. ОБЩАЯ ЧАСТЬ

1.1. Цель разработки

1.2. Средства разработки

#### 2. СПЕЦИАЛЬНАЯ ЧАСТЬ

2.1. Постановка задачи

2.1.1. Входные данные

2.1.2. Выходные данные

2.1.3. Подробные требования к проекту

2.2. Внешняя спецификация

2.2.1. Описание задачи

2.2.2. Входные и выходные данные

2.2.3. Методы

2.2.4. Тесты

2.3. Проектирование

2.3.1. Схема архитектуры приложения

2.3.2. Логическая схема данных

2.3.3. Физическая схема данных

2.3.4. Диаграмма классов

2.3.5. Функциональная схема

2.4. Результат работы программы

### 3. ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ

3.1. Инstrumentальные средства

3.2. Отладка программы

3.3. Защитное программирование

3.4. Характеристики программы

### ЗАКЛЮЧЕНИЕ

### СПИСОК ИСПОЛЬЗУЕМЫХ МАТЕРИАЛОВ

### ПРИЛОЖЕНИЕ А. Техническое задание

### ПРИЛОЖЕНИЕ Б. Текст программы

### ПРИЛОЖЕНИЕ В. Программа испытаний

## ПРИЛОЖЕНИЕ Г. Руководство пользователя

## ПРИЛОЖЕНИЕ Д. Скрипт базы данных

2. Содержание задания по проекту (работе) — перечень вопросов, подлежащих разработке

	Разрабатываемый вопрос	Объем от всего задания, %	Срок выполнения
A	Описательная часть проекта (введение, общее описание и т. д.)	5	24.01.2025
1.	Введение	-	24.01.2025
2.	Цель разработки	-	24.01.2025
3.	Средства разработки	-	24.01.2025
B	Анализ задачи и её постановка	15	31.01.2025
1.	Определение требований к программе	-	31.01.2025
2.	Спецификация программы (описание задачи, описание входных и выходных данных, методы)	-	31.01.2025
3.	Тесты, контроль целостности данных	-	31.01.2025
B	Проектирование и реализация	55	14.02.2025
1.	Схемы проекта (схема архитектуры, логическая схема данных, физическая схема данных, функциональная и структурная схемы, диаграмма классов, схема тестирования, схема пользовательского интерфейса)	-	14.02.2025
2.	Реализация в инструментальной среде	-	28.02.2025
Г	Технологическая часть проекта	5	14.03.2025
1.	Инструментальные средства разработки	-	14.03.2025
2.	Отладка программы	-	14.03.2025
3.	Защитное программирование	-	14.03.2025
4.	Характеристика программы	-	14.03.2025
Д	Программная документация	10	28.03.2025
1.	ПРИЛОЖЕНИЕ А. Техническое задание	-	28.03.2025
2.	ПРИЛОЖЕНИЕ Б. Текст программы	-	28.03.2025
3.	ПРИЛОЖЕНИЕ В. Программа испытаний	-	28.03.2025
4.	ПРИЛОЖЕНИЕ Г. Руководство пользователя	-	28.03.2025
5.	ПРИЛОЖЕНИЕ Д. Скрипт базы данных	-	28.03.2025
E	Экспериментальная часть проекта	10	04.04.2025
1.	Программа на машинном носителе. Информация на носителе разбита на разделы: эксплуатационный пакет, тексты программы, документация.	-	04.04.2025

Руководитель курсового проекта (работы) Соколова Лариса Алексеевна, преподаватель

«21» января 2025 года \_\_\_\_\_ / Л.А. Соколова /

Дата выдачи курсового задания

«21» января 2025 года

Срок сдачи законченного проекта (работы)

«04» апреля 2025 года

Задание принял к исполнению

«21» января 2025 года \_\_\_\_\_ / С.А. Куртева /

## СОДЕРЖАНИЕ

Введение .....	3
1. Общая часть .....	5
1.1. Цель разработки.....	5
1.2. Средства разработки.....	5
2. Специальная часть.....	8
2.1. Постановка задачи .....	8
2.1.1. Входные данные .....	11
2.1.2. Выходные данные.....	11
2.1.3. Подробные требования к проекту.....	12
2.2. Внешняя спецификация .....	12
2.2.1. Описание задачи .....	12
2.2.2. Входные и выходные данные.....	13
2.2.3. Методы .....	16
2.2.4. Тесты .....	18
2.3.Проектирование.....	20
2.3.1. Схема архитектуры приложения .....	20
2.3.2. Логическая схема данных .....	21
2.3.3. Физическая схема данных .....	22
2.3.4. Диаграмма классов .....	27
2.3.4. Функциональная схема.....	28
2.4. Результат работы программы .....	28
3. ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ .....	31
3.1. Инструментальная среда разработки.....	31
Заключение .....	33

Список используемых материалов .....	34
--------------------------------------	----

ПРИЛОЖЕНИЕ А. Техническое задание

ПРИЛОЖЕНИЕ Б. Текст программы

ПРИЛОЖЕНИЕ В. Программа испытаний

ПРИЛОЖЕНИЕ Г. Руководство пользователя

ПРИЛОЖЕНИЕ Д. Скрипт базы данных

## ВВЕДЕНИЕ

В связи с современной тенденцией и развитием современных технологий, которые уже используются, практические, во всех сферах жизнедеятельности человека, то один из самых важных аспектов развития человека не мог пройти мимо данную тенденцию стороной. Система образования всегда развивалась в ногу со временем, предоставляя все больше возможностей для получения знаний и их закрепления, поэтому появление систем онлайн-образований было только делом времени. Данный формат обучения может предоставить студентам или же обучающимся возможность обучаться в том формате и темпе, который будет удобен для них, изучать те темы, которые интересны и могут быть полезны по их мнению. Система онлайн-образования также помогает и преподавателям, что дает возможность вести прямой контакт со студентами на расстоянии, выдавать лекционный материал и проводить оценку знаний. Составление, выдача и проверка тестов отнимает много времени и сил, таким образом и появилась идея по реализации веб-приложения «Система онлайн-образования», которое сможет предоставить студентам возможность обучаться в удобном формате, а преподавателям автоматизировать процесс выдачи и проверки тестов.

Так как подразумевается система онлайн-образования, то предстоит реализовать клиент-серверное веб-приложение с использованием, в качестве передачи данных из базы данных в веб-приложение – API.

Целью курсовой работы является повысить знания в области разработки веб-приложений на фреймворке «Django», а также по реализации API.

Задачей курсовой работы является оптимизация процесса обучения и автоматизация деятельности преподавателя по выдаче тестов и проверке их для оценивания знаний студентов.

Для определения подробностей предметной области, необходимо провести ее анализ с помощью построения бизнес-процессов, выполняющихся без использования информационной системы, определением функциональных возможностей каждого из ролей, по средству по строение диаграммы

прецедентов и определением поток данных, пока происходит выполнение бизнес-процесса.

## 1. ОБЩАЯ ЧАСТЬ

### 1.1. Цель разработки

Целью разработки веб-приложения «Информационная система контроля освоения знаний и практических умений обучающихся (на примере ФГБОУ ВО РЭУ им. Г.В. Плеханова)» является предоставления возможности удобной организации взаимодействия студентов и преподавателя в процессе обучения. Проект позволит оптимизировать учебный процесс и автоматизировать проведение тестовых срезов по изученным темам.

### 1.2. Средства разработки

Для проектирования, разработки и тестирования веб-приложения «Информационная система контроля освоения знаний и практических умений обучающихся (на примере ФГБОУ ВО РЭУ им. Г.В. Плеханова)» были использованы программные средства, представленные в Таблице 1.

Таблица 1 -Программные средства разработки веб-приложения

№	Тип средства	Название средства	Назначение
1	2	3	4
1	Операционная система	Microsoft Windows 11	Организация взаимодействия программ и пользователя
2	Инструментальные средства разработки программного обеспечения	Microsoft Visual Studio 2022 17.9.6	Разработка API
3	Инструментальные средства разработки программного обеспечения	Microsoft Visual Studio Code	Разработка веб-приложения
4	СУБД	SQL Server Management Studio 18	Разработка базы данных
5	Браузер	Google Chrome ver 111.0.5563.65	Просмотр API, просмотр веб-приложения

В Таблице 2 представлены минимальные и рекомендованные системные требования, на базе которых возможно комфортное использование реализуемого веб-приложения.

Таблица 2 - Минимальные и рекомендованные системные требования

№	Тип оборудования	Наименование оборудования
1	2	3
Минимальные системные требования		

№	Тип оборудования	Наименование оборудования
1	2	3
Персональный компьютер		
1	Операционная система	Microsoft Windows 7, macOS 10.12, Linux Ubuntu 16.04
2	Браузер	Google Chrome 80+, Mozilla Firefox 78+, Microsoft Edge 80+, Safari 12+
3	Процессор	Процессор с 2 физическими ядрами с тактовой частотой 1,5 ГГц
4	Оперативная память	4 Гб
5	Видеокарта	Интегрированная с поддержкой WebGL
6	Свободное место на накопителе	1 Гб
7	Пропускная способность интернет-соединения	3 Мбит/с
8	Размер экрана	14,0
9	Разрешение экрана	1366x768
Рекомендованные системные требования		
Персональный компьютер		
1	Операционная система	Windows 10/11, macOS 12+, Linux (Ubuntu 22.04+)
2	Браузер	Google Chrome (Последней версии), Mozilla Firefox (Последней версии), Samsung Internet (Последней версии), Safari (Последней версии)
3	Процессор	Процессор с 4 физическими ядрами с тактовой частотой 2,5 ГГц
4	Оперативная память	8 Гб+
5	Видеокарта	Дискретная или интегрированная, поддержка WebGL 2.0
6	Свободное место на накопителе	1 Гб и больше
7	Пропускная способность интернет-соединения	25 Мбит/с
8	Размер экрана	14,0
9	Разрешение экрана	1920x1080

В качестве средств вычислительной техники при разработке и для использования веб-приложения использовался ноутбук Asus Zenbook 14 UM433IQ, в Таблице 3 представлены подробные характеристики.

Таблица 3 - Подробные характеристики средства вычислительной техники

№	Тип средства	Название средства
1	2	3
Для разработки и использования		
Ноутбук Asus Zenbook 14 UM433IQ		
1	Размер экрана	14
2	Разрешение экрана	1920x1080
3	Линейка процессора	AMD Ryzen 7 4700U with Radeon Graphics

№	Тип средства	Название средства
1	2	3
4	Количество ядер процессора	8
5	Оперативная память	16 Гб
6	Видеокарта	NVIDIA GeForce MX350
7	Конфигурация накопителей	SSD
8	Общий объем всех накопителей	1000

## 2. СПЕЦИАЛЬНАЯ ЧАСТЬ

### 2.1. Постановка задачи

На рисунке 1 представлен основной бизнес-процесс «Контролирование освоения знаний и практических умений обучающихся» модели То-Ве. Ниже расположены задачи Веб-приложения (основные бизнес-процессы)

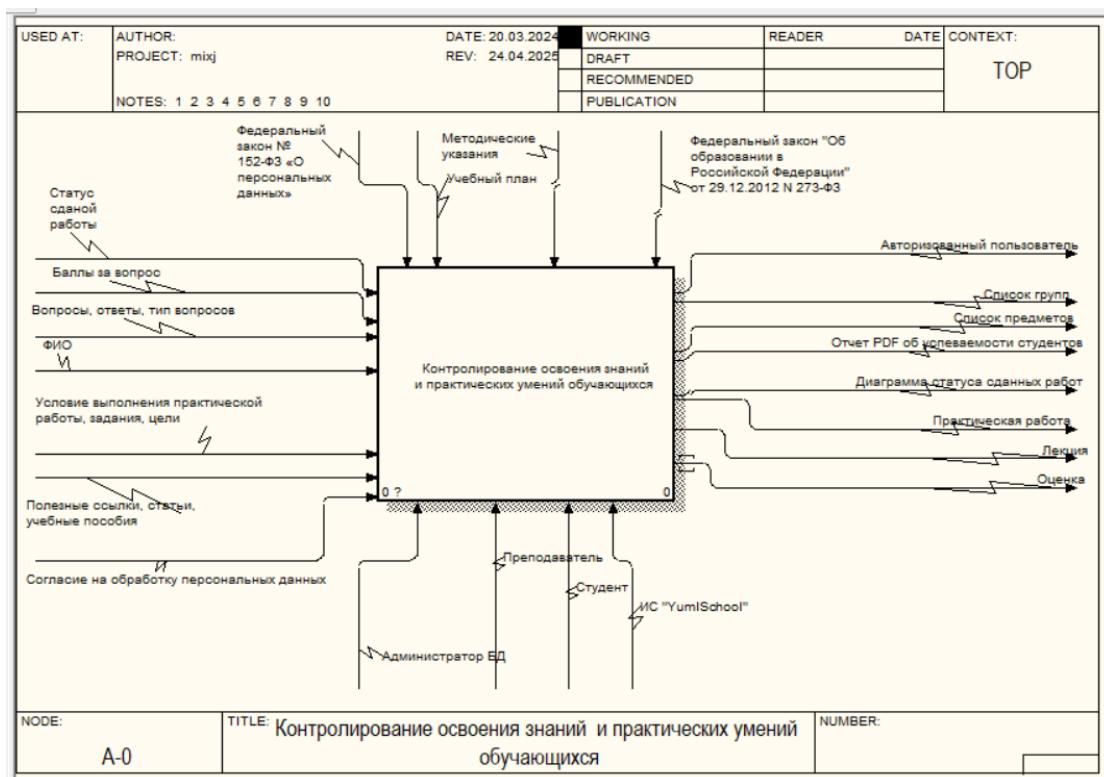


Рисунок 1 - Основной бизнес-процесс (То-Ве)

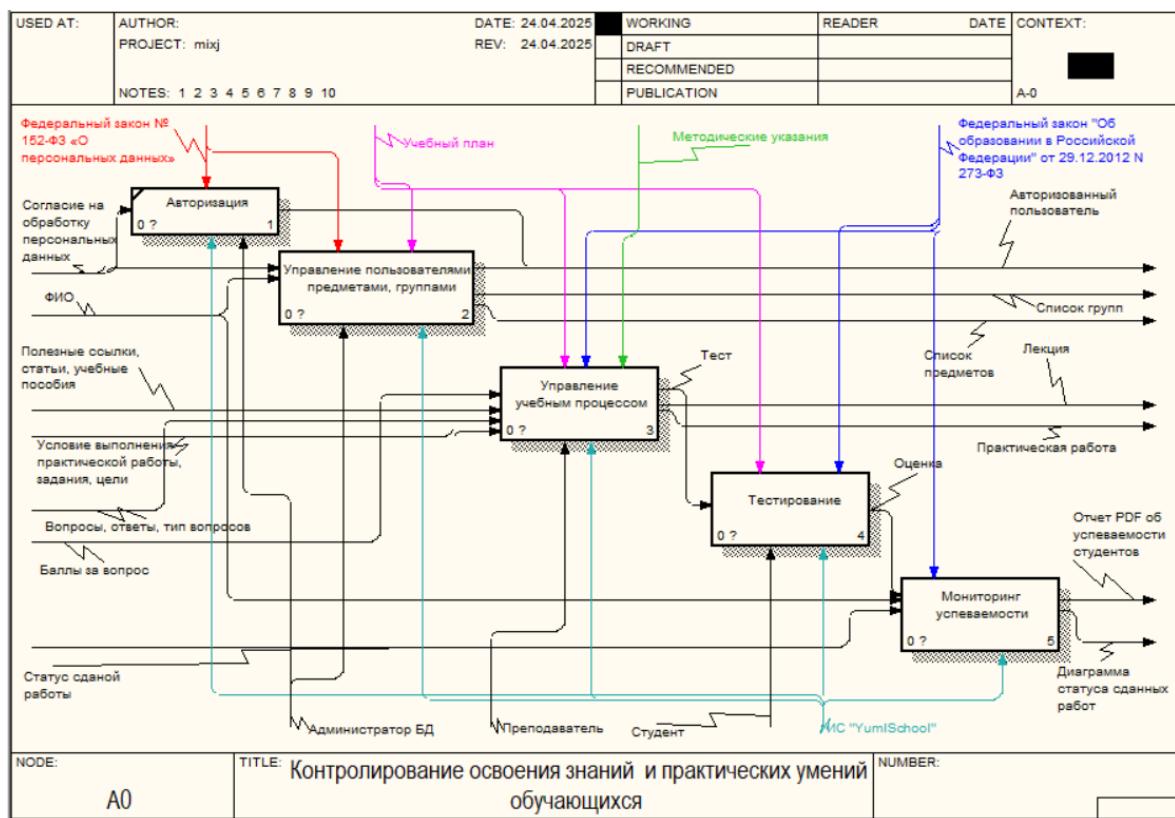


Рисунок 2 - Декомпозиция основного процесса

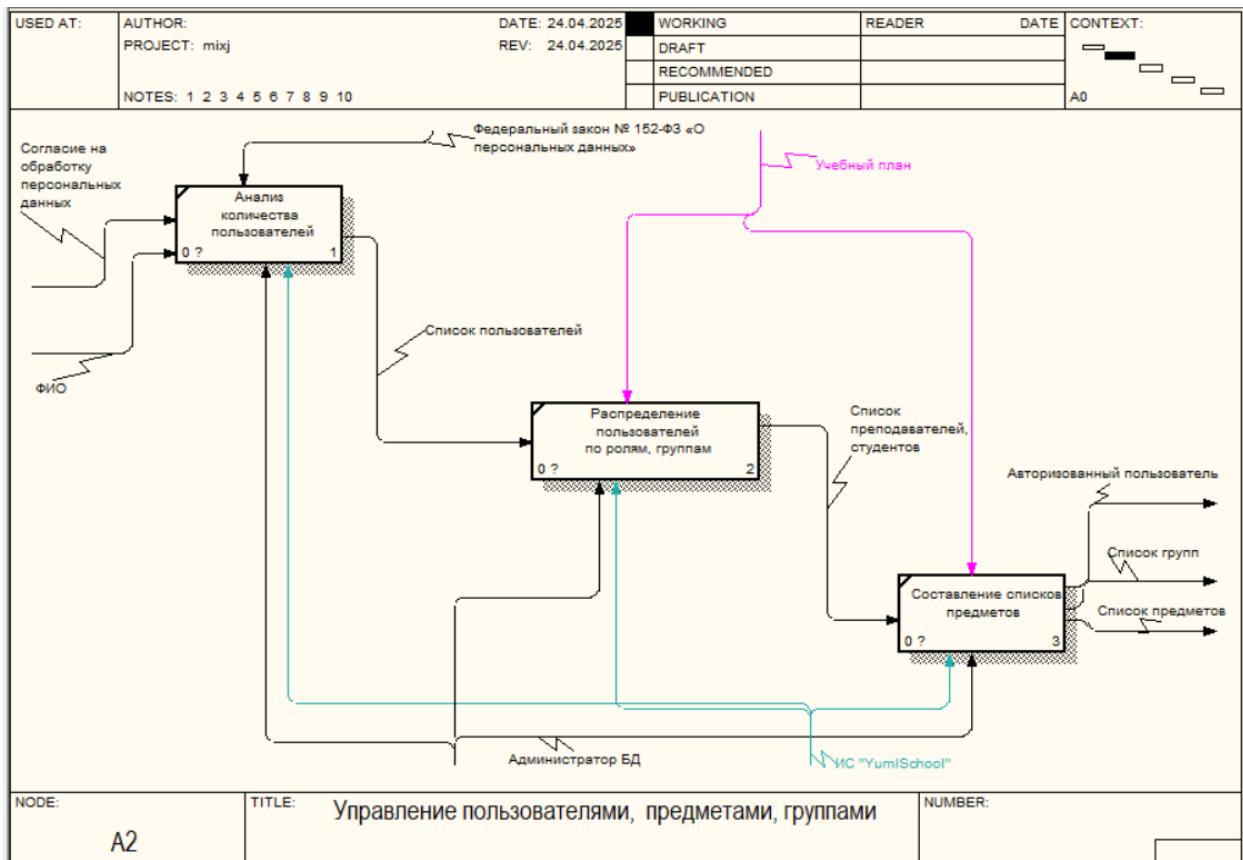


Рисунок 3 - Декомпозиция управления пользователями, предметами, группами

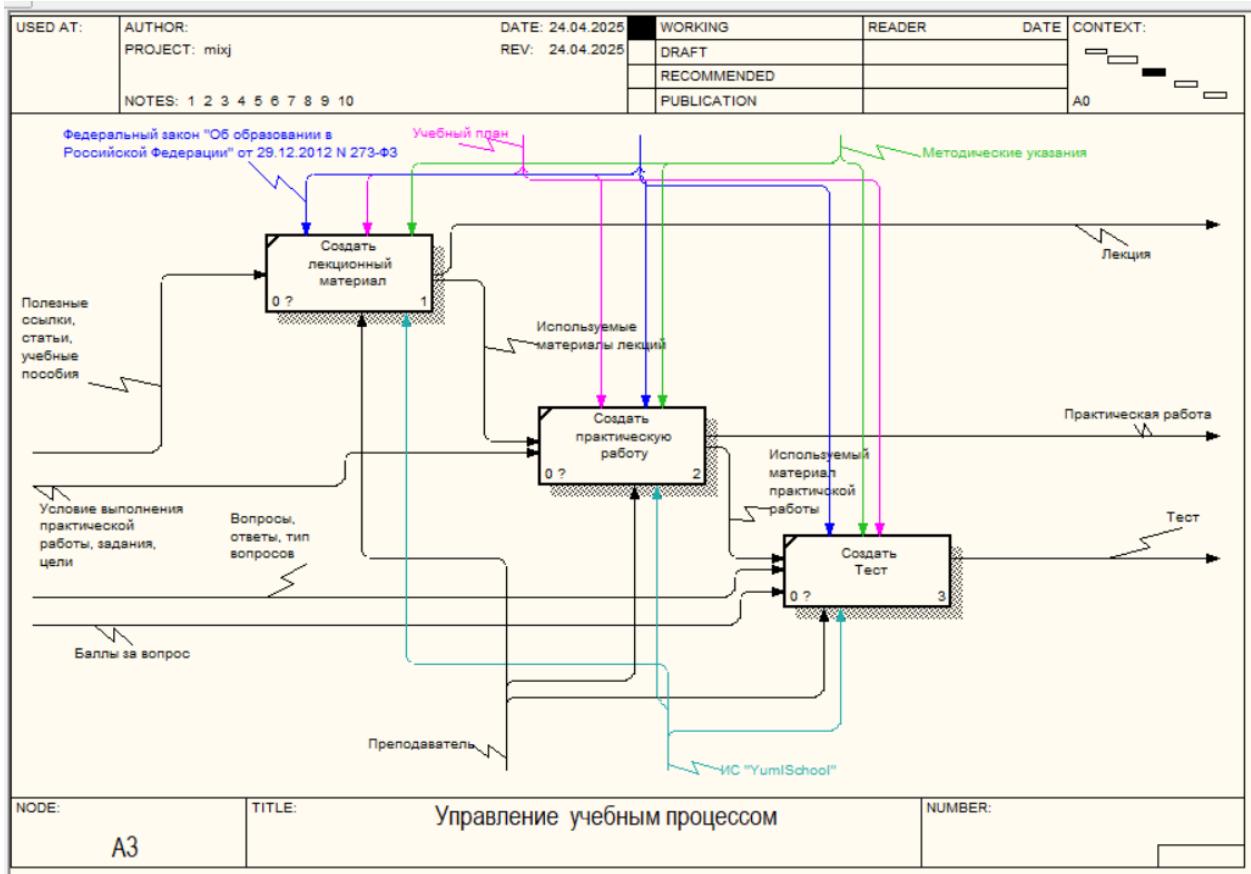


Рисунок 4 - Декомпозиция управления учебным процессом

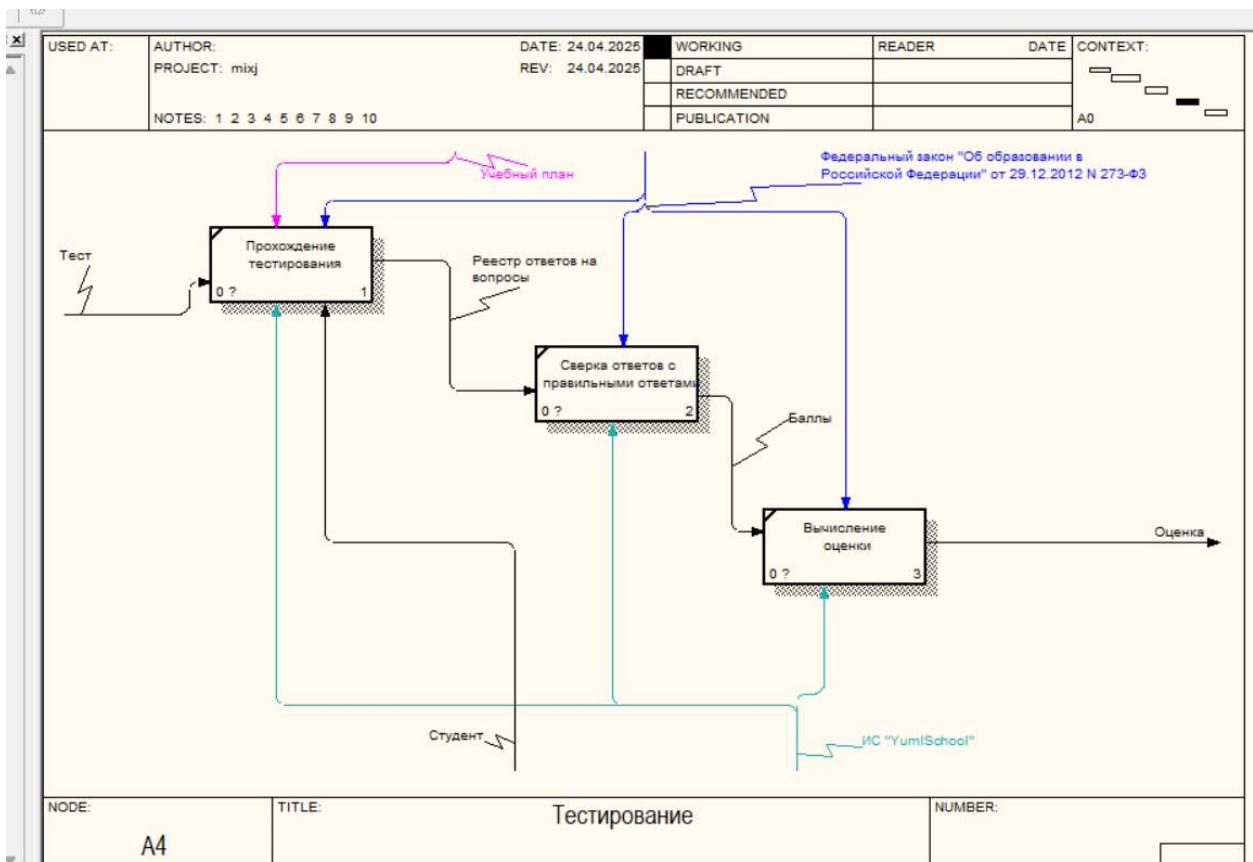


Рисунок 5 - Декомпозиция тестирования

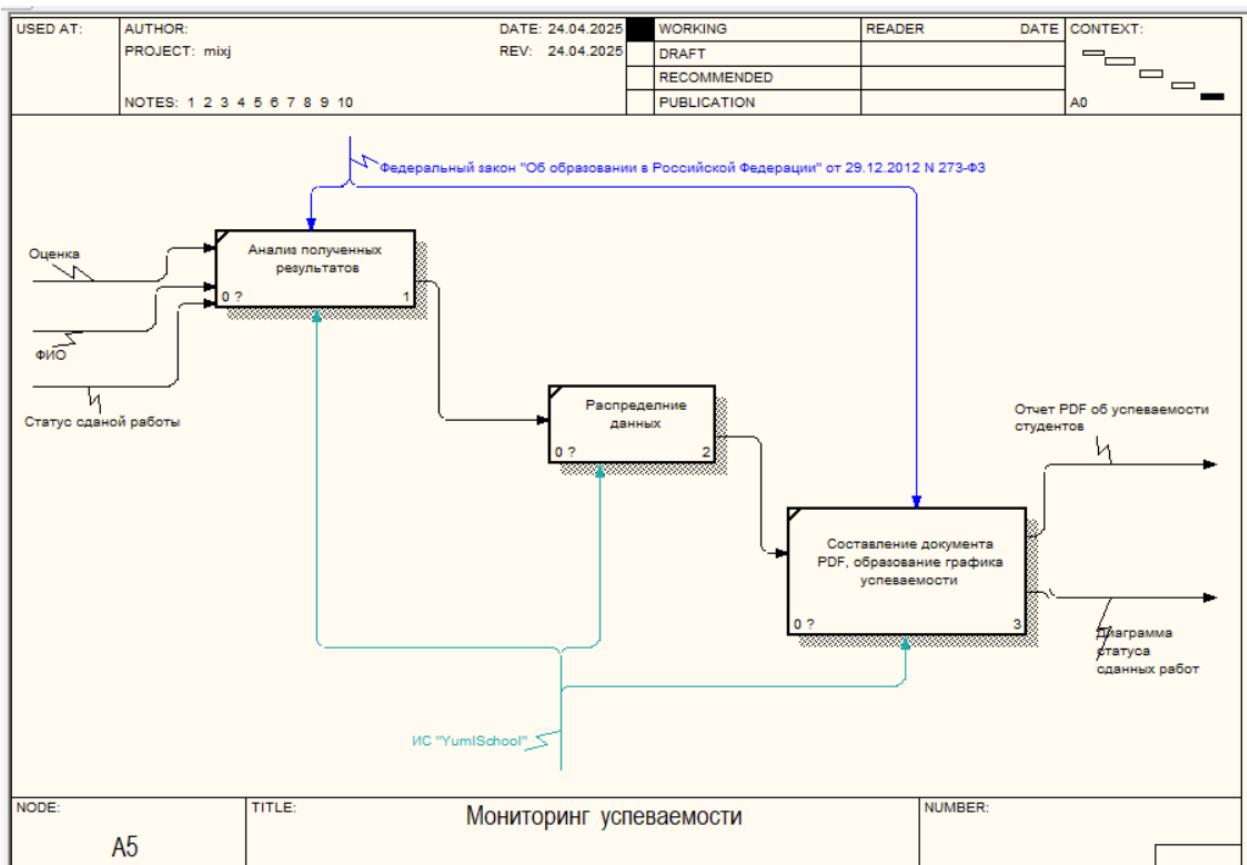


Рисунок 6 - Декомпозиция мониторинга успеваемости

### 2.1.1. Входные данные

Входные данные Веб-приложения представлены в следующем виде:

- Данные о пользователе (ФИО, почта, логин и пароль)
- Данные о практической работе (Название, требования, ссылка на файл), лекции (Название, ссылка на файл), тесте (Название, вопросы, ответы, баллы, критерии оценивания)
- Данные о статусе сданной работы (Статус, оценка)
- Данные о предметах (Название, группа, преподаватель)

### 2.1.2. Выходные данные

Выходные данные Веб-приложения представлены в следующем виде:

- Данные о практической работе, teste (Оценка, баллы)
- Данные об выгруженной отчетности и диаграммах (Отчет об успеваемости формата PDF, диаграмма статусов работы)
- Данные о пользователях (Список групп, предметов, пользователей)

### 2.1.3. Подробные требования к проекту

Подробные требования к разработке Web-приложения «YumlSchool» представлены в Приложении А. Техническое задание.

## 2.2. Внешняя спецификация

### 2.2.1. Описание задачи

Разработать Web-приложение для контроля освоений знаний и практических умений обучающихся.

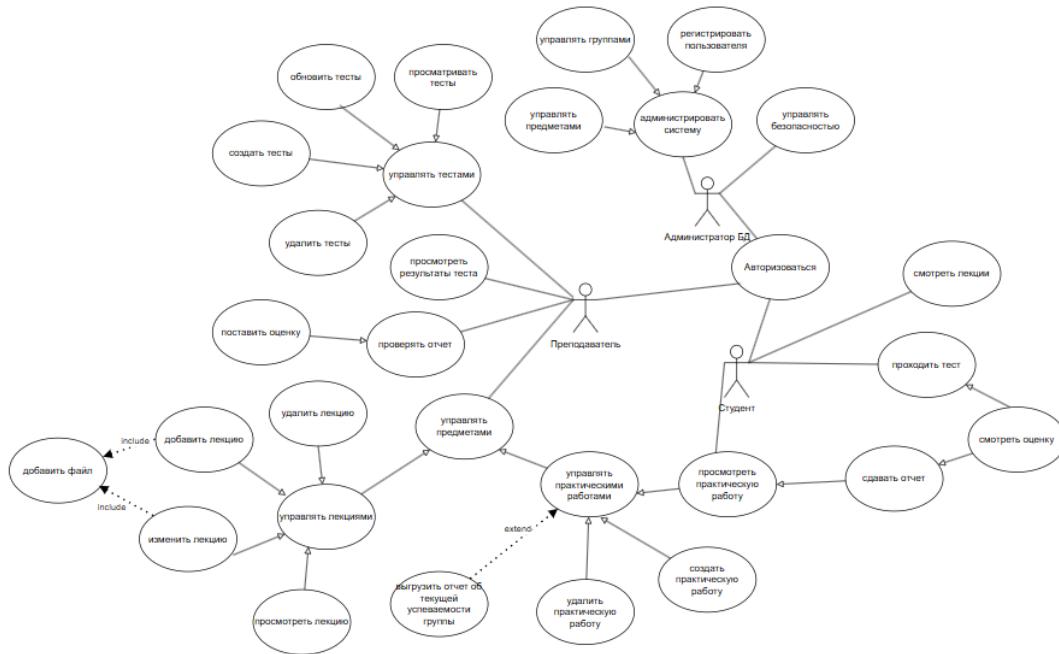
Данное программное решение должно состоять из реляционной базы данных под управлением СУБД Microsoft SQL Server Management Studio.

Приложение должно иметь возможность отображения различной информации, хранящейся в базе данных, с помощью технологии API. Помимо отображения данных должно быть предоставлено пользователю возможности добавления, изменения и удаления различной информации.

В приложении предусмотрены три роли:

- Администратор БД отвечает за администрирование системы.
- Преподавателю доступны управление тестами, лекциями, практическими работами и их проверка.
- Студенту доступно взаимодействие с практическими работами, прохождение тестирования, просмотр лекций.

Описание перечня пользователей с их функциями изображено на рисунке 7.



### Рисунок 7 - Диаграмма прецедентов

## 2.2.2. Входные и выходные данные

В таблице 4 представлены входные данные.

Таблица 4. Входные данные

Имя	Тип	Ограничение	Формат ввода	Описание
1	2	3	4	5
Персональные данные				
Login	Строка	[A-Za-z0-9@]{5, 100}	Поле ввода	Логин пользователя
Password	Строка	[A-Za-z0-9@]{5, 100}	Поле ввода	Пароль пользователя
First_Name	Строка	[А-Яа-я]{2, 45}	Поле ввода	Имя пользователя
Second_Name	Строка	[А-Яа-я]{2, 45}	Поле ввода	Фамилия пользователя

Middle_Name	Строка	[А-Яа-я]{2,45}	Поле ввода	Отчество пользователя
Данные лекции				
Name_Lection	Строка	[A-Za-z0-9@]{5, 100}	Поле ввода	Название лекции
Description_lection	Строка	[A-Za-z0-9@]{5, 100}	Поле ввода	Описание лекции
URL_file_lection	Строка	Нет ограничений	Поле ввода	Ссылка на файл лекции
Данные практической работы				
Name_PracWork	Строка	[A-Za-z0-9@]{5, 100}	Поле ввода	Название практической работы
Description_Name_PracWork	Строка	[A-Za-z0-9@]{5, 100}	Поле ввода	Описание практической работы
URL_file_Name_PracWork	Строка	Нет ограничений	Поле ввода	Ссылка на файл практической работы
Данные о тесте				
Name_Test	Строка	[A-Za-z0-9@]{5, 100}	Поле ввода	Название теста
Description_Name_Test	Строка	[A-Za-z0-9@]{5, 100}	Поле ввода	Описание теста
Question	Строка	[A-Za-z0-9@]{5, 100}	Поле ввода	Вопрос

Answer	Строка	[A-Za-z0-9@] {5, 100}	Поле ввода	Ответ
Данные о предмете				
Subject_Name	Строка	[А-Яа-я]{2, 45}	Поле ввода	Название предмета

В таблице 5 представлены выходные данные.

Таблица 5. Выходные данные

Имя	Тип	Ограничение	Описание
1	2	3	4
Выходные данные			
Персональные данные			
First_Name	Строка	[А-Яа-я]{2, 45}	Имя пользователя
Second_Name	Строка	[А-Яа-я]{2, 45}	Фамилия пользоваться
Middle_Name	Строка	[А-Яа-я]{2, 45}	Отчество пользователя
Данные о лекции			
Name_Lection	Строка	[A-Za-z0-9@] {5, 100}	Название лекции
Description_lection	Строка	[A-Za-z0-9@] {5, 100}	Описание лекции
URL_file_lection	Строка	Нет ограничений	Ссылка на файл лекции
Данные о практической работе			

Name_PracWork	Строка	[A-Za-z0-9@] {5, 100}	Название практической работы
Description_Name_PracWork	Строка	[A-Za-z0-9@] {5, 100}	Описание практической работы
URL_file_Name_PracWork	Строка	Нет ограничений	Ссылка на файл практической работы
Grade_PracWork	Строка	[2-5]	Оценка за практическую работу
Данные о teste			
Grade_Test	Строка	[2-5]	Оценка за тест
Score_Test	Строка	[2-5]	Балл за тест
Name_Test	Строка	[A-Za-z0-9@] {5, 100}	Название теста
Description_Name_Test	Строка	[A-Za-z0-9@] {5, 100}	Описание теста
Question	Строка	[A-Za-z0-9@] {5, 100}	Вопрос
Answer	Строка	[A-Za-z0-9@] {5, 100}	Ответ

### 2.2.3. Методы

В ходе написания программы были использованы следующие методы разработки программного обеспечения:

При разработке API использовалась методология ООП, а именно абстракция, в виде классов моделей с характеристиками объекта, которым обладает он же в БД, инкапсуляция, в виде методов, позволяющих скрыть подробности характеристик объекта, которые возвращают значения и

наследование, каждый контроллер наследуется от родительского класса «ControllerBase».

Также использовалась методология REST API – это архитектурный стиль взаимодействия между клиентов и сервером по протоколу HTTP. REST API использует принципы REST:

- Клиент-серверная архитектура – клиент делает запрос на сервер, сервер обрабатывает запросы и отправляет клиенту ответ
- Отсутствие состояния – сервер не хранит состояние клиента между запросами;
- Единообразие интерфейса – ресурсы представлены в виде URL, для всех ресурсов используется одни и те же HTTP-методы (GET, POST, PUT, DELETE).

Для разработки веб-приложения использовался паттерн MVC (Model-View-Controller), но в рамках фреймворка «Django» он именуется как MTV (Model-Template-View). В данном паттерне подразумевается, что модели (Model) используются для работы с базой данных, описывает структуру данных и логику их обработки; шаблон (Template) отвечает за отображение данных пользователю – это файлы HTML, в которых можно динамически выводить данные, переданные из представлений; представления (View) прослойка между моделями и шаблонами, которая позволяет управлять логикой обработки запросов, т.е. представления запрашивают данные из моделей, после чего передают их в шаблоны для вывода на страницу. Схема паттерна MTV представлена на Рисунке 8.

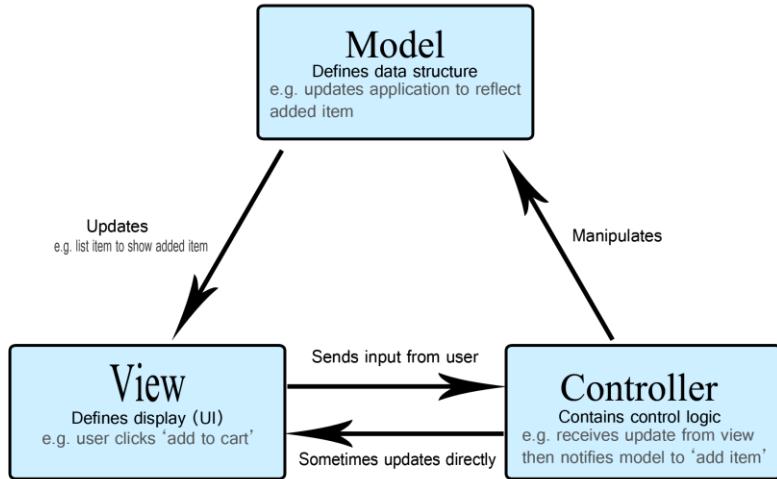


Рисунок 8 - Схема паттерна MTV

Для разработки веб-приложения, использовалась технология программирования – Django. Представляющую свою реализацию в виде паттерна MTV (подробнее в п. 2.2.3).

Для разработки веб-API, использовалась технология программирования – ASP.NET Core WEB-API, с использованием EntityFramework.

Для разработки базы данных, использовалась технология программирования – Microsoft SQL Server Management.

#### 2.2.4. Тесты

##### 1. По формальности тестирования.

Тестирование по тестам – тестирование по предварительно написанным тест-кейсам.

##### 2. По исполнению кода.

Динамическое тестирование - во время тестирования код исполняется.

##### 3. По уровню тестирования.

Системное – проверка работы всей системы на соответствие заявленным требованиям к программному продукту.

##### 4. По целям.

Функциональное тестирование направлено на проверку того, какие функции ПО реализованы, и того, насколько верно они реализованы.

5. По степени автоматизации.

Ручное – без использования дополнительных программных средств.

6. По позитивности сценария.

Позитивным – проверка ПО на соответствие ожидаемому поведению.

Негативным – проверяет, будет ли ПО работать в случае, когда поведение пользователя отличается от ожидаемого.

7. По знанию системы.

Тестирование «белого ящика» – тестирование программного продукта с доступом к коду.

Тестирование «черного ящика» – тестирование без доступа к коду продукта.

8. По разработке тестовых испытаний.

На основе требований – требование было определено до начала тестирования.

Контроль целостности, описывающих ситуации и реакции приложения на выполнения функций представлен в таблице

Таблица 4 - Тест возможности регистрации

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/н е пройден)
1	2	3	4	5
1	Логин: vasin3485 Пароль: vasim4ek	Успешная авторизация	Успешная авторизация	Тест пройден
2	Логин: vasin3485 Пароль: 123	Авторизация не прошла. Над полями ввода отображается надпись «Неверный пароль»	Авторизация не прошла. Над полями ввода отображается надпись «Неверный пароль»	Тест пройден
3	Логин: vasin34851 Пароль: vasim4ek	Авторизация не прошла. Над полями ввода отображается надпись «Такого	Авторизация не прошла. Над полями ввода отображается надпись «Такого логина не существует»	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/н е пройден)
1	2	3	4	5
		логина не существует»		
4	Логин: пустота Пароль: пустота	Авторизация не прошла. Над полем ввода логина и пароля отображается сообщение «Заполните это поле.»	Авторизация не прошла. Над полем ввода логина и пароля отображается сообщение «Заполните это поле.»	Тест пройден
5	Логин: vasin3485 Пароль: пустота	Авторизация не прошла. Над полем ввода пароля отображается сообщение «Заполните это поле»	Авторизация не прошла. Над полем ввода пароля отображается сообщение «Заполните это поле»	Тест пройден
6	Логин: пустота Пароль: vasim4ek	Авторизация не прошла. Над полем ввода логина отображается сообщение «Заполните это поле.»	Авторизация не прошла. Над полем ввода логина отображается сообщение «Заполните это поле.»	Тест Пройден

## 2.3.Проектирование

### 2.3.1. Схема архитектуры приложения

На Рисунке 9 представлена схема «Клиент-серверной» архитектуры веб-приложения.

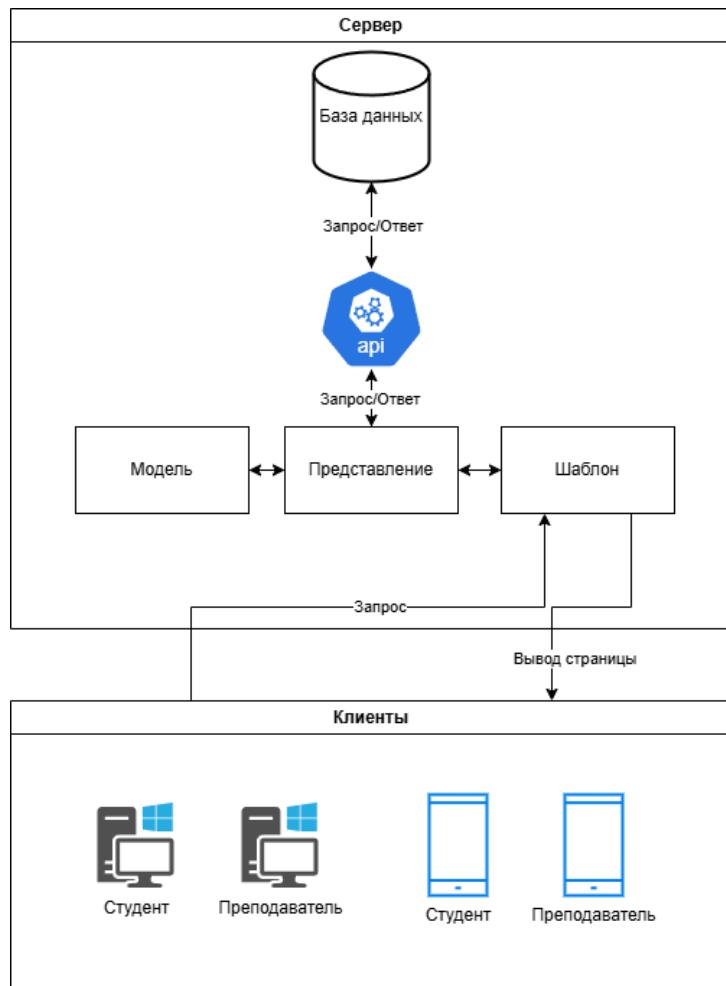


Рисунок 9 - Схема архитектуры веб-приложения

### 2.3.2. Логическая схема данных

Разработанная логическая модель базы данных, представленная на Рисунке 10, послужила основой для реализации логики манипуляции данными в проектируемой базе данных.

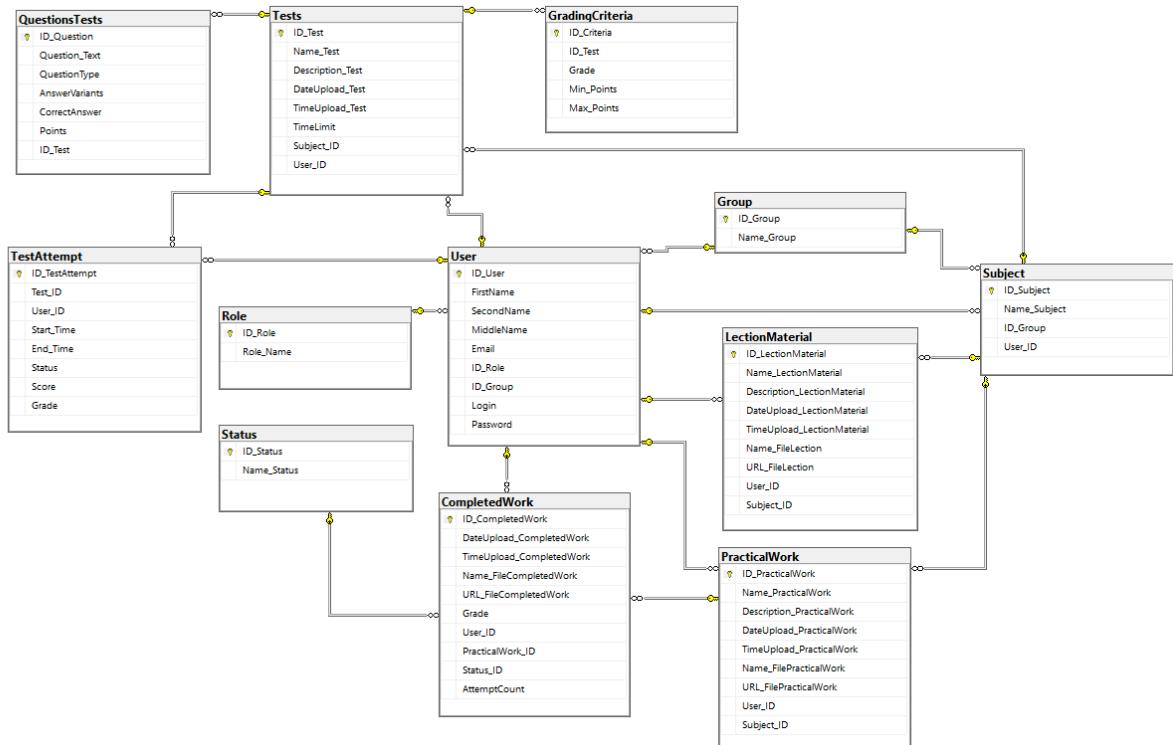


Рисунок 10 - Логическая схема данных

### 2.3.3. Физическая схема данных

На Рисунке 11 представлена физическая схема данных.

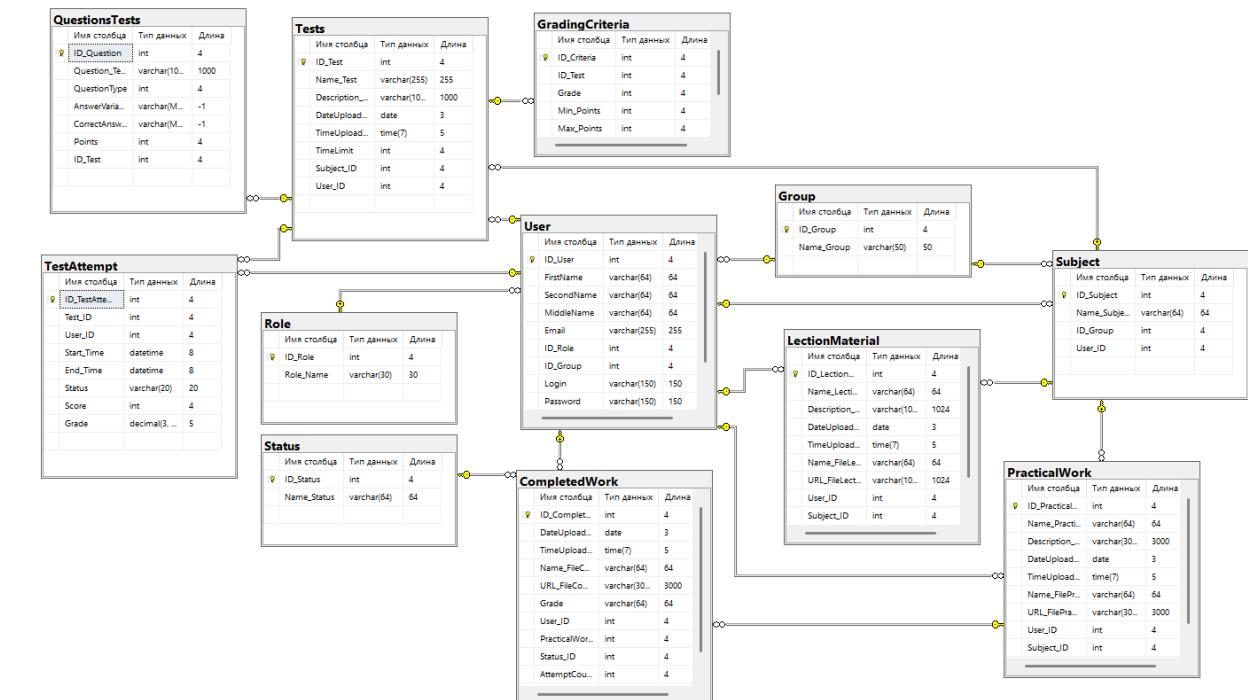


Рисунок 11 - Физическая схема данных

В проектируемой базе данных было выделено 12 сущностей (таблиц), каждая из которых содержит набор атрибутов (полей) для записи информации.

В таблице 6 представлен словарь данных реализуемой базы данных для данного мобильного приложения.

Таблица 6. Словарь

Ключ	Поле	Тип данных	Обязательность заполнения	Описание
1	2	3	4	5
PK	ID_TestAttempt	Int	Not Null	Идентификатор попытки теста
FK	Test_ID	int	Not Null	Код теста
FK	User_ID	int	Not Null	Код пользователя
	Start_Time	datetime	Not Null	Начало времени
	End_Time	datetime	Не обязательно	Конец времени
	Status	varchar(20)	Not Null	Статус
	Score	int	Not Null	Выполненная работа
	Grade	decimal(3, 1)	Not Null	Оценка за тест
PK	ID_Criteria	int	Not Null	Идентификатор критерия теста
FK	ID_Test	int	Not Null	Код теста
	Grade	int	Not Null	Оценка критерия
	Min_Points	int	Not Null	Минимальное количество баллов

	Max_Points	int	Not Null	Максимальное количество баллов
PK	ID_Test	int	Not Null	Идентификатор теста
	Name_Test	varchar(255)	Not Null	Название
	Description_Test	varchar(1000)	Не обязательно	Описание
	DateUpload_Test	date	Не обязательно	Дата выдачи
	TimeUpload_Test	time(7)	Не обязательно	Время выдачи
	TimeLimit	int	Не обязательно	Лимит теста
FK	Subject_ID	int	Не обязательно	Код предмета
FK	User_ID	int	Не обязательно	Код пользователя
PK	ID_LectionMaterial	int	Not Null	Идентификатор лекции
	Name_LectionMaterial	varchar(64)	Not Null	Название лекции
	Description_LectureMaterial	varchar(1024)	Not Null	Описание
	DateUpload_LectureMaterial	date	Не обязательно	Дата выдачи
	TimeUpload_LectureMaterial	time(7)	Не обязательно	Время выдачи
	Name_FileLecture	varchar(64)	Not Null	Название для файла лекции
	URL_FileLecture	varchar(1024)	Not Null	URL лекции
FK	User_ID	int	Не обязательно	Код пользователя

FK	Subject_ID	int	Не обязательно	Код предмета
PK	ID_Subject	int	Not Null	Идентификатор предмета
	Name_Subject	varchar(64)	Not Null	Название предмета
FK	ID_Group	int	Not Null	Код группы
FK	User_ID	int	Not Null	Код пользователя
PK	ID_Group	int	Not Null	Идентификатор группы
	Name_Group	varchar(50)	Not Null	Название группы
PK	ID_Question	int	Not Null	Идентификатор вопроса в тесте
	Question_Text	varchar(1000)	Not Null	Текст вопроса
	QuestionType	int	Not Null	Тип вопроса
	AnswerVariants	varchar(MAX)	Not Null	Вариант ответа
	CorrectAnswer	varchar(MAX)	Not Null	Правильный ответ
	Points	int	Не обязательно	Баллы за тест
FK	ID_Test	int	Не обязательно	Код теста
PK	ID_User	int	Not Null	Идентификатор пользователя
	FirstName	varchar(64)	Not Null	Имя
	SecondName	varchar(64)	Not Null	Фамилия
	MiddleName	varchar(64)	Не обязательно	Отчество
	Email	varchar(255)	Not Null	Почта

FK	ID_Role	int	Not Null	Код роли
FK	ID_Group	int	Not Null	Код группы
	Login	varchar(150)	Not Null	Логин
	Password	varchar(150)	Not Null	Пароль
PK	ID_Group	int	Not Null	Идентификатор группы
	Name_Group	varchar(50)	Not Null	Название группы
PK	ID_Role	int	Not Null	Идентификатор роли
	Role_Name	varchar(30)	Not Null	Название роли
PK	ID_Status	int	Not Null	Идентификатор статуса
	Name_Status	varchar(64)	Not Null	Название статуса
PK	ID_CompletedWork	int	Not Null	Идентификатор выполненной работы
	DateUpload_CompletedWork	date	Не обязательно	Дата сдачи
	TimeUpload_Co mpletedWork	time(7)	Не обязательно	Время сдачи
	Name_FileCompl etedWork	varchar(64)	Not Null	Название файла
	URL_FileComple tedWork	varchar(3000)	Not Null	URL файла
	Grade	varchar(64)	Not Null	Оценка
FK	User_ID	int	Не обязательно	Код пользователя

FK	PracticalWork_ID	int	Не обязательно	Код практической работы
FK	Status_ID	int	Не обязательно	Статус работы
	AttemptCount	int	Не обязательно	Количество попыток
PK	ID_PracticalWork	int	Not Null	Идентификатор практической работы
	Name_PracticalWork	varchar(64)	Not Null	Название практической
	Description_PracticalWork	varchar(3000)	Not Null	Описание
	DateUpload_PracticalWork	date	Не обязательно	Дата выдачи
	TimeUpload_PracticalWork	time(7)	Не обязательно	Время выдачи
	Name_FilePracticalWork	varchar(64)	Not Null	Название файла
	URL_FilePracticalWork	varchar(3000)	Not Null	URL файла
FK	User_ID	int	Не обязательно	Код пользователя
FK	Subject_ID	int	Не обязательно	Код предмета

#### 2.3.4. Диаграмма классов

На рисунке 12 представлена диаграмма классов Web-приложения. На ней графически изображены классы, которые состоят из полей, методов и свойств.

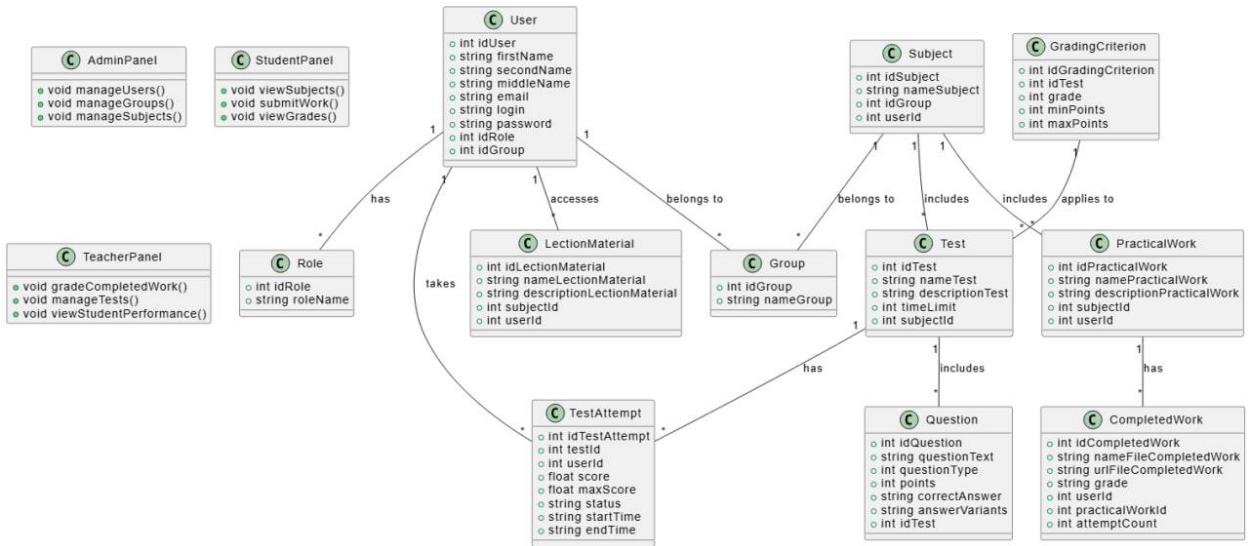


Рисунок 12 - Диаграмма классов

### 2.3.4. Функциональная схема

На рисунке 13 изображена функциональная схема

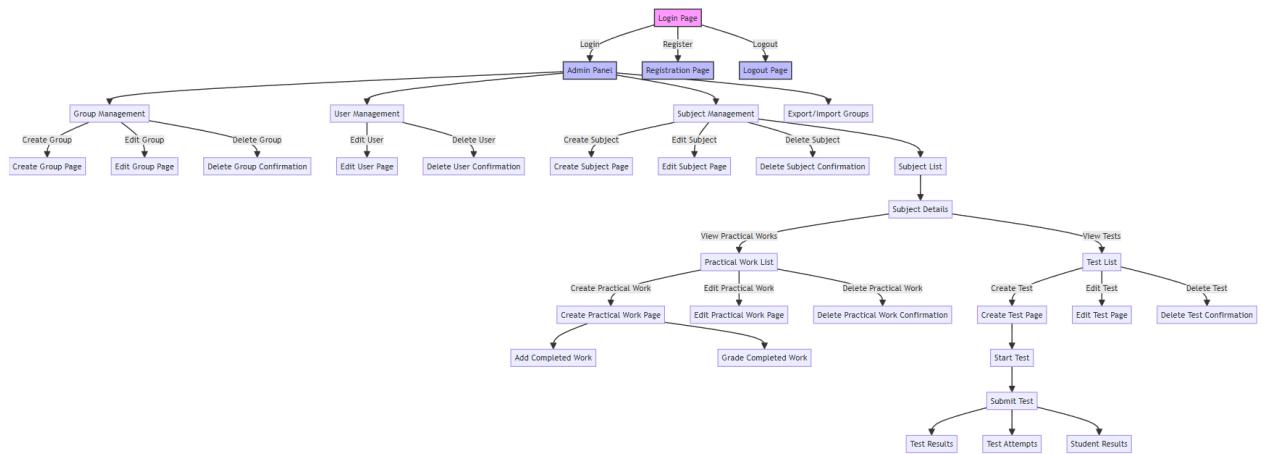


Рисунок 13 - Функциональная схема

### 2.4. Результат работы программы

На рисунке 14 видны элементы основного окна для студента, здесь видны:

- Данные о сессии пользователя, его фамилия и имя, кнопка «Выход» - для выхода из системы и возврата на страницу авторизации.
- Данные о предметах пользователя в которых он состоит. Здесь видны название предмета, группа, преподаватель предмета. По нажатию на предмет происходит переход на страницу предмета.

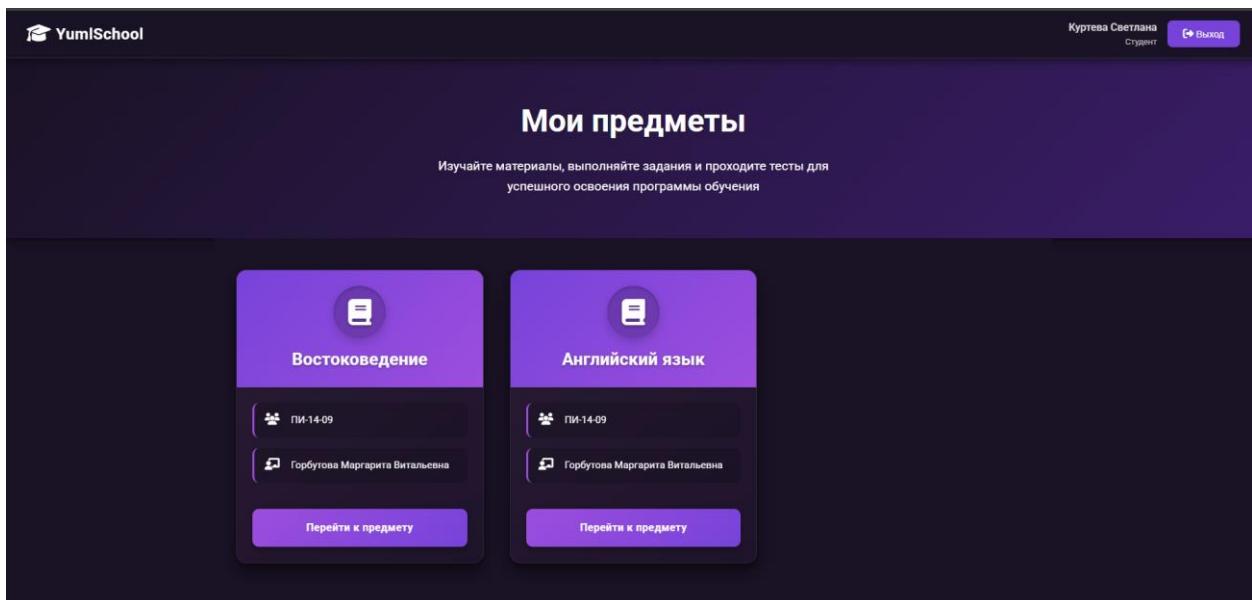


Рисунок 14 - Элементы основного окна у студента

На рисунке 15 изображена авторизация пользователей

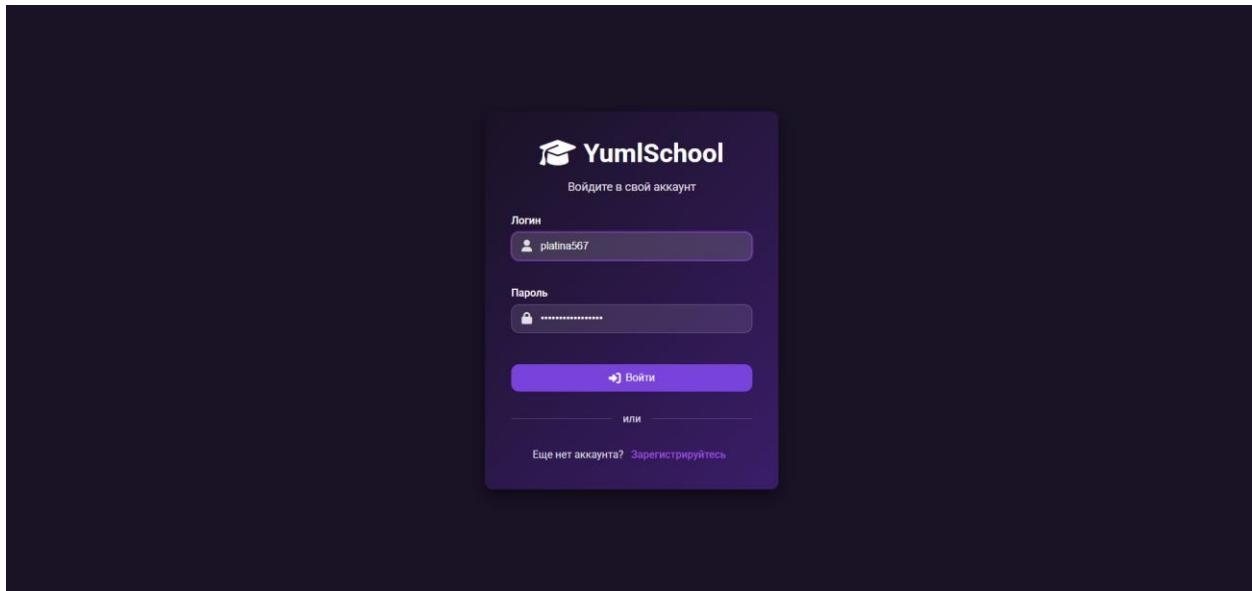


Рисунок 15 – Авторизация

На рисунке 16 изображена страница результатов тестирования

**Тест №1 - Результаты студентов**

тест по теме востоковедение

### Общая статистика

3	Всего студентов
1	Сдали тест
2	Не сдали тест

### Результаты по студентам

Имя студента	Попыток: 1	Лучший результат: 4 баллов	Оценка: 3.0
Светлана Куртева Андреевна	1	4	3.0
Иван Дремин Александрович	0	Нет попыток	Нет оценки
Максим Андреев Александрович	0	Нет попыток	Нет оценки

[← Вернуться к тесту](#)

Рисунок 16 - Результаты тестирования

### 3. ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ

#### 3.1. Инструментальная среда разработки

Для написания базы данных была выбрана СУБД – SSMS. Выбор обусловлен тем, что разработка серверной части подразумевается на языке программирования «C#», с использованием технологии «ASP.NET Core».

Для написания Веб-API был выбран язык программирования «C#» и интеграционная среда разработки «Visual Studio 2022» и технология программирования – Веб-API ASP.Net Core. Выбор был сделан в пользу данного инструмента потому, что Visual Studio имеет непосредственно напрямую подключаться к базе данных СУБД SSMS. Также не маловажным фактом было наличие обширного количества сторонних библиотек, добавление проект которых, происходит посредством использования пакетов NuGet. Фреймворк, использованный при разработке API, был EntityFramework, позволяющий автоматически, создать модели, контроллеры и контекст базы данных, на основе подключения к самой базе данных, с последующей возможность вручную изменять, добавлять или же удалять модели или контроллеры, добавляя необходимые эндпоинты. Также EntityFramework имеет множество функций, позволяющих выполнять LINQ-запросы с List в асинхронном потоке, что значительно ускоряет выполнения и отправки ответа назад клиенту.

В качестве редактора программного кода, для верстки страниц был выбран «Visual Studio Code 1.78.2». Главным фактором в выборе данного редактора было наличие маркетплейса с внушительным количеством расширений, которые позволяют упростить и сделать процесс разработки комфортней и продуктивней. Второстепенным фактором является возможность работы с файлами любого формата.

Для написания веб-приложения «Информационная система контроля освоения знаний и практических умений обучающихся (на примере ФГБОУ ВО РЭУ им. Г.В. Плеханова)» был выбран язык программирования «Python», текстовый редактор «Microsoft Visual Studio Code» и технология

программирования – «Django», данный выбор обусловлен тем, что «Django» предоставляет всё необходимое для разработки веб-приложения изначально, что позволяет использовать уже готовые и протестированные функции вместо реализации их с нуля. Структура проекта «Django» позволяет реализовывать разбиение логики на приложения – это означает, что проект можно без усилий масштабировать, добавляя в него новый функционал. «Django» автоматически предоставляет защиту для веб-приложения от таких критических уязвимостей как: XSS и CSRF-атак, и SQL-инъекций.

## ЗАКЛЮЧЕНИЕ

Главной задачей данной курсовой работы было: оптимизировать учебный процесс и автоматизировать деятельность преподавателя по выдаче тестов и их проверки для оценивания знаний студентов, по средству реализации веб-приложения «Информационная система контроля освоения знаний и практических умений обучающихся (на примере ФГБОУ ВО РЭУ им. Г.В. Плеханова)» с возможностью управления курсами: создавать их, изменять, приглашать студентов, и материалами: загружать материалы по темам в курсы со студентами; проведением тестирования: создание тестирований, добавление в них вопросов, с возможностью указания вариантов ответа и выбором правильного для того, чтобы автоматизировать их проверку и подведения оценки знаний.

В результате выполнения курсовой работы, были получены навыки по работе с передачей файлов между клиентом и сервером, с возможностью реализации функции скачивания этого файла с веб-приложения на устройство клиента. Были получены навыки по реализации WEB-API с использованием фреймворка «ASP.NET Core».

В результате выполнения курсовой работы была разработана система онлайн-образования, которая закрывала проблематику, описанную в задаче курсовой работы, была достигнута поставленная цель по повышению навыков реализации веб-приложения с использованием фреймворка «Django», реализации API с использованием фреймворка «ASP.NET Core», реализации валидации навыков на стороне сервера базы данных.

## СПИСОК ИСПОЛЬЗУЕМЫХ МАТЕРИАЛОВ

1. Django введение - Изучение веб-разработки - MDN Web Docs, URL -  
[https://developer.mozilla.org/ru/docs/Learn\\_web\\_development/Extensions/Server-side/Django/Introduction](https://developer.mozilla.org/ru/docs/Learn_web_development/Extensions/Server-side/Django/Introduction) (Дата обращения: 23.09.2024)
2. Руководство по созданию веб-API с помощью ASP.NET Core, URL -  
<https://learn.microsoft.com/ru-ru/aspnet/core/tutorials/first-web-api?view=aspnetcore-9.0> (Дата обращения: 24.09.2024)
3. ASP.NET Core и C# | Полное руководство – Metanit, URL -  
<https://metanit.com/sharp/aspnet6/> (Дата обращения: 24.09.2024)
4. Руководство. Создание минимального API с помощью ASP.NET Core, URL - <https://learn.microsoft.com/ru-ru/aspnet/core/tutorials/min-web-api?view=aspnetcore-9.0> (Дата обращения: 24.09.2024)
5. Введение в Django: создание веб-приложений на Python – Skapro, URL - <https://sky.pro/wiki/python/vvedenie-v-django-sozdanie-veb-prilozhenij-na-python/> (Дата обращения: 23.09.2024)
6. Общие сведения об Entity Framework, URL -  
<https://learn.microsoft.com/ru-ru/dotnet/framework/data/adonet/ef/overview> (Дата обращения: 16.10.2024)
7. Официальная документация Django, URL -  
<https://docs.djangoproject.com/ru/stable/> (Дата обращения: 24.10.2024)
8. Документация по ASP.NET Core, URL - <https://learn.microsoft.com/ru-ru/aspnet/core/?view=aspnetcore-9.0> (Дата обращения: 24.10.2024)
9. Документация по Entity Framework, URL -  
<https://learn.microsoft.com/ru-ru/ef/> (Дата обращения: 16.10.2024)
10. HTML: HyperText Markup Language MDN, URL -  
<https://developer.mozilla.org/ru/docs/Web/HTML> (Дата обращения: 12.10.2024)

- 11.CSS: Каскадные таблицы стилей MDN, URL -  
<https://developer.mozilla.org/ru/docs/Web/CSS> (Дата обращения: 12.10.2024)
- 12.Интеграция Django с внешними API, URL -  
<https://sky.pro/wiki/python/integraciya-django-s-vneshnimi-api/> (Дата обращения: 20.11.2024)
- 13.Понимание архитектуры Django: паттерн MTV, URL -  
<https://dev.to/vincenttommi/2-understanding-djangos-architecture-the-mtv-pattern-1gl> (Дата обращения: 27.10.2024)
14. Архитектура в Django проектах, URL -  
[https://habr.com/ru/companies/vivid\\_money/articles/544856/](https://habr.com/ru/companies/vivid_money/articles/544856/) (Дата обращения: 27.10.2024)
- 15.Основы Django: Архитектура Model-View-Template (MVT), URL -  
<https://angelogentileiii.medium.com/basics-of-django-model-view-template-mvt-architecture-8585aecffbf6> (Дата обращения: 27.10.2024)
- 16.Архитектура REST, URL - <https://habr.com/ru/articles/38730/> (Дата обращения: 24.09.2024)
- 17.REST API: принципы, применение, URL - <https://gb.ru/blog/rest-api/> (Дата обращения: 25.09.2024)
- 18.Что такое RESTful API, URL - <https://aws.amazon.com/ru/what-is/restful-api/> (Дата обращения: 25.09.2024)
- 19.Что такое RESTful на самом деле, URL -  
<https://habr.com/ru/companies/hexlet/articles/274675/> (Дата обращения: 26.09.2024)
- 20.Профессиональная разработка технической документации – ГОСТ 34, URL - <https://www.swrit.ru/gost-34.html> (Дата обращения 25.11.2024)

## ПРИЛОЖЕНИЕ А. Техническое задание АННОТАЦИЯ

В данном программном документе приведено техническое задание для системы «Информационная система контроля освоения знаний и практических умений обучающихся (на примере ФГБОУ ВО РЭУ им. Г.В. Плеханова)»

ТЗ оформлено в соответствии с требованиями ГОСТ 19.106-78 и ГОСТ 19.104-78. Документ содержит основные разделы, включая введение, основания для разработки, назначение, требования к программе, требования к программной документации, технико-экономические показатели, стадии и этапы разработки, порядок контроля и приемки, а также приложения.

Техническое задание представляет собой основной документ, определяющий требования и цели проекта, устанавливая критерии успешной приемки работы. Разработка ТЗ является важным этапом в создании программного продукта, и его правильное оформление способствует четкому определению задачи и облегчает процесс разработки и контроля за выполнением проекта.

## СОДЕРЖАНИЕ

<b>ОБЩИЕ СВЕДЕНИЯ .....</b>	<b>6</b>
1. Полное наименование АС и ее условное обозначение .....	6
2. Шифр темы .....	6
3. Организация-заказчик и разработчик .....	6
4. Перечень документов, на основании которых создаётся АС .....	6
5. Плановые сроки начала и окончания работ по созданию АС .....	6
6. Источники и порядок финансирования работ .....	7
<b>ЦЕЛИ И НАЗНАЧЕНИЯ СОЗДАНИЯ АС .....</b>	<b>8</b>
1. Цели создания АС .....	8
2. Назначение АС .....	8
<b>ХАРАКТЕРИСТИКА ОБЪЕКТОВ АВТОМАТИЗАЦИИ.....</b>	<b>9</b>
1. Основные сведения об объекте автоматизации.....	9
2. Условия эксплуатации объекта автоматизации.....	9
<b>ТРЕБОВАНИЯ К АВТОМАТИЗИРОВАННОЙ СИСТЕМЕ .....</b>	<b>10</b>
1. Требования к структуре АС.....	10
1.1. Подсистемы: перечень, назначение и характеристики.....	10
1.1.1. Временной регламент реализации каждой функции .....	10
1.1.2. Требования к реализации каждой функции, к форме представления выходной информации, характеристики необходимой точности и времени выполнения, требования одновременности выполнения группы функций, достоверности выдачи результатов ..	11
1.1.3. Перечень и критерии отказов для каждой функции, по которой задаются требования по надежности .....	13
1.2. Взаимодействие компонентов АС .....	13
1.3. Совместимость со смежными системами .....	13

1.4. Режимы функционирования.....	14
1.5. Диагностика и модернизация.....	14
 2. Требования к видам обеспечения АС .....	14
2.1. Перечень требования к математическому виду обеспечения АС.....	14
2.2. Требования к информационному виду обеспечения АС, ....	14
2.3. Требования к лингвистическому виду обеспечения АС,.....	15
2.4. Требования к программному виду обеспечения АС .....	15
2.5. Требования к техническому виду обеспечения АС,.....	15
2.6. Требования к метрологическому виду обеспечения АС.....	15
2.7. Требования к организационному виду обеспечения АС,.....	15
2.8. Требования к методическому виду обеспечения АС .....	16
 3. Общие технические требования к АС .....	16
3.1. Требования к численности и квалификации персонала и пользователей АС .....	16
3.2. Требования к показателям назначения .....	16
3.3. Требования к надежности.....	16
3.4. Требования по безопасности.....	17
3.5. Требования к эргономике и технической эстетике.....	17
3.6. Требования к транспортабельности для подвижных АС .....	17
3.7. Требования к патентной чистоте и патентоспособности;.....	17
3.8. Требования по стандартизации и унификации .....	17
3.9. Дополнительные требования.....	17
 СОСТАВ И СОДЕРЖАНИЕ РАБОТ ПО СОЗДАНИЮ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ .....	18
1. Перечень этапов разработки.....	18
2. Сроки выполнения каждого этапа .....	18

# ПОРЯДОК РАЗРАБОТКИ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ

19

1. Организация разработки ..... 19

2. Исходные данные..... 19

3. Документы на каждом этапе..... 19

4. Проведение экспертизы документации ..... 19

5. Порядок утверждения и согласования..... 19

ПОРЯДОК КОНТРОЛЯ И ПРИЕМКИ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ; ..... 20

1. Виды испытаний ..... 20

2. Порядок приемки работ ..... 20

3. Состав приемочной комиссии ..... 20

ТРЕБОВАНИЯ К СОСТАВУ И СОДЕРЖАНИЮ РАБОТ ПО ПОДГОТОВКЕ ОБЪЕКТА АВТОМАТИЗАЦИИ К ВВОДУ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ В ДЕЙСТВИЕ; ..... 21

1. Условие функционирования объекта..... 21

2. Организационно-штатные мероприятия ..... 21

3. Обучения персонала ..... 21

ТРЕБОВАНИЯ К ДОКУМЕНТИРОВАНИЮ ..... 22

1. Перечень подлежащих разработке документов..... 22

2. Формат и количество документов..... 22

3. Использование ЕСКД и ЕСПД ..... 22

ИСТОЧНИКИ РАЗРАБОТКИ. В ТЗ НА АС МОГУТ БЫТЬ ВКЛЮЧЕНЫ ПРИЛОЖЕНИЯ..... 23

1. Технико-экономическое обоснование ..... 23

2. Научные исследования и отчеты.....	23
3. Аналоги систем .....	23
<b>ПРИЛОЖЕНИЯ .....</b>	<b>24</b>

## ОБЩИЕ СВЕДЕНИЯ

### 1. Полное наименование АС и ее условное обозначение

Полное наименование: «Информационная система контроля освоения знаний и практических умений обучающихся (на примере ФГБОУ ВО РЭУ им. Г.В. Плеханова)»

Условное обозначение: YumlSchool.

### 2. Шифр темы

Шифр темы: КП-МДК-02.01-ТРПО-П50-3-21

### 3. Организация-заказчик и разработчик

Организация-заказчик: ФГБОУ ВО РЭУ им. Г.В. Плеханова  
Московский приборостроительный техникум.

Организация-разработчик: Московский приборостроительный  
техникум

### 4. Перечень документов, на основании которых создаётся АС

- 1) Приказ директора Московского приборостроительного техникума №15/2025 от 10.01.2025.
- 2) ГОСТ 34.602-2020.
- 3) Внутренний регламент учебного заведения.

### 5. Плановые сроки начала и окончания работ по созданию АС.

- Описательная часть проекта (введение, общее описание) – 21 января – 24 января
- Анализ задачи и ее постановка 24 января – 31 января
- Проектирование и реализация – 31 января – 14 февраля
- Реализация в инструментальной среде 14 февраля – 28 февраля
- Технологическая часть проекта – 28 февраля – 14 марта
- Программная документация 14 марта – 28 марта
- Экспериментальная часть проекта 28 марта – 4 апреля

## **6. Источники и порядок финансирования работ**

Источники финансирования: внутренний бюджет Московского приборостроительного техникума.

Порядок финансирования: средства выделяются по приказу директора техникума.

## ЦЕЛИ И НАЗНАЧЕНИЯ СОЗДАНИЯ АС

### 1. Цели создания АС

- 1) Сократить время на проведение тестирования и проведения оценки знаний путем практических работ на 50% за счет автоматизации процессов с использованием алгоритмов оптимизации.
- 2) Обеспечить оперативный доступ к актуальной информации о лекциях, практических работах, тестах в режиме реального времени для всех участников учебного процесса.
- 3) Автоматизировать рутинные операции (проверка тестов у студентов) для снижения нагрузки на сотрудников учебного заведения.
- 4) Повысить эффективность взаимодействия между администраторами, преподавателями и студентами путем предоставления удобных инструментов для ведения учебного процесса.
- 5) Уменьшить количество ошибок при проверке умений обучающихся в ходе тестирования на 30% за счет интеллектуального анализа введенных данных и проверки на соответствие требованиям.

### 2. Назначение АС

- Автоматизировать управление учебным процессом, включая создание, редактирование и удаление данных о лекциях, практических работах, тестах.
- Оптимизировать процессы тестирования, проверки практических работ, ведения лекций, взаимодействия между студентами и преподавателями
- Формирование отчетности для анализа успеваемости практической

## **ХАРАКТЕРИСТИКА ОБЪЕКТОВ АВТОМАТИЗАЦИИ**

### **1. Основные сведения об объекте автоматизации**

Объект автоматизации:

- Московский приборостроительный техникум
- 30 учебных групп, более 600 студентов, 20 преподавателей

Основные задачи: сократить временные издержки, автоматизировать рутинные процессы

### **2. Условия эксплуатации объекта автоматизации**

Доступ через веб-интерфейс для удаленного управления.

Серверная часть предприятия расположена в административном корпусе техникума.

# ТРЕБОВАНИЯ К АВТОМАТИЗИРОВАННОЙ СИСТЕМЕ

## 1. Требования к структуре АС

### 1.1. Подсистемы: перечень, назначение и характеристики

Перечень подсистем:

Подсистема управления предметами: обеспечивает создание, редактирование и удаление данных о предметах, лекциях, практических работах, тестах внутри предметов. Подсистема также формирует отчетность по практическим работам.

Подсистема управления пользователями и ролями: отвечает за регистрацию, аутентификацию, разграничение прав доступа и настройку ролей (администратор, студент, преподаватель).

Подсистема автоматической проверки тестов. Подсистема автоматически сверяет ответы студента с правильными ответами в системе и проставляется оценка исходя из набранных баллов теста.

Подсистема формирования отчетов: генерирует отчеты об успеваемости студентов в формате PDF

Иерархия и степень централизации:  
Система построена на основе клиент-серверной архитектуры с централизованным управлением базой данных.

#### 1.1.1. Временной регламент реализации каждой функции

- Создание и редактирование данных о лекции, практической работе, teste: мгновенно после ввода данных пользователем.

- Автоматическая проверка теста студента: от 2 до 10 секунд в зависимости от количества вопросов в teste.

- Обновление статуса практической работы и получение оценки: мгновенно при внесении изменений.

- Обработка пользовательских запросов в системе: от 1 до 5 секунд в зависимости от сложности запроса.

1.1.2. Требования к реализации каждой функции, к форме представления выходной информации, характеристики необходимой точности и времени выполнения, требования одновременности выполнения группы функций, достоверности выдачи результатов

- Функции авторизации, регистрации:

- Форма представления: веб-интерфейс с полями ввода.
- Точность: корректность заполнения всех обязательных полей, проверка на соответствие типов данных и ограничений, валидация введенных значений.
- Время выполнения: мгновенно после ввода данных пользователем.
- Одновременность: возможна работа нескольких пользователей с разными данными.
- Достоверность: сохранение данных в базе данных с обеспечением целостности и непротиворечивости.

- Создание и редактирование данных о предметах, практических работах, тестах, лекциях:

- Форма представления: веб-интерфейс с полями ввода.
- Точность: корректность заполнения всех обязательных полей, проверка на соответствие типов данных и ограничений, валидация введенных значений.
- Время выполнения: мгновенно после ввода данных пользователем.
- Одновременность: возможна работа нескольких пользователей с разными данными.
- Достоверность: сохранение данных в базе данных с обеспечением целостности и непротиворечивости.

- Формирование отчетов и аналитики:

- Форма представления: веб-интерфейс с диаграммами на странице, PDF-файл с табличным представлением.
  - Точность: Корректное отображение всех данных, проверка на соответствие типов данных и ограничений, валидация введенных значений.
  - Время выполнения: мгновенное обновление отчетов и диаграмм после ввода данных пользователей
  - Одновременность: возможна работа нескольких пользователей с разными данными.
    - Достоверность: сохранение данных в базе данных с обеспечением целостности и непротиворечивости.
- Автоматическая проверка тестов:
- Форма представления: веб-интерфейс с полями вывода, ввода, выпадающими списками.
  - Точность: автоматическая сверка ответов студента с правильными ответами, обеспечивающая корректное выставление оценок на основе набранных баллов. Проверка на соответствие формата ответов и валидация введенных значений.
  - Время выполнения: мгновенное обновление после завершения теста.
  - Одновременность: возможна работа нескольких пользователей с разными данными.
  - Достоверность: сохранение данных в базе данных с обеспечением целостности и непротиворечивости.

1.1.3. Перечень и критерии отказов для каждой функции, по которой задаются требования по надежности

- Создание, редактирование и удаление данных о предметах, лекциях, практических работ, тестах внутри предметов: отказ возможен при недоступности базы данных, нарушении целостности данных или попытке ввода некорректных значений. Критерий: успешное сохранение данных в базе данных и отображение сообщения об ошибке в случае отказа.

- Генерация отчетов и аналитики: отказ возможен при недостаточном количестве данных, противоречивых ограничениях или отсутствии доступных ресурсов (студентов, оценок). Критерий: успешная генерация файла или сообщение об ошибке с указанием причины отказа.

- Автоматическая проверка тестов.: отказ возможен при недоступности базы данных, нарушении целостности данных или попытке ввода некорректных значений. Критерий: Успешное сохранение данных в базе данных. Отображение сообщения об ошибке в случае отказа с указанием причины.

- Обработка пользовательских запросов в системе: отказ возможен при некорректном запросе, отсутствии необходимой информации в базе данных или перегрузке сервера. Критерий: выдача релевантного ответа или сообщение об ошибке с указанием причины отказа.

## 1.2. Взаимодействие компонентов АС

Компоненты системы взаимодействуют через API.ASP, обеспечивая обмен данными между клиентской частью, сервером и базой данных.

Безопасность взаимодействия обеспечивается через HTTPS, аутентификацию пользователей и защиту от несанкционированного доступа посредством шифрования base64 и хешированием bcrypt.

## 1.3. Совместимость со смежными системами

Система поддерживает взаимодействие с базами данных MS SQL для хранения данных.

## **1.4. Режимы функционирования**

Система должна поддерживать круглосуточный режим работы с возможностью обработки запросов в реальном времени. Допускается кратковременное отключение для технического обслуживания, которое должно выполняться в ночное время или в периоды наименьшей нагрузки.

АС должна функционировать в штатном режиме при нормальной загрузке серверов и обеспечивать корректное масштабирование при увеличении числа пользователей.

## **1.5. Диагностика и модернизация**

В дальнейшем система может быть доработана за счет расширения функционала автоматизированных инструментов, включая широкие методы по выдаче практических работ и их проверки, усовершенствования тестирования.

Возможна интеграция с другими сервисами и корпоративными системами управления техникума (например, с системой электронного документооборота) для автоматизации бизнес-процессов в образовательном учреждении.

## **2. Требования к видам обеспечения АС**

### **2.1. Перечень требований к математическому виду обеспечения АС**

Математическое обеспечение:

- Алгоритм вычисления автоматической проверки тестирования
- Алгоритм вычисления составления отчета об успеваемости студента
- Алгоритм вычисления попыток прохождения тестирования
- Алгоритм вычисления попыток прохождения практической работы

### **2.2. Требования к информационному виду обеспечения АС,**

Требования к информационному виду обеспечения АС включают создание интуитивно понятного интерфейса для пользователей с

различными ролями (администратор, преподаватель, студент), обеспечивающего простоту взаимодействия с системой. Интерфейс должен быть адаптивным. Для отображения данных должны использоваться поля данных, диаграммы и списки, которые облегчают восприятие информации.

### 2.3. Требования к лингвистическому виду обеспечения АС,

Лингвистическое обеспечение: язык интерфейса — русский, язык базы данных — SQL, язык фреймворка Django – Python, язык API.ASP – C#.

### 2.4. Требования к программному виду обеспечения АС

- Серверная часть: Python (Django).
- Клиентская часть: API.ASP.
- База данных: MSSQL.
- Требования: высокая производительность, модульность, возможность обновления.

### 2.5. Требования к техническому виду обеспечения АС,

- Сервер: 8 ГБ ОЗУ, 4 ядра процессора, 500 ГБ SSD.
- Клиентские устройства: любой ПК с браузером (Chrome, Firefox).

### 2.6. Требования к метрологическому виду обеспечения АС

- Контроль точности расчётов успеваемости студента в выгруженном файле PDF.
- Использование тестовых данных для проверки корректности.

### 2.7. Требования к организационному виду обеспечения АС,

- Администратор системы отвечает за настройку пользователей, предметов, групп
- Преподаватель отвечает за данные лекций, практических работ, тестов.
- Студенту доступен просмотр лекций, практических работ, прохождения тестирования

## 2.8. Требования к методическому виду обеспечения АС

- Руководство пользователя: описание интерфейса и функций системы.
- Руководство администратора: настройка и управление системой.

## 3. Общие технические требования к АС

### 3.1. Требования к численности и квалификации персонала и пользователей АС

Численность и квалификация:

- Администратор: 1 человек, знание основ администрирования баз данных.
- Преподаватели: 20 человек, базовые навыки работы с веб-интерфейсами.
- Студенты: 600 человек, базовые навыки работы с веб-интерфейсами.

### 3.2. Требования к показателям назначения

Система должна обеспечивать:

- 1) Высокую производительность и минимальные задержки.
- 2) Надежность с минимальными сбоями и восстановлением данных.
- 3) Доступность 24/7.
- 4) Защиту данных и безопасность от несанкционированного доступа.
- 5) Масштабируемость для увеличения числа пользователей и функционала.

### 3.3. Требования к надежности

Система должна обеспечивать:

- 1) Минимальный процент сбоев в работе (не более 1% в месяц).
- 2) Автоматическое восстановление после сбоя с минимальными потерями данных.
- 3) Высокую отказоустойчивость и возможность продолжения работы

в случае частичных сбоев.

### 3.4. Требования по безопасности

Система должна обеспечивать:

1. Шифрование данных при передаче и хранении.
2. Аутентификацию для пользователей с разными ролями через логин и пароль.
3. Ограничение доступа к данным и функционалу в зависимости от прав пользователя.

### 3.5. Требования к эргономике и технической эстетике

Система должна обеспечивать:

- 1) Интуитивно понятный и удобный интерфейс с минимальным количеством действий для выполнения основной функциональности.
- 2) Адаптивный дизайн, обеспечивающий комфортную работу на различных устройствах (ПК, планшетах, мобильных устройствах).
- 3) Чистый, современный стиль с использованием корпоративных цветов и шрифтов, создающий приятное визуальное восприятие.

### 3.6. Требования к транспортабельности для подвижных АС

Не применимо.

### 3.7. Требования к патентной чистоте и патентоспособности;

Система должна быть разработана с учетом патентной чистоты, чтобы избежать нарушений интеллектуальной собственности.

### 3.8. Требования по стандартизации и унификации

Система должна соответствовать международным и отраслевым стандартам в области разработки программного обеспечения и безопасности данных.

### 3.9. Дополнительные требования

Система должна обеспечивать возможность масштабирования для поддержки увеличения числа пользователей и объёмов данных.

## СОСТАВ И СОДЕРЖАНИЕ РАБОТ ПО СОЗДАНИЮ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ

### 1. Перечень этапов разработки

- Описательная часть проекта (введение, общее описание)
- Анализ задачи и ее постановка
- Проектирование и реализация
- Реализация в инструментальной среде
- Технологическая часть проекта
- Программная документация
- Экспериментальная часть проекта

### 2. Сроки выполнения каждого этапа

- Описательная часть проекта (введение, общее описание) – 21 января – 24 января
- Анализ задачи и ее постановка 24 января – 31 января
- Проектирование и реализация – 31 января – 14 февраля
- Реализация в инструментальной среде 14 февраля – 28 февраля
- Технологическая часть проекта – 28 февраля – 14 марта
- Программная документация 14 марта – 28 марта
- Экспериментальная часть проекта 28 марта – 4 апреля

# ПОРЯДОК РАЗРАБОТКИ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ

## 1. Организация разработки

Участник проекта: студент-практикант в ФГБОУ ВО РЭУ им. Г.В. Плеханова

## 2. Исходные данные

- ГОСТ 34.602-2020 "Автоматизированные системы. Общие положения".
- Информация о текущих системах и программных решениях, используемых для ведения учебной деятельности в ИС

## 3. Документы на каждом этапе

- Постановка задачи: техническое задание.
- Проектирование: эскизный проект, технический проект.
- Разработка: спецификация программы, текст программы.
- Тестирование: программа и методика испытаний, протокол испытаний.
- Внедрение: руководство пользователя.

## 4. Проведение экспертизы документации

- Экспертиза документации:
- Экспертиза проводится руководителем КП.
- Проверяются соответствие ТЗ требованиям ГОСТ, полнота проектной документации.

## 5. Порядок утверждения и согласования

- Техническое задание утверждается директором ФГБОУ ВО РЭУ им. Г.В. Плеханова".
- Проектная документация согласовывается с руководителями КП
- Программы тестирования и внедрения системы утверждаются руководителем КП

# ПОРЯДОК КОНТРОЛЯ И ПРИЕМКИ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ;

## 1. Виды испытаний

- Предварительные испытания: проверка функциональности системы на уровне разработчика.
- Приёмочные испытания: проверка системы заказчиком на соответствие ТЗ.
- Сертификационные испытания: проверка на соответствие стандартам безопасности и производительности.

## 2. Порядок приемки работ

- 1) Проведение приёмочных испытаний.
- 2) Подготовка отчёта о результатах испытаний.
- 3) Подписание акта приёмки-передачи системы.

## 3. Состав приемочной комиссии

- Заказчик – ФГБОУ ВО РЭУ им. Г.В. Плеханова
- Руководитель КП

**ТРЕБОВАНИЯ К СОСТАВУ И СОДЕРЖАНИЮ РАБОТ ПО  
ПОДГОТОВКЕ ОБЪЕКТА АВТОМАТИЗАЦИИ К ВВОДУ  
АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ В ДЕЙСТВИЕ;**

**1. Условие функционирования объекта**

- Подготовка серверной комнаты с системой охлаждения и бесперебойным питанием.
- Обеспечение стабильного интернет-соединения с пропускной способностью не менее 100 Мбит/с.

**2. Организационно-штатные мероприятия**

- Назначение администратора системы из числа IT-персонала компании.
- Проведение инструктажа для сотрудников компании.

**3. Обучения персонала**

Проведение двухдневного тренинга для сотрудников.

Разработка инструкции пользователя в формате PDF.

## **ТРЕБОВАНИЯ К ДОКУМЕНТИРОВАНИЮ**

### **1. Перечень подлежащих разработке документов**

Перечень документов:

- Бланк задания
- Титульный лист КП
- Пояснительная записка
- Приложения А. Техническое задание
- Приложение Б. Текст программы
- Приложение В. Скрипт БД
- Приложение Г. Программа испытаний
- Приложение Д. Руководство пользователя

### **2. Формат и количество документов**

Электронный формат. 8 документов

### **3. Использование ЕСКД и ЕСПД**

Использование стандартов:

Разработка документов в соответствии с ГОСТ 19.106-78 (ЕСПД).

Применение графических схем согласно ГОСТ 2.701-84 (ЕСКД).

## ИСТОЧНИКИ РАЗРАБОТКИ. В ТЗ НА АС МОГУТ БЫТЬ ВКЛЮЧЕНЫ ПРИЛОЖЕНИЯ.

### 1. Технико-экономическое обоснование

- Ожидаемое снижение затрат на 40% за счёт автоматизации процессов управления.
- Прогнозируемое сокращение времени на проверку тестирования на 90%, что повысит общую эффективность работы сотрудников учебной части.

### 2. Научные исследования и отчеты

Научные исследования:

Исследования по внедрению автоматизированных систем в корпоративных структурах для повышения эффективности.

### 3. Аналоги систем

Аналоги систем:

Система «Classroom» от компании Google

Система «SkillBox» от холдинга Skillbox Holding Limited

## **ПРИЛОЖЕНИЯ**

Приложения:

- Диаграмма классов
- Схема базы данных
- IDEF0 (TO BE)
- Структурная схема
- Схема пользовательского интерфейса
- Схема тестирования

**ПРИЛОЖЕНИЕ Б. Текст программы**  
**АННОТАЦИЯ**

В данном программном документе «Текст программы» приведен исходный текст программного изделия.

В документе представлен раздел «Текст программы» с подразделами:

- «Наименование программы»
- «Область применения программы»
- «Модули»

В подразделе «Наименование программы» содержится наименование самой программы.

В подразделе «Область применения» содержится описание области, в которой применяется программа.

В подразделе «Модули» содержится перечисление всех фалов программы с детальной информацией.

## СОДЕРЖАНИЕ

Текст программы .....	3
1.1. Наименование объекта .....	3
1.2. Область применения объекта .....	3
1.3. Модули.....	3
1.4. Код программы .....	7

## ТЕКСТ ПРОГРАММЫ

### 1.1. Наименование объекта

Наименование - «Веб приложение YumlSchool».

### 1.2. Область применения объекта

Приложение предназначено к использованию в учебных заведениях. Веб-приложение позволит оптимизировать учебный процесс и автоматизировать проведение практических работ и тестирования студентов по изученным темам.

### 1.3. Модули

Таблица 1. Модули

№	Название	Описание	Количество строк	Размер файла в Кб
1	2	3	4	5
1	YumlsschooldbContext.cs	Класс для всех моделей базы данных	320	13 КБ
2	UserController.cs	Класс для работы с CRUD операциями API сервиса модели User	171	6 КБ
3	TestsController.cs	Класс для работы с CRUD операциями API сервиса модели Test	131	4 КБ
4	TestAttemptsController.cs	Класс для работы с CRUD операциями API сервиса модели TestAttempt	108	3 КБ
5	SubjectsController.cs	Класс для работы с CRUD операциями API сервиса модели Subject	108	3 КБ
6	StatusController.cs	Класс для работы с CRUD операциями API сервиса модели Status	108	3 КБ
7	RolesController.cs	Класс для работы с CRUD	108	3 КБ

		операциями API сервиса модели Role		
8	PracticalWorksController.cs	Класс для работы с CRUD операциями API сервиса модели PracticalWork	108	3 КБ
9	QuestionTestsController.cs	Класс для работы с CRUD операциями API сервиса модели QuestionTest	124	4 КБ
10	LectonMaterialsController.cs	Класс для работы с CRUD операциями API сервиса модели LectionMaterial	108	4 КБ
11	GroupsController.cs	Класс для работы с CRUD операциями API сервиса модели Group	108	3 КБ
12	GradingCriterionsController.cs	Класс для работы с CRUD операциями API сервиса модели GradingCriterion	108	4 КБ
13	CompleteWorksController.cs	Класс для работы с CRUD операциями API сервиса модели CompleteWorks	108	3 КБ
14	AdminPanel.css	Файл формата css для стилей административной панели	254	6 КБ
15	AllSubjectPage.css	Файл формата css для стилей страницы предметов	676	14 КБ
16	base.css	Файл формата css для базовых стилей страниц	635	13 КБ
17	CreateTest.css	Файл формата css для стилей создания теста	812	20 КБ
18	GroupCRUD.css	Файл формата css для стилей действий над группами	357	8 КБ
19	LectionCRUD.css	Файл формата css для стилей действий над лекциями	237	6 КБ
20	LectionMaterial.css	Файл формата css для стилей лекции	335	8 КБ
21	Logreg.css	Файл формата css для стилей регистрации и авторизации	364	8 КБ

22	PracticalWork.css	Файл формата css для стилей практической работы	747	18 КБ
23	SubjectCRUD.css	Файл формата css для стилей действий над предметом	267	7 КБ
24	SubjectPage.css	Файл формата css для стилей страницы предмета	405	10 КБ
25	TestAttempts.css	Файл формата css для стилей страницы вывода о попытках прохождения теста	278	5 КБ
26	TestEditor.css	Файл формата css для стилей изменения теста	121	3 КБ
27	TestMaterial.css	Файл формата css для стилей теста	583	13 КБ
28	TestProcess.css	Файл формата css для стилей теста в процессе прохождения	714	18 КБ
29	TestResult.css	Файл формата css для стилей результата теста для студента	525	13 КБ
30	TestStudentResults.css	Файл формата css для стилей результата теста для преподавателя	551	12 КБ
31	TestStyles.css	Файл формата css для стилей теста	217	5 КБ
32	UserCRUD.css	Файл формата css для стилей действий над пользователем	348	8 КБ
33	WorkMaterial.css	Файл формата css для стилей выполненной работы	671	17 КБ
34	GroupCRUD.html	Страница действий над группой	63	3 КБ
35	SubjectCRUD.html	Страница действий над предметом	86	4 КБ
36	SubjectEdit.html	Страница изменения над предметом	53	3 КБ

37	UserCRUD.html	Страница действий над пользователем	106	5 КБ
38	AddCompleteWork.html	Страница добавления выполненной работы	542	19 КБ
39	CompletedWork.html	Страница выполненной работы	53	3 КБ
40	CreateTest.html	Страница создания теста	381	19 КБ
41	CRUDPracticalWork.html	Страница действий над практической работой	54	3 КБ
42	EditTest.html	Страница изменения теста	445	24 КБ
43	LectureCRUD.html	Страница действий над лекцией	54	3 КБ
44	LectureEdit.html	Страница изменения лекции	75	4 КБ
45	LectureMaterial.html	Страница лекции	55	3 КБ
46	PracticalWorkEdit.html	Страница изменения практической работы	41	2 КБ
47	PracticalWorkMaterial.html	Страница практической работы	497	31 КБ
48	SuccessMessage.html	Страница успешного сообщения о прикреплении работы	22	1 КБ
49	TestAttempts.html	Страница информации о попытках теста	117	6 КБ
50	TestMaterial.html	Страница теста	643	23 КБ
51	TestProcess.html	Страница прохождения теста	290	18 КБ
52	TestResults.html	Страница результата теста	136	7 КБ
53	TestStart.html	Страница начала теста	66	4 КБ
54	TestStudentsResults.html	Страница результатов тестирования студентов	115	6 КБ
55	AdminPanel.html	Страница административной панели	81	4 КБ
56	AllSubjectPage.html	Страница предметов	103	5 КБ

57	header.html	Заголовок страниц	234	7 КБ
58	login.html	Страница авторизации	63	3 КБ
59	NullSubjects.html	Страница для незарегистрированного в системе пользователя	160	6 КБ
60	registration.html	Страница регистрации	93	4 КБ
61	SubjectPage.html	Страница предмета	148	6 КБ
62	Views.py	Файл методов для страниц	2873	135 КБ

#### 1.4. Код программы

##### YumlsschooldbContext.cs

```

using System;
using System.Collections.Generic;
using Microsoft.EntityFrameworkCore;

namespace YumlsSchoolAPI.Models;

public partial class YumlsschooldbContext : DbContext
{
    public YumlsschooldbContext()
    {
    }

    public YumlsschooldbContext(DbContextOptions<YumlsschooldbContext> options)
        : base(options)
    {
    }

    public virtual DbSet<CompletedWork> CompletedWorks { get; set; }

    public virtual DbSet<GradingCriterion> GradingCriteria { get; set; }

    public virtual DbSet<Group> Groups { get; set; }

    public virtual DbSet<LectionMaterial> LectionMaterials { get; set; }

    public virtual DbSet<PracticalWork> PracticalWorks { get; set; }

```

```
public virtual DbSet<QuestionsTest> QuestionsTests { get; set; }
```

```
public virtual DbSet<Role> Roles { get; set; }
```

```
public virtual DbSet<Status> Statuses { get; set; }
```

```
public virtual DbSet<Subject> Subjects { get; set; }
```

```
public virtual DbSet<Test> Tests { get; set; }
```

```
public virtual DbSet<TestAttempt> TestAttempts { get; set; }
```

```
public virtual DbSet<User> Users { get; set; }
```

```
protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
```

```
#warning To protect potentially sensitive information in your connection string, you should move it out of source code. You can avoid scaffolding the connection string by using the Name= syntax to read it from configuration - see https://go.microsoft.com/fwlink/?linkid=2131148. For more guidance on storing connection strings, see https://go.microsoft.com/fwlink/?LinkId=723263.
```

```
=> optionsBuilder.UseSqlServer("Data Source=LAPTOP-T6NC0EHD\\MSSQLSERVER01;Initial Catalog=yumlsschooldb;Integrated Security=True;Encrypt=True;Trust Server Certificate=True");
```

```
protected override void OnModelCreating(ModelBuilder modelBuilder)
```

```
{
```

```
    modelBuilder.Entity<CompletedWork>(entity =>
```

```
{
```

```
        entity.HasKey(e => e.IdCompletedWork).HasName("PK__Complete__CAC448AD1752761D");
```

```
        entity.ToTable("CompletedWork");
```

```
        entity.Property(e => e.IdCompletedWork).HasColumnName("ID_CompletedWork");
```

```
        entity.Property(e =>
```

```
=>
```

```
e.DateUploadCompletedWork).HasColumnName("DateUpload_CompletedWork");
```

```
        entity.Property(e => e.Grade)
```

```
            .HasMaxLength(64)
```

```
            .IsUnicode(false);
```

```
        entity.Property(e => e.NameFileCompletedWork)
```

```
            .HasMaxLength(64)
```

```
            .IsUnicode(false)
```

```
            .HasColumnName("Name_FileCompletedWork");
```

```
        entity.Property(e => e.PracticalWorkId).HasColumnName("PracticalWork_ID");
```

```
        entity.Property(e => e.StatusId).HasColumnName("Status_ID");
```

```

entity.Property(e => e.TimeUploadCompletedWork)
    .HasDefaultValueSql("(CONVERT([time],getdate()))")
    .HasColumnName("TimeUpload_CompletedWork");
entity.Property(e => e.UrlFileCompletedWork)
    .HasMaxLength(3000)
    .IsUnicode(false)
    .HasColumnName("URL_FileCompletedWork");
entity.Property(e => e.UserId).HasColumnName("User_ID");

});

modelBuilder.Entity<GradingCriterion>(entity =>
{
    entity.HasKey(e => e.IdCriteria).HasName("PK__GradingC__6D5D9FD929FAF82A");

    entity.Property(e => e.IdCriteria).HasColumnName("ID_Criteria");
    entity.Property(e => e.IdTest).HasColumnName("ID_Test");
    entity.Property(e => e.MaxPoints).HasColumnName("Max_Points");
    entity.Property(e => e.MinPoints).HasColumnName("Min_Points");

});

modelBuilder.Entity<Group>(entity =>
{
    entity.HasKey(e => e.IdGroup).HasName("PK__Group__96125DD8511AD3F0");

    entity.ToTable("Group");

    entity.Property(e => e.IdGroup).HasColumnName("ID_Group");
    entity.Property(e => e.NameGroup)
        .HasMaxLength(50)
        .IsUnicode(false)
        .HasColumnName("Name_Group");
});

modelBuilder.Entity<LectionMaterial>(entity =>
{
    entity.HasKey(e => e.IdLectionMaterial).HasName("PK__LectionM__2A1A1672EFF687DE");

    entity.ToTable("LectionMaterial");
}

```

```

entity.Property(e => e.IdLectionMaterial).HasColumnName("ID_LectionMaterial");
entity.Property(e => e.DateUploadLectionMaterial)
    .HasDefaultValueSql("(CONVERT([date],getdate()))")
    .HasColumnName("DateUpload_LectionMaterial");
entity.Property(e => e.DescriptionLectionMaterial)
    .HasMaxLength(1024)
    .IsUnicode(false)
    .HasColumnName("Description_LectionMaterial");
entity.Property(e => e.NameFileLection)
    .HasMaxLength(64)
    .IsUnicode(false)
    .HasColumnName("Name_FileLection");
entity.Property(e => e.NameLectionMaterial)
    .HasMaxLength(64)
    .IsUnicode(false)
    .HasColumnName("Name_LectionMaterial");
entity.Property(e => e.SubjectId).HasColumnName("Subject_ID");
entity.Property(e => e.TimeUploadLectionMaterial)
    .HasDefaultValueSql("(CONVERT([time],getdate()))")
    .HasColumnName("TimeUpload_LectionMaterial");
entity.Property(e => e.UrlFileLection)
    .HasMaxLength(1024)
    .IsUnicode(false)
    .HasColumnName("URL_FileLection");
entity.Property(e => e.UserId).HasColumnName("User_ID");

});

modelBuilder.Entity<PracticalWork>(entity =>
{
    entity.HasKey(e => e.IdPracticalWork).HasName("PK__Practica__67699CC8E4B2BDDC");
    entity.ToTable("PracticalWork");

    entity.Property(e => e.IdPracticalWork).HasColumnName("ID_PracticalWork");
    entity.Property(e => e.DateUploadPracticalWork)
        .HasDefaultValueSql("(CONVERT([date],getdate()))")
        .HasColumnName("DateUpload_PracticalWork");
    entity.Property(e => e.DescriptionPracticalWork)
        .HasMaxLength(3000)

```

```

        .IsUnicode(false)
        .HasColumnName("Description_PracticalWork");
entity.Property(e => e.NameFilePracticalWork)
        .HasMaxLength(64)
        .IsUnicode(false)
        .HasColumnName("Name_FilePracticalWork");
entity.Property(e => e.NamePracticalWork)
        .HasMaxLength(64)
        .IsUnicode(false)
        .HasColumnName("Name_PracticalWork");
entity.Property(e => e.SubjectId).HasColumnName("Subject_ID");
entity.Property(e => e.TimeUploadPracticalWork)
        .HasDefaultValueSql("(CONVERT([time],getdate()))")
        .HasColumnName("TimeUpload_PracticalWork");
entity.Property(e => e.UrlFilePracticalWork)
        .HasMaxLength(3000)
        .IsUnicode(false)
        .HasColumnName("URL_FilePracticalWork");
entity.Property(e => e.UserId).HasColumnName("User_ID");

```

});

```

modelBuilder.Entity<QuestionsTest>(entity =>
{
    entity.HasKey(e => e.IdQuestion).HasName("PK__Question__7F5FD854C1D71974");

    entity.Property(e => e.IdQuestion).HasColumnName("ID_Question");
    entity.Property(e => e.AnswerVariants).IsUnicode(false);
    entity.Property(e => e.CorrectAnswer).IsUnicode(false);
    entity.Property(e => e.IdTest).HasColumnName("ID_Test");
    entity.Property(e => e.Points).HasDefaultValue(1);
    entity.Property(e => e.QuestionText)
        .HasMaxLength(1000)
        .IsUnicode(false)
        .HasColumnName("Question_Text");

```

});

```

modelBuilder.Entity<Role>(entity =>
{

```

```

entity.HasKey(e => e.IdRole).HasName("PK__Role__43DCD32DB47A97A2");

entity.ToTable("Role");

entity.Property(e => e.IdRole).HasColumnName("ID_Role");
entity.Property(e => e.RoleName)
    .HasMaxLength(30)
    .IsUnicode(false)
    .HasColumnName("Role_Name");
});

modelBuilder.Entity<Status>(entity =>
{
    entity.HasKey(e => e.IdStatus).HasName("PK__Status__5AC2A734635C9330");

    entity.ToTable("Status");

    entity.Property(e => e.IdStatus).HasColumnName("ID_Status");
    entity.Property(e => e.NameStatus)
        .HasMaxLength(64)
        .IsUnicode(false)
        .HasColumnName("Name_Status");
});

modelBuilder.Entity<Subject>(entity =>
{
    entity.HasKey(e => e.IdSubject).HasName("PK__Subject__20028FF458068C53");

    entity.ToTable("Subject");

    entity.Property(e => e.IdSubject).HasColumnName("ID_Subject");
    entity.Property(e => e.IdGroup).HasColumnName("ID_Group");
    entity.Property(e => e.NameSubject)
        .HasMaxLength(64)
        .IsUnicode(false)
        .HasColumnName("Name_Subject");
    entity.Property(e => e.UserId).HasColumnName("User_ID");
});

modelBuilder.Entity<Test>(entity =>

```

```

{
    entity.HasKey(e => e.IdTest).HasName("PK__Tests__D6560E255434BEEA");

    entity.Property(e => e.IdTest).HasColumnName("ID_Test");
    entity.Property(e => e.DateUploadTest)
        .HasDefaultValueSql("(CONVERT([date],getdate()))")
        .HasColumnName("DateUpload_Test");
    entity.Property(e => e.DescriptionTest)
        .HasMaxLength(1000)
        .IsUnicode(false)
        .HasColumnName("Description_Test");
    entity.Property(e => e.NameTest)
        .HasMaxLength(255)
        .IsUnicode(false)
        .HasColumnName("Name_Test");
    entity.Property(e => e.SubjectId).HasColumnName("Subject_ID");
    entity.Property(e => e.TimeUploadTest)
        .HasDefaultValueSql("(CONVERT([time],getdate()))")
        .HasColumnName("TimeUpload_Test");
    entity.Property(e => e.UserId).HasColumnName("User_ID");
}

modelBuilder.Entity<TestAttempt>(entity =>
{
    entity.HasKey(e => e.IdTestAttempt).HasName("PK__TestAtte__34F010C98BED223F");

    entity.ToTable("TestAttempt");

    entity.Property(e => e.IdTestAttempt).HasColumnName("ID_TestAttempt");
    entity.Property(e => e.EndTime)
        .HasColumnType("datetime")
        .HasColumnName("End_Time");
    entity.Property(e => e.Grade).HasColumnType("decimal(3, 1)");
    entity.Property(e => e.StartTime)
        .HasDefaultValueSql("(getdate())")
        .HasColumnType("datetime")
        .HasColumnName("Start_Time");
    entity.Property(e => e.Status)
        .HasMaxLength(20)
        .IsUnicode(false);
}
);

```

```

entity.Property(e => e.TestId).HasColumnName("Test_ID");
entity.Property(e => e.UserId).HasColumnName("User_ID");

});

modelBuilder.Entity<User>(entity =>
{
    entity.HasKey(e => e.IdUser).HasName("PK__User__ED4DE442F97E89DC");

    entity.ToTable("User");

    entity.HasIndex(e => e.Email, "UQ__User__A9D105342E3385EC").IsUnique();

    entity.Property(e => e.IdUser).HasColumnName("ID_User");
    entity.Property(e => e.Email)
        .HasMaxLength(255)
        .IsUnicode(false);
    entity.Property(e => e.FirstName)
        .HasMaxLength(64)
        .IsUnicode(false);
    entity.Property(e => e.IdGroup).HasColumnName("ID_Group");
    entity.Property(e => e.IdRole).HasColumnName("ID_Role");
    entity.Property(e => e.Login)
        .HasMaxLength(150)
        .IsUnicode(false);
    entity.Property(e => e.MiddleName)
        .HasMaxLength(64)
        .IsUnicode(false)
        .HasDefaultValue("Онлайн");
    entity.Property(e => e.Password)
        .HasMaxLength(150)
        .IsUnicode(false);
    entity.Property(e => e.SecondName)
        .HasMaxLength(64)
        .IsUnicode(false);

});

OnModelCreatingPartial(modelBuilder);
}

```

```
        partial void OnModelCreatingPartial(ModelBuilder modelBuilder);  
    }  
}
```

## UserController.cs

```
using System;  
using System.Collections.Generic;  
using System.Linq;  
using System.Threading.Tasks;  
using Microsoft.AspNetCore.Http;  
using Microsoft.AspNetCore.Mvc;  
using Microsoft.EntityFrameworkCore;  
using YumlsSchoolAPI.Models;  
  
namespace YumlsSchoolAPI.Models  
{  
    [Route("api/[controller]")]  
    [ApiController]  
    public class UsersController : ControllerBase  
    {  
        private readonly YumlsschooldbContext _context;  
  
        public UsersController(YumlsschooldbContext context)  
        {  
            _context = context;  
        }  
  
        // GET: api/Users  
        [HttpGet]  
        public async Task<ActionResult<IEnumerable<User>>> GetUsers()  
        {  
            return await _context.Users.ToListAsync();  
        }  
  
        // GET: api/Users/5  
        [HttpGet("{id}")]  
        public async Task<ActionResult<User>> GetUser(int? id)  
        {  
            var user = await _context.Users.FindAsync(id);  
  
            if (user == null)  
            {  
                return NotFound();  
            }  
        }  
}
```

```

    }

    return user;
}

// PUT: api/Users/5
// To protect from overposting attacks, see https://go.microsoft.com/fwlink/?LinkId=2123754
[HttpPut("{id}")]
public async Task<IActionResult> PutUser(int? id, User user)
{
    if (id != user.IdUser)
    {
        return BadRequest();
    }

    // Проверяем, существует ли пользователь
    var existingUser = await _context.Users.FindAsync(id);
    if (existingUser == null)
    {
        return NotFound();
    }

    // Обновляем поля существующего пользователя
    existingUser.FirstName = user.FirstName;
    existingUser.SecondName = user.SecondName;
    existingUser.MiddleName = user.MiddleName;
    existingUser.Email = user.Email;
    existingUser.IdRole = user.IdRole;
    existingUser.IdGroup = user.IdGroup;

    _context.Entry(existingUser).State = EntityState.Modified;

    try
    {
        await _context.SaveChangesAsync();
    }
    catch (DbUpdateConcurrencyException)
    {
        if (!UserExists(id))
        {
            return NotFound();
        }
    }
}

```

```

        else
        {
            throw;
        }
    }

    return NoContent();
}

// POST: api/Users
// To protect from overposting attacks, see https://go.microsoft.com/fwlink/?LinkId=2123754
[HttpPost]
public async Task<ActionResult<User>> PostUser(User user)
{
    _context.Users.Add(user);
    await _context.SaveChangesAsync();

    return CreatedAtAction(" GetUser", new { id = user.IdUser }, user);
}

// DELETE: api/Users/5
[HttpDelete("{id}")]
public async Task<IActionResult> DeleteUser(int? id)
{
    var user = await _context.Users.FindAsync(id);
    if (user == null)
    {
        return NotFound();
    }

    _context.Users.Remove(user);
    await _context.SaveChangesAsync();

    return NoContent();
}

private bool UserExists(int? id)
{
    return _context.Users.Any(e => e.IdUser == id);
}

```

```

// POST: api/Users/Authenticate
[HttpPost("Authenticate")]
public async Task<ActionResult<User>> Authenticate([FromBody] User user)
{
    // Проверка на наличие логина и пароля
    if (string.IsNullOrEmpty(user.Login) || string.IsNullOrEmpty(user.Password))
    {
        return BadRequest("Логин и пароль обязательны.");
    }

    // Поиск пользователя по логину и паролю
    var foundUser = await _context.Users
        .FirstOrDefaultAsync(u => u.Login == user.Login && u.Password == user.Password);

    if (foundUser == null)
    {
        return Unauthorized("Неверный логин или пароль.");
    }

    // Возвращаем информацию о пользователе
    return Ok(new
    {
        idUser = foundUser.IdUser,
        idRole = foundUser.IdRole,
        firstName = foundUser.FirstName,
        secondName = foundUser.SecondName,
        middleName = foundUser.MiddleName
    });
}

// GET: api/Users/filter
[HttpGet("filter")]
public async Task<ActionResult<IEnumerable<User>>> GetUsersByGroupId(int groupId)
{
    var students = await _context.Users
        .Where(u => u.IdGroup == groupId)
        .ToListAsync();
}

```

```

        return Ok(students);
    }

}

```

## TestsController.cs

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using YumlsSchoolAPI.Models;

namespace YumlsSchoolAPI.Models
{
    [Route("api/[controller]")]
    [ApiController]
    public class TestsController : ControllerBase
    {
        private readonly YumlsschooldbContext _context;

        public TestsController(YumlsschooldbContext context)
        {
            _context = context;
        }

        // GET: api/Tests
        [HttpGet]
        public async Task<ActionResult<IEnumerable<Test>>> GetTests()
        {
            return await _context.Tests.ToListAsync();
        }

        // GET: api/Tests/5
        [HttpGet("{id}")]
        public async Task<ActionResult<Test>> GetTest(int? id)
        {
            var test = await _context.Tests.FindAsync(id);

```

```

        if (test == null)
        {
            return NotFound();
        }

        return test;
    }

// PUT: api/Tests/5
// To protect from overposting attacks, see https://go.microsoft.com/fwlink/?LinkId=2123754
[HttpPut("{id}")]
public async Task<IActionResult> PutTest(int? id, Test test)
{
    if (id != test.IdTest)
    {
        return BadRequest();
    }

    _context.Entry(test).State = EntityState.Modified;

    try
    {
        await _context.SaveChangesAsync();
    }
    catch (DbUpdateConcurrencyException)
    {
        if (!TestExists(id))
        {
            return NotFound();
        }
        else
        {
            throw;
        }
    }
}

return NoContent();
}

// POST: api/Tests
// To protect from overposting attacks, see https://go.microsoft.com/fwlink/?LinkId=2123754
[HttpPost]

```

```

public async Task<ActionResult<Test>> PostTest(Test test)
{
    _context.Tests.Add(test);
    await _context.SaveChangesAsync();

    return CreatedAtAction("GetTest", new { id = test.IdTest }, test);
}

// DELETE: api/Tests/5
[HttpDelete("{id}")]
public async Task<IActionResult> DeleteTest(int? id)
{
    var test = await _context.Tests.FindAsync(id);
    if (test == null)
    {
        return NotFound();
    }

    try
    {
        // Удаляем все попытки прохождения теста
        var attempts = await _context.TestAttempts.Where(a => a.TestId == id).ToListAsync();
        _context.TestAttempts.RemoveRange(attempts);

        // Удаляем все вопросы теста
        var questions = await _context.QuestionsTests.Where(q => q.IdTest == id).ToListAsync();
        _context.QuestionsTests.RemoveRange(questions);

        // Удаляем критерии оценивания
        var grading = await _context.GradingCriteria.Where(g => g.IdTest == id).ToListAsync();
        _context.GradingCriteria.RemoveRange(grading);

        // Удаляем сам тест
        _context.Tests.Remove(test);

        // Сохраняем все изменения
        await _context.SaveChangesAsync();

        return NoContent();
    }
    catch (Exception ex)
    {

```

```

        // Логируем ошибку
        return StatusCode(500, $"Internal server error: {ex.Message}");
    }
}

private bool TestExists(int? id)
{
    return _context.Tests.Any(e => e.IdTest == id);
}
}

```

### TestAttemptsController.cs

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using YumlsSchoolAPI.Models;

namespace YumlsSchoolAPI.Models
{
    [Route("api/[controller]")]
    [ApiController]
    public class TestAttemptsController : ControllerBase
    {
        private readonly YumlsschooldbContext _context;

        public TestAttemptsController(YumlsschooldbContext context)
        {
            _context = context;
        }

        // GET: api/TestAttempts
        [HttpGet]
        public async Task<ActionResult<IEnumerable<TestAttempt>>> GetTestAttempts()
        {
            return await _context.TestAttempts.ToListAsync();
        }

        // GET: api/TestAttempts/5

```

```

[HttpGet("{id}")]
public async Task<ActionResult<TestAttempt>> GetTestAttempt(int? id)
{
    var testAttempt = await _context.TestAttempts.FindAsync(id);

    if (testAttempt == null)
    {
        return NotFound();
    }

    return testAttempt;
}

// PUT: api/TestAttempts/5
// To protect from overposting attacks, see https://go.microsoft.com/fwlink/?linkid=2123754
[HttpPut("{id}")]
public async Task<IActionResult> PutTestAttempt(int? id, TestAttempt testAttempt)
{
    if (id != testAttempt.Id)
    {
        return BadRequest();
    }

    _context.Entry(testAttempt).State = EntityState.Modified;

    try
    {
        await _context.SaveChangesAsync();
    }
    catch (DbUpdateConcurrencyException)
    {
        if (!TestAttemptExists(id))
        {
            return NotFound();
        }
        else
        {
            throw;
        }
    }
}

return NoContent();

```

```

    }

    // POST: api/TestAttempts
    // To protect from overposting attacks, see https://go.microsoft.com/fwlink/?linkid=2123754
    [HttpPost]
    public async Task<ActionResult<TestAttempt>> PostTestAttempt(TestAttempt testAttempt)
    {
        _context.TestAttempts.Add(testAttempt);
        await _context.SaveChangesAsync();

        return CreatedAtAction("GetTestAttempt", new { id = testAttempt.IdTestAttempt }, testAttempt);
    }

    // DELETE: api/TestAttempts/5
    [HttpDelete("{id}")]
    public async Task<IActionResult> DeleteTestAttempt(int? id)
    {
        var testAttempt = await _context.TestAttempts.FindAsync(id);
        if (testAttempt == null)
        {
            return NotFound();
        }

        _context.TestAttempts.Remove(testAttempt);
        await _context.SaveChangesAsync();

        return NoContent();
    }

    private bool TestAttemptExists(int? id)
    {
        return _context.TestAttempts.Any(e => e.IdTestAttempt == id);
    }
}

```

## SubjectsController.cs

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;

```

```

using Microsoft.EntityFrameworkCore;
using YumlsSchoolAPI.Models;

namespace YumlsSchoolAPI.Models
{
    [Route("api/[controller]")]
    [ApiController]
    public class SubjectsController : ControllerBase
    {
        private readonly YumlsschooldbContext _context;

        public SubjectsController(YumlsschooldbContext context)
        {
            _context = context;
        }

        // GET: api/Subjects
        [HttpGet]
        public async Task<ActionResult<IEnumerable<Subject>>> GetSubjects()
        {
            return await _context.Subjects.ToListAsync();
        }

        // GET: api/Subjects/5
        [HttpGet("{id}")]
        public async Task<ActionResult<Subject>> GetSubject(int? id)
        {
            var subject = await _context.Subjects.FindAsync(id);

            if (subject == null)
            {
                return NotFound();
            }

            return subject;
        }

        // PUT: api/Subjects/5
        // To protect from overposting attacks, see https://go.microsoft.com/fwlink/?linkid=2123754
        [HttpPut("{id}")]
        public async Task<IActionResult> PutSubject(int? id, Subject subject)
        {

```

```

        if (id != subject.IdSubject)
        {
            return BadRequest();
        }

        _context.Entry(subject).State = EntityState.Modified;

        try
        {
            await _context.SaveChangesAsync();
        }
        catch (DbUpdateConcurrencyException)
        {
            if (!SubjectExists(id))
            {
                return NotFound();
            }
            else
            {
                throw;
            }
        }

        return NoContent();
    }

    // POST: api/Subjects
    // To protect from overposting attacks, see https://go.microsoft.com/fwlink/?linkid=2123754
    [HttpPost]
    public async Task<ActionResult<Subject>> PostSubject(Subject subject)
    {
        _context.Subjects.Add(subject);
        await _context.SaveChangesAsync();

        return CreatedAtAction("GetSubject", new { id = subject.IdSubject }, subject);
    }

    // DELETE: api/Subjects/5
    [HttpDelete("{id}")]
    public async Task<IActionResult> DeleteSubject(int? id)
    {
        var subject = await _context.Subjects.FindAsync(id);

```

```

        if (subject == null)
        {
            return NotFound();
        }

        _context.Subjects.Remove(subject);
        await _context.SaveChangesAsync();

        return NoContent();
    }

private bool SubjectExists(int? id)
{
    return _context.Subjects.Any(e => e.IdSubject == id);
}
}

```

### StatusController.cs

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using YumlsSchoolAPI.Models;

```

namespace YumlsSchoolAPI.Models

```

{
    [Route("api/[controller]")]
    [ApiController]
    public class StatusController : ControllerBase
    {
        private readonly YumlsschooldbContext _context;

```

```

public StatusController(YumlsschooldbContext context)
{
    _context = context;
}

// GET: api>Status
[HttpGet]
public async Task<ActionResult<IEnumerable<Status>>>
GetStatuses()
{
    return await _context.Statuses.ToListAsync();
}

// GET: api>Status/5
[HttpGet("{id}")]
public async Task<ActionResult<Status>> GetStatus(int? id)
{
    var status = await _context.Statuses.FindAsync(id);

    if (status == null)
    {
        return NotFound();
    }

    return status;
}

// PUT: api>Status/5
// To protect from overposting attacks, see
https://go.microsoft.com/fwlink/?linkid=2123754

```

```

[HttpPut("{id}")]
public async Task<IActionResult> PutStatus(int? id, Status status)
{
    if (id != status.IdStatus)
    {
        return BadRequest();
    }

    _context.Entry(status).State = EntityState.Modified;

    try
    {
        await _context.SaveChangesAsync();
    }
    catch (DbUpdateConcurrencyException)
    {
        if (!StatusExists(id))
        {
            return NotFound();
        }
        else
        {
            throw;
        }
    }
}

return NoContent();
}

// POST: api>Status

```

// To protect from overposting attacks, see  
<https://go.microsoft.com/fwlink/?linkid=2123754>

```

[HttpPost]
public async Task<ActionResult<Status>> PostStatus(Status status)
{
    _context.Statuses.Add(status);
    await _context.SaveChangesAsync();

    return CreatedAtAction("GetStatus", new { id = status.IdStatus },
status);
}

// DELETE: api>Status/5
[HttpDelete("{id}")]
public async Task<IActionResult> DeleteStatus(int? id)
{
    var status = await _context.Statuses.FindAsync(id);
    if (status == null)
    {
        return NotFound();
    }

    _context.Statuses.Remove(status);
    await _context.SaveChangesAsync();

    return NoContent();
}

private bool StatusExists(int? id)
{

```

```

        return _context.Statuses.Any(e => e.IdStatus == id);

    }

}

```

## RolesController.cs

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using YumlsSchoolAPI.Models;

namespace YumlsSchoolAPI.Models
{
    [Route("api/[controller]")]
    [ApiController]
    public class RolesController : ControllerBase
    {
        private readonly YumlsschooldbContext _context;

        public RolesController(YumlsschooldbContext context)
        {
            _context = context;
        }

        // GET: api/Roles
        [HttpGet]
        public async Task<ActionResult<IEnumerable<Role>>> GetRoles()
        {
            return await _context.Roles.ToListAsync();
        }

        // GET: api/Roles/5
        [HttpGet("{id}")]
        public async Task<ActionResult<Role>> GetRole(int? id)
        {
            var role = await _context.Roles.FindAsync(id);

```

```

        if (role == null)
        {
            return NotFound();
        }

        return role;
    }

// PUT: api/Roles/5
// To protect from overposting attacks, see https://go.microsoft.com/fwlink/?LinkId=2123754
[HttpPut("{id}")]
public async Task<IActionResult> PutRole(int? id, Role role)
{
    if (id != role.IdRole)
    {
        return BadRequest();
    }

    _context.Entry(role).State = EntityState.Modified;

    try
    {
        await _context.SaveChangesAsync();
    }
    catch (DbUpdateConcurrencyException)
    {
        if (!RoleExists(id))
        {
            return NotFound();
        }
        else
        {
            throw;
        }
    }
}

return NoContent();
}

// POST: api/Roles
// To protect from overposting attacks, see https://go.microsoft.com/fwlink/?LinkId=2123754
[HttpPost]

```

```

public async Task<ActionResult<Role>> PostRole(Role role)
{
    _context.Roles.Add(role);
    await _context.SaveChangesAsync();

    return CreatedAtAction("GetRole", new { id = role.IdRole }, role);
}

// DELETE: api/Roles/5
[HttpDelete("{id}")]
public async Task<IActionResult> DeleteRole(int? id)
{
    var role = await _context.Roles.FindAsync(id);
    if (role == null)
    {
        return NotFound();
    }

    _context.Roles.Remove(role);
    await _context.SaveChangesAsync();

    return NoContent();
}

private bool RoleExists(int? id)
{
    return _context.Roles.Any(e => e.IdRole == id);
}
}

```

## PracticalWorksController.cs

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using YumlsSchoolAPI.Models;

```

```

namespace YumlsSchoolAPI.Models
{

```

```

[Route("api/[controller]")]
[ApiController]
public class PracticalWorksController : ControllerBase
{
    private readonly YumlsschooldbContext _context;

    public PracticalWorksController(YumlsschooldbContext context)
    {
        _context = context;
    }

    // GET: api/PracticalWorks
    [HttpGet]
    public async Task<ActionResult<IEnumerable<PracticalWork>>> GetPracticalWorks()
    {
        return await _context.PracticalWorks.ToListAsync();
    }

    // GET: api/PracticalWorks/5
    [HttpGet("{id}")]
    public async Task<ActionResult<PracticalWork>> GetPracticalWork(int? id)
    {
        var practicalWork = await _context.PracticalWorks.FindAsync(id);

        if (practicalWork == null)
        {
            return NotFound();
        }

        return practicalWork;
    }

    // PUT: api/PracticalWorks/5
    // To protect from overposting attacks, see https://go.microsoft.com/fwlink/?linkid=2123754
    [HttpPut("{id}")]
    public async Task<IActionResult> PutPracticalWork(int? id, PracticalWork practicalWork)
    {
        if (id != practicalWork.IdPracticalWork)
        {
            return BadRequest();
        }
    }
}

```

```

_context.Entry(practicalWork).State = EntityState.Modified;

try
{
    await _context.SaveChangesAsync();
}

catch (DbUpdateConcurrencyException)
{
    if (!PracticalWorkExists(id))
    {
        return NotFound();
    }
    else
    {
        throw;
    }
}

return NoContent();
}

// POST: api/PracticalWorks
// To protect from overposting attacks, see https://go.microsoft.com/fwlink/?LinkId=2123754
[HttpPost]
public async Task<ActionResult<PracticalWork>> PostPracticalWork(PracticalWork practicalWork)
{
    _context.PracticalWorks.Add(practicalWork);
    await _context.SaveChangesAsync();

    return CreatedAtAction("GetPracticalWork", new { id = practicalWork.IdPracticalWork }, practicalWork);
}

// DELETE: api/PracticalWorks/5
[HttpDelete("{id}")]
public async Task<IActionResult> DeletePracticalWork(int? id)
{
    var practicalWork = await _context.PracticalWorks.FindAsync(id);
    if (practicalWork == null)
    {
        return NotFound();
    }
}

```

```

        _context.PracticalWorks.Remove(practicalWork);
        await _context.SaveChangesAsync();

        return NoContent();
    }

    private bool PracticalWorkExists(int? id)
    {
        return _context.PracticalWorks.Any(e => e.IdPracticalWork == id);
    }

    // GET: api/PracticalWorks/{id}/completed
    [HttpGet("{id}/completed")]
    public async Task<ActionResult<IEnumerable<CompletedWork>>>
GetCompletedWorksByPracticalWork(int id)
{
    var completedWorks = await _context.CompletedWorks
        .Where(cw => cw.PracticalWorkId == id)
        .ToListAsync();

    if (completedWorks == null || !completedWorks.Any())
    {
        return NotFound();
    }

    return completedWorks;
}
}
}

```

### QuestionTestsController.cs

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using YumlsSchoolAPI.Models;

```

```

namespace YumlsSchoolAPI.Models
{

```

```

[Route("api/[controller]")]
[ApiController]
public class QuestionsTestsController : ControllerBase
{
    private readonly YumlsschooldbContext _context;

    public QuestionsTestsController(YumlsschooldbContext context)
    {
        _context = context;
    }

    // GET: api/QuestionsTests
    [HttpGet]
    public async Task<ActionResult<IEnumerable<QuestionsTest>>> GetQuestionsTests()
    {
        return await _context.QuestionsTests.ToListAsync();
    }

    // GET: api/QuestionsTests/5
    [HttpGet("{id}")]
    public async Task<ActionResult<QuestionsTest>> GetQuestionsTest(int? id)
    {
        var questionsTest = await _context.QuestionsTests.FindAsync(id);

        if (questionsTest == null)
        {
            return NotFound();
        }

        return questionsTest;
    }

    // PUT: api/QuestionsTests/5
    // To protect from overposting attacks, see https://go.microsoft.com/fwlink/?linkid=2123754
    [HttpPut("{id}")]
    public async Task<IActionResult> PutQuestionsTest(int? id, QuestionsTest questionsTest)
    {
        if (id != questionsTest.IdQuestion)
        {
            return BadRequest();
        }
    }
}

```

```

_context.Entry(questionsTest).State = EntityState.Modified;

try
{
    await _context.SaveChangesAsync();
}

catch (DbUpdateConcurrencyException)
{
    if (!QuestionsTestExists(id))
    {
        return NotFound();
    }
    else
    {
        throw;
    }
}

return NoContent();
}

// POST: api/QuestionsTests
// To protect from overposting attacks, see https://go.microsoft.com/fwlink/?LinkId=2123754
[HttpPost]
public async Task<ActionResult<QuestionsTest>> PostQuestionsTest(QuestionsTest questionsTest)
{
    _context.QuestionsTests.Add(questionsTest);
    await _context.SaveChangesAsync();

    return CreatedAtAction("GetQuestionsTest", new { id = questionsTest.IdQuestion }, 
questionsTest);
}

// DELETE: api/QuestionsTests/5
[HttpDelete("{id}")]
public async Task<IActionResult> DeleteQuestionsTest(int? id)
{
    var questionsTest = await _context.QuestionsTests.FindAsync(id);
    if (questionsTest == null)
    {
        return NotFound();
    }
}

```

```

        _context.QuestionsTests.Remove(questionsTest);
        await _context.SaveChangesAsync();

        return NoContent();
    }

    private bool QuestionsTestExists(int? id)
    {
        return _context.QuestionsTests.Any(e => e.IdQuestion == id);
    }
}

```

## LectonMaterialsController.cs

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using YumlsSchoolAPI.Models;

namespace YumlsSchoolAPI.Models
{
    [Route("api/[controller]")]
    [ApiController]
    public class LectionMaterialsController : ControllerBase
    {
        private readonly YumlsschooldbContext _context;

        public LectionMaterialsController(YumlsschooldbContext context)
        {
            _context = context;
        }

        // GET: api/LectionMaterials
        [HttpGet]
        public async Task<ActionResult<IEnumerable<LectionMaterial>>> GetLectionMaterials()
        {
            return await _context.LectionMaterials.ToListAsync();
        }
}

```

```

// GET: api/LectionMaterials/5
[HttpGet("{id}")]
public async Task<ActionResult<LectureMaterial>> GetLectureMaterial(int? id)
{
    var lectureMaterial = await _context.LectureMaterials.FindAsync(id);

    if (lectureMaterial == null)
    {
        return NotFound();
    }

    return lectureMaterial;
}

// PUT: api/LectureMaterials/5
// To protect from overposting attacks, see https://go.microsoft.com/fwlink/?linkid=2123754
[HttpPut("{id}")]
public async Task<IActionResult> PutLectureMaterial(int? id, LectureMaterial lectureMaterial)
{
    if (id != lectureMaterial.IdLectureMaterial)
    {
        return BadRequest();
    }

    _context.Entry(lectureMaterial).State = EntityState.Modified;

    try
    {
        await _context.SaveChangesAsync();
    }
    catch (DbUpdateConcurrencyException)
    {
        if (!LectureMaterialExists(id))
        {
            return NotFound();
        }
        else
        {
            throw;
        }
    }
}

```

```

        return NoContent();
    }

    // POST: api/LectionMaterials
    // To protect from overposting attacks, see https://go.microsoft.com/fwlink/?linkid=2123754
    [HttpPost]
    public async Task<ActionResult<LectionMaterial>> PostLectionMaterial(LectionMaterial
        lectionMaterial)
    {
        _context.LectionMaterials.Add(lectionMaterial);
        await _context.SaveChangesAsync();

        return CreatedAtAction("GetLectionMaterial", new { id = lectionMaterial.IdLectionMaterial },
            lectionMaterial);
    }

    // DELETE: api/LectionMaterials/5
    [HttpDelete("{id}")]
    public async Task<IActionResult> DeleteLectionMaterial(int? id)
    {
        var lectionMaterial = await _context.LectionMaterials.FindAsync(id);
        if (lectionMaterial == null)
        {
            return NotFound();
        }

        _context.LectionMaterials.Remove(lectionMaterial);
        await _context.SaveChangesAsync();

        return NoContent();
    }

    private bool LectionMaterialExists(int? id)
    {
        return _context.LectionMaterials.Any(e => e.IdLectionMaterial == id);
    }
}

```

## GroupsController.cs

```

using System;
using System.Collections.Generic;

```

```

using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using YumlsSchoolAPI.Models;

namespace YumlsSchoolAPI.Models
{
    [Route("api/[controller]")]
    [ApiController]
    public class GroupsController : ControllerBase
    {
        private readonly YumlsschooldbContext _context;

        public GroupsController(YumlsschooldbContext context)
        {
            _context = context;
        }

        // GET: api/Groups
        [HttpGet]
        public async Task<ActionResult<IEnumerable<Group>>> GetGroups()
        {
            return await _context.Groups.ToListAsync();
        }

        // GET: api/Groups/5
        [HttpGet("{id}")]
        public async Task<ActionResult<Group>> GetGroup(int? id)
        {
            var @group = await _context.Groups.FindAsync(id);

            if (@group == null)
            {
                return NotFound();
            }

            return @group;
        }

        // PUT: api/Groups/5

```

```

// To protect from overposting attacks, see https://go.microsoft.com/fwlink/?linkid=2123754
[HttpPut("{id}")]
public async Task<IActionResult> PutGroup(int? id, Group @group)
{
    if (id != @group.IdGroup)
    {
        return BadRequest();
    }

    _context.Entry(@group).State = EntityState.Modified;

    try
    {
        await _context.SaveChangesAsync();
    }
    catch (DbUpdateConcurrencyException)
    {
        if (!GroupExists(id))
        {
            return NotFound();
        }
        else
        {
            throw;
        }
    }
}

return NoContent();
}

// POST: api/Groups
// To protect from overposting attacks, see https://go.microsoft.com/fwlink/?linkid=2123754
[HttpPost]
public async Task<ActionResult<Group>> PostGroup(Group @group)
{
    _context.Groups.Add(@group);
    await _context.SaveChangesAsync();

    return CreatedAtAction("GetGroup", new { id = @group.IdGroup }, @group);
}

// DELETE: api/Groups/5

```

```

[HttpDelete("{id}")]
public async Task<IActionResult> DeleteGroup(int? id)
{
    var @group = await _context.Groups.FindAsync(id);
    if (@group == null)
    {
        return NotFound();
    }

    _context.Groups.Remove(@group);
    await _context.SaveChangesAsync();

    return NoContent();
}

private bool GroupExists(int? id)
{
    return _context.Groups.Any(e => e.IdGroup == id);
}

```

## GradingCriterionsController.cs

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using YumlsSchoolAPI.Models;

namespace YumlsSchoolAPI.Models
{
    [Route("api/[controller]")]
    [ApiController]
    public class GradingCriterionsController : ControllerBase
    {
        private readonly YumlsschooldbContext _context;

        public GradingCriterionsController(YumlsschooldbContext context)
        {
            _context = context;
        }
    }
}

```

```

    }

// GET: api/GradingCriterions
[HttpGet]
public async Task<ActionResult<IEnumerable<GradingCriterion>>> GetGradingCriteria()
{
    return await _context.GradingCriteria.ToListAsync();
}

// GET: api/GradingCriterions/5
[HttpGet("{id}")]
public async Task<ActionResult<GradingCriterion>> GetGradingCriterion(int? id)
{
    var gradingCriterion = await _context.GradingCriteria.FindAsync(id);

    if (gradingCriterion == null)
    {
        return NotFound();
    }

    return gradingCriterion;
}

// PUT: api/GradingCriterions/5
// To protect from overposting attacks, see https://go.microsoft.com/fwlink/?linkid=2123754
[HttpPut("{id}")]
public async Task<IActionResult> PutGradingCriterion(int? id, GradingCriterion gradingCriterion)
{
    if (id != gradingCriterion.Id)
    {
        return BadRequest();
    }

    _context.Entry(gradingCriterion).State = EntityState.Modified;

    try
    {
        await _context.SaveChangesAsync();
    }
    catch (DbUpdateConcurrencyException)
    {
        if (!GradingCriterionExists(id))

```

```

    {
        return NotFound();
    }
    else
    {
        throw;
    }
}

return NoContent();
}

// POST: api/GradingCriterions
// To protect from overposting attacks, see https://go.microsoft.com/fwlink/?linkid=2123754
[HttpPost]
public async Task<ActionResult<GradingCriterion>> PostGradingCriterion(GradingCriterion
gradingCriterion)
{
    _context.GradingCriteria.Add(gradingCriterion);
    await _context.SaveChangesAsync();

    return CreatedAtAction("GetGradingCriterion", new { id = gradingCriterion.IdCriteria }, gradingCriterion);
}

// DELETE: api/GradingCriterions/5
[HttpDelete("{id}")]
public async Task<IActionResult> DeleteGradingCriterion(int? id)
{
    var gradingCriterion = await _context.GradingCriteria.FindAsync(id);
    if (gradingCriterion == null)
    {
        return NotFound();
    }

    _context.GradingCriteria.Remove(gradingCriterion);
    await _context.SaveChangesAsync();

    return NoContent();
}

private bool GradingCriterionExists(int? id)

```

```

    {
        return _context.GradingCriteria.Any(e => e.IdCriteria == id);
    }
}
}

```

## CompleteWorksController.cs

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using YumlsSchoolAPI.Models;

namespace YumlsSchoolAPI.Models
{
    [Route("api/[controller]")]
    [ApiController]
    public class CompletedWorksController : ControllerBase
    {
        private readonly YumlsschooldbContext _context;

        public CompletedWorksController(YumlsschooldbContext context)
        {
            _context = context;
        }

        // GET: api/CompletedWorks
        [HttpGet]
        public async Task<ActionResult<IEnumerable<CompletedWork>>> GetCompletedWorks()
        {
            return await _context.CompletedWorks.ToListAsync();
        }

        // GET: api/CompletedWorks/5
        [HttpGet("{id}")]
        public async Task<ActionResult<CompletedWork>> GetCompletedWork(int? id)
        {
            var completedWork = await _context.CompletedWorks.FindAsync(id);

            if (completedWork == null)

```

```

    {
        return NotFound();
    }

    return completedWork;
}

// PUT: api/CompletedWorks/5
// To protect from overposting attacks, see https://go.microsoft.com/fwlink/?linkid=2123754
[HttpPut("{id}")]
public async Task<IActionResult> PutCompletedWork(int? id, CompletedWork completedWork)
{
    if (id != completedWork.IdCompletedWork)
    {
        return BadRequest();
    }

    _context.Entry(completedWork).State = EntityState.Modified;

    try
    {
        await _context.SaveChangesAsync();
    }
    catch (DbUpdateConcurrencyException)
    {
        if (!CompletedWorkExists(id))
        {
            return NotFound();
        }
        else
        {
            throw;
        }
    }
}

return NoContent();
}

// POST: api/CompletedWorks
// To protect from overposting attacks, see https://go.microsoft.com/fwlink/?linkid=2123754
[HttpPost]

```

```

public     async     Task<ActionResult<CompletedWork>>     PostCompletedWork(CompletedWork
completedWork)
{
    _context.CompletedWorks.Add(completedWork);
    await _context.SaveChangesAsync();

    return CreatedAtAction("GetCompletedWork", new { id = completedWork.IdCompletedWork },
completedWork);
}

// DELETE: api/CompletedWorks/5
[HttpDelete("{id}")]
public async Task<IActionResult> DeleteCompletedWork(int? id)
{
    var completedWork = await _context.CompletedWorks.FindAsync(id);
    if (completedWork == null)
    {
        return NotFound();
    }

    _context.CompletedWorks.Remove(completedWork);
    await _context.SaveChangesAsync();

    return NoContent();
}

private bool CompletedWorkExists(int? id)
{
    return _context.CompletedWorks.Any(e => e.IdCompletedWork == id);
}
}

```

## AdminPanel.css

```

:root {
    --dark-purple: #1a1225;
    --medium-purple: #7743DB;
    --light-purple: #B39DDB;
    --accent-purple: #9D4EDD;
    --text-light: #ffffff;
    --text-secondary: rgba(255, 255, 255, 0.85);
}

```

```
--card-bg: #272134;
--card-hover: #362b48;
--gradient-bg: linear-gradient(135deg, #1a1225 0%, #3a1d69 100%);
--shadow-sm: 0 4px 6px rgba(0, 0, 0, 0.3);
--shadow-md: 0 6px 12px rgba(0, 0, 0, 0.4);
--shadow-lg: 0 10px 25px rgba(0, 0, 0, 0.5);
--border-radius: 12px;
}
```

```
/* Глобальные стили для текста */
```

```
*, *::before, *::after {
    color: var(--text-light) !important;
}
```

```
body {
    font-family: 'Roboto', sans-serif;
    background-color: var(--dark-purple) !important;
    margin: 0;
    min-height: 100vh;
    color: var(--text-light) !important;
}
```

```
/* Стили для header */
```

```
.main-header {
    background-color: var(--dark-purple) !important;
}
```

```
.main-header .container {
    background: transparent !important;
    animation: none !important;
    box-shadow: none !important;
}
```

```
.logo,
.logo i,
.logo span {
    color: var(--text-light) !important;
}
```

```
.user-info,
.user-name,
.user-role {
```

```
color: var(--text-light) !important;
}

.logout-form {
  margin: 0;
}

.logout-btn {
  display: inline-flex;
  align-items: center;
  gap: 0.5rem;
  padding: 0.5rem 1rem;
  background-color: transparent;
  border: 1px solid rgba(255, 255, 255, 0.1);
  border-radius: var(--border-radius);
  color: var(--text-light) !important;
  font-size: 0.9rem;
  cursor: pointer;
  transition: all 0.3s ease;
  width: auto;
}

.logout-btn:hover {
  background-color: rgba(255, 255, 255, 0.1);
  transform: translateY(-2px);
  box-shadow: var(--shadow-md);
}

.logout-btn i {
  font-size: 1rem;
  color: var(--text-light) !important;
}

/* Основной контейнер */
.container {
  max-width: 1200px;
  margin: 2rem auto;
  padding: 2rem;
  background: var(--gradient-bg) !important;
  border-radius: var(--border-radius);
  box-shadow: var(--shadow-md);
  position: relative;
}
```

```

        overflow: hidden;
        animation: fadeIn 0.5s ease-out forwards;
    }

.container::before {
    content: "";
    position: absolute;
    top: 0;
    left: 0;
    right: 0;
    bottom: 0;
    background: url('data:image/svg+xml;utf8,<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 100 100" preserveAspectRatio="none"><path fill="none" stroke="rgba(255,255,255,0.05)" stroke-width="1" d="M0,0 L100,100 M100,0 L0,100"/></svg>');
    background-size: 20px 20px;
    opacity: 0.3;
    z-index: 0;
}

h1 {
    text-align: center;
    color: var(--text-light) !important;
    font-size: 2.5rem;
    font-weight: 700;
    margin-bottom: 2rem;
    text-shadow: 0 2px 4px rgba(0, 0, 0, 0.3);
    position: relative;
    z-index: 1;
}

.menu {
    position: relative;
    z-index: 1;
}

.menu ul {
    list-style-type: none;
    padding: 0;
    display: grid;
    grid-template-columns: repeat(auto-fit, minmax(250px, 1fr));
    gap: 1.5rem;
    margin: 2rem 0;
}

```

```
}

.menu li {
    margin: 0;
}

.menu a {
    text-decoration: none;
    padding: 1.5rem;
    background-color: var(--card-bg) !important;
    color: var(--text-light) !important;
    border-radius: var(--border-radius);
    transition: all 0.3s cubic-bezier(0.165, 0.84, 0.44, 1);
    display: block;
    text-align: center;
    font-size: 1.1rem;
    font-weight: 500;
    border: 1px solid rgba(255, 255, 255, 0.1);
    box-shadow: var(--shadow-sm);
}

.menu a:hover {
    background-color: var(--card-hover) !important;
    transform: translateY(-5px);
    box-shadow: var(--shadow-lg);
    border-color: var(--accent-purple);
}

.back-button {
    display: inline-flex;
    align-items: center;
    gap: 0.5rem;
    padding: 1rem 2rem;
    background-color: var(--medium-purple);
    color: var(--text-light);
    border-radius: var(--border-radius);
    text-decoration: none;
    font-weight: 500;
    transition: all 0.3s ease;
    border: 1px solid rgba(255, 255, 255, 0.1);
    box-shadow: var(--shadow-sm);
    margin-top: 2rem;
}
```

```
}

.back-button:hover {
    background-color: var(--accent-purple);
    transform: translateY(-2px);
    box-shadow: var(--shadow-md);
}

/* Анимация появления */
@keyframes fadeIn {
    from {
        opacity: 0;
        transform: translateY(20px);
    }
    to {
        opacity: 1;
        transform: translateY(0);
    }
}

/* Адаптивность */
@media (max-width: 768px) {
    .container {
        margin: 1rem;
        padding: 1.5rem;
    }

    h1 {
        font-size: 2rem;
    }

    .menu ul {
        grid-template-columns: 1fr;
    }

    .menu a {
        padding: 1.2rem;
    }
}

/* Экспорт данных */
.export-actions {
```

```

display: flex;
justify-content: center;
gap: 1.5rem;
padding: 1.5rem;
flex-wrap: wrap;
background-color: var(--card-bg);
}

.export-button {
padding: 1rem 1.5rem;
background: var(--gradient-primary);
min-width: 220px;
font-size: 1.05rem;
}

/* Дополнительные стили для обеспечения белого цвета текста */
h1, h2, h3, h4, h5, h6,
p, span, a, div {
color: var(--text-light) !important;
}

.menu a {
color: var(--text-light) !important;
}

.menu a:hover {
color: var(--text-light) !important;
}

/* Стили для иконок */
i, .fas, .far, .fab, .fa {
color: var(--text-light) !important;
}

```

## AllSubjectPage.css

```

/* Современный адаптивный стиль для страницы предметов - темная тема */
:root {
--dark-purple: #1a1225;

```

```
--medium-purple: #7743DB;  
--light-purple: #B39DDB;  
--accent-purple: #9D4EDD;  
--text-light: #ffffff;  
--text-secondary: rgba(255, 255, 255, 0.85);  
--card-bg: #272134;  
--card-hover: #362b48;  
--gradient-bg: linear-gradient(135deg, #1a1225 0%, #3a1d69 100%);  
--shadow-sm: 0 4px 6px rgba(0, 0, 0, 0.3);  
--shadow-md: 0 6px 12px rgba(0, 0, 0, 0.4);  
--shadow-lg: 0 10px 25px rgba(0, 0, 0, 0.5);  
--transition-slow: all 0.4s cubic-bezier(0.165, 0.84, 0.44, 1);  
--transition-fast: all 0.2s ease;  
--border-radius: 12px;  
--card-spacing: 1.75rem;  
}  
  
/* Принудительное переопределение базовых стилей */
```

```
html {  
    background-color: var(--dark-purple) !important;  
}
```

```
body {  
    background-color: var(--dark-purple) !important;  
    color: var(--text-light) !important;  
    font-family: 'Roboto', sans-serif;  
    margin: 0;  
    padding: 0;  
    min-height: 100vh;  
    width: 100%;  
    overflow-x: hidden; /* Предотвращает горизонтальный скролл */  
}
```

```
/* Корректировка стилей header */  
.main-header {  
    background-color: var(--dark-purple) !important;  
    width: 100%;  
    box-shadow: var(--shadow-md);  
    position: relative;  
    z-index: 100;  
    border-bottom: 1px solid rgba(255, 255, 255, 0.1);  
}
```

```
.main-header .container {
    max-width: 100% !important;
    width: 100%;
    padding: 0.8rem 2rem;
    display: flex;
    justify-content: space-between;
    align-items: center;
    margin: 0 !important;
    background-color: transparent !important;
    box-shadow: none !important;
}

.main-header .logo {
    color: var(--accent-purple);
}

.main-header .user-info {
    color: var(--text-light);
}

.logout-btn {
    background-color: var(--medium-purple);
    color: white;
    padding: 0.5rem 1rem;
    border-radius: 6px;
    text-decoration: none;
    display: inline-flex;
    align-items: center;
    gap: 0.5rem;
    transition: var(--transition-fast);
}

.logout-btn:hover {
    background-color: var(--accent-purple);
    transform: translateY(-2px);
    text-decoration: none;
}

/* Шапка страницы */
.hero-section {
    background: var(--gradient-bg) !important;
```

```

padding: 4rem 2rem;
text-align: center;
box-shadow: var(--shadow-md);
position: relative;
overflow: hidden;
width: 100%;
}

.hero-section::before {
content: "";
position: absolute;
top: 0;
left: 0;
right: 0;
bottom: 0;
background: url('data:image/svg+xml;utf8,<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 100 100" preserveAspectRatio="none"><path fill="none" stroke="rgba(255,255,255,0.05)" stroke-width="1" d="M0,0 L100,100 M100,0 L0,100"/></svg>');
background-size: 20px 20px;
opacity: 0.3;
}

.hero-section .container {
position: relative;
z-index: 1;
max-width: 800px;
margin: 0 auto;
background: transparent !important;
box-shadow: none !important;
}

.hero-section h1 {
font-size: 3rem;
font-weight: 700;
margin-bottom: 1.5rem;
color: white !important;
text-shadow: 0 2px 4px rgba(0, 0, 0, 0.2);
animation: fadeInDown 0.6s ease-out;
}

.hero-section p {
font-size: 1.2rem;
}

```

```
line-height: 1.6;  
max-width: 600px;  
margin: 0 auto;  
color: white !important;  
animation: fadeInUp 0.6s ease-out 0.2s both;  
font-weight: 400;  
}  
  
/* Контейнер для предметов */
```

```
.subjects-container {  
width: 100%;  
max-width: 1280px;  
margin: 0 auto;  
padding: 3rem 2rem;  
position: relative;  
box-sizing: border-box;  
background-color: var(--dark-purple) !important;  
}
```

```
/* Сетка предметов */  
.subject-list {  
display: grid;  
grid-template-columns: repeat(auto-fill, minmax(320px, 1fr));  
gap: 2.5rem;  
animation: fadeIn 0.8s ease-out;  
width: 100%;  
}
```

```
/* ПОЛНОСТЬЮ ОБНОВЛЕННЫЙ ДИЗАЙН КАРТОЧЕК */  
.subject-card {  
position: relative;  
height: 100%;  
min-height: 380px;  
perspective: 1000px;  
transform-style: preserve-3d;  
}
```

```
.card-link {  
display: block;  
position: relative;  
width: 100%;  
height: 100%;
```

```
text-decoration: none;  
color: inherit;  
transform-style: preserve-3d;  
transition: transform 0.8s cubic-bezier(0.175, 0.885, 0.32, 1.275);  
}
```

```
.subject-card:hover .card-link {  
    transform: rotateY(10deg) translateZ(10px);  
}
```

```
/* Основная часть карточки */  
.card-content {  
    position: relative;  
    width: 100%;  
    height: 100%;  
    border-radius: 16px;  
    overflow: hidden;  
    background: linear-gradient(145deg, #2d1f3a, #231630);  
    box-shadow: 0 15px 25px rgba(0, 0, 0, 0.4),  
               0 5px 10px rgba(0, 0, 0, 0.3),  
               inset 0 1px 1px rgba(255, 255, 255, 0.1);  
    transform: translateZ(0);  
    transition: all 0.5s;  
    border: 1px solid rgba(255, 255, 255, 0.05);  
}
```

```
.subject-card:hover .card-content {  
    box-shadow: 0 20px 35px rgba(0, 0, 0, 0.5),  
               0 10px 15px rgba(0, 0, 0, 0.4),  
               inset 0 1px 1px rgba(255, 255, 255, 0.15);  
    border-color: rgba(255, 255, 255, 0.1);  
}
```

```
/* Эффект свечения вокруг карточки при наведении */  
.card-content::before {  
    content: " ";  
    position: absolute;  
    top: -2px;  
    left: -2px;  
    right: -2px;  
    bottom: -2px;  
    background: linear-gradient(45deg,
```

```

    var(--accent-purple),
    transparent,
    var(--medium-purple),
    transparent,
    var(--accent-purple)
);
z-index: -1;
border-radius: 18px;
opacity: 0;
transition: opacity 0.5s;
animation: glowingBorder 3s linear infinite;
filter: blur(8px);
}

.subject-card:hover .card-content::before {
    opacity: 0.7;
}

@keyframes glowingBorder {
    0% { background-position: 0 0; }
    50% { background-position: 400% 0; }
    100% { background-position: 0 0; }
}

/* Заголовок карточки */
.card-header {
    position: relative;
    padding: 2rem 1.5rem;
    text-align: center;
    background: linear-gradient(135deg, var(--medium-purple) 0%, var(--accent-purple) 100%);
    overflow: hidden;
    z-index: 1;
}

.card-header::before {
    content: "";
    position: absolute;
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
}

```

```

background: url('data:image/svg+xml;utf8,<svg xmlns="http://www.w3.org/2000/svg" width="100" height="100" viewBox="0 0 100 100"><circle cx="12" cy="12" r="1.5" fill="rgba(255,255,255,0.2)" /></svg>');
background-size: 20px 20px;
opacity: 0.3;
z-index: -1;
}

.subject-icon {
position: relative;
display: inline-flex;
align-items: center;
justify-content: center;
width: 80px;
height: 80px;
margin-bottom: 1rem;
background: rgba(0, 0, 0, 0.2);
border-radius: 50%;
box-shadow: 0 5px 15px rgba(0, 0, 0, 0.2),
            inset 0 2px 5px rgba(0, 0, 0, 0.2);
transition: all 0.3s;
}

.subject-icon i {
font-size: 2.5rem;
color: white;
text-shadow: 0 2px 5px rgba(0, 0, 0, 0.3);
transition: all 0.3s;
}

.subject-card:hover .subject-icon {
transform: scale(1.1) translateY(-5px);
box-shadow: 0 8px 20px rgba(0, 0, 0, 0.3),
            inset 0 2px 5px rgba(0, 0, 0, 0.2);
}

.subject-card:hover .subject-icon i {
transform: rotate(10deg);
}

.subject-title {
font-size: 1.6rem;
font-weight: 700;
}

```

```
color: white !important;  
margin: 0;  
text-shadow: 0 2px 4px rgba(0, 0, 0, 0.3);  
transition: all 0.3s;  
letter-spacing: 0.5px;  
word-break: break-word;  
}
```

```
.subject-card:hover .subject-title {  
    transform: translateY(-2px);  
    text-shadow: 0 4px 8px rgba(0, 0, 0, 0.4);  
}
```

```
/* Тело карточки */  
.card-body {  
    padding: 1.5rem;  
    display: flex;  
    flex-direction: column;  
    flex: 1;  
    background-color: rgba(0, 0, 0, 0.1);  
    backdrop-filter: blur(5px);  
}
```

```
.subject-info {  
    flex: 1;  
    margin-bottom: 1.5rem;  
}
```

```
.info-item {  
    position: relative;  
    display: flex;  
    align-items: flex-start;  
    margin-bottom: 1rem;  
    padding: 1rem;  
    background: rgba(0, 0, 0, 0.15);  
    border-radius: 10px;  
    border-left: 3px solid var(--accent-purple);  
    transition: all 0.3s;  
    overflow: hidden;  
}
```

```
.info-item::before {
```

```
content: "";
position: absolute;
top: 0;
left: 0;
width: 100%;
height: 100%;
background: linear-gradient(90deg,
    rgba(157, 78, 221, 0.1),
    transparent
);
opacity: 0;
transition: opacity 0.3s;
}
```

```
.subject-card:hover .info-item::before {
    opacity: 1;
}
```

```
.info-item i {
    min-width: 24px;
    margin-right: 1rem;
    color: var(--light-purple);
    font-size: 1.1rem;
    transition: all 0.3s;
}
```

```
.subject-card:hover .info-item i {
    transform: scale(1.2);
    color: var(--accent-purple);
}
```

```
.info-item span {
    font-size: 0.95rem;
    line-height: 1.4;
    color: white !important;
    word-break: break-word;
    transition: all 0.3s;
    font-weight: 400;
}
```

```
.subject-card:hover .info-item span {
    color: white;
```

```
}

/* Кнопка действия */
.subject-actions {
    margin-top: auto;
}

.view-subject {
    position: relative;
    display: block;
    width: 100%;
    padding: 1rem;
    background: linear-gradient(135deg, var(--accent-purple) 0%, var(--medium-purple) 100%);
    color: white !important;
    text-align: center;
    text-decoration: none;
    border-radius: 10px;
    font-weight: 600;
    font-size: 1rem;
    letter-spacing: 0.5px;
    border: none;
    overflow: hidden;
    transition: all 0.3s;
    z-index: 1;
    box-shadow: 0 4px 10px rgba(0, 0, 0, 0.3);
}

.view-subject::before {
    content: "";
    position: absolute;
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
    background: linear-gradient(90deg,
        transparent,
        rgba(255, 255, 255, 0.2),
        transparent
    );
    transform: translateX(-100%);
    transition: transform 0.6s;
    z-index: -1;
}
```

```
}

.subject-card:hover .view-subject {
    transform: translateY(-3px);
    box-shadow: 0 7px 15px rgba(0, 0, 0, 0.4);
}

.subject-card:hover .view-subject::before {
    transform: translateX(100%);
}

/* Статус предмета */
.subject-status {
    position: absolute;
    top: 15px;
    right: 15px;
    padding: 0.5rem 1rem;
    background: rgba(0, 0, 0, 0.4);
    color: white !important;
    font-size: 0.8rem;
    font-weight: 600;
    border-radius: 30px;
    z-index: 10;
    backdrop-filter: blur(5px);
    border: 1px solid rgba(255, 255, 255, 0.1);
    box-shadow: 0 2px 8px rgba(0, 0, 0, 0.2);
    transition: all 0.3s;
}

.subject-card:hover .subject-status {
    background: var(--accent-purple);
    transform: translateY(-2px);
    box-shadow: 0 4px 12px rgba(0, 0, 0, 0.3);
}

/* Состояние "нет предметов" */
.no-subjects {
    text-align: center;
    padding: 5rem 2rem;
    color: var(--text-light);
    opacity: 0.8;
    background: rgba(255, 255, 255, 0.03);
```

```
border-radius: var(--border-radius);
backdrop-filter: blur(5px);
max-width: 600px;
margin: 0 auto;
animation: fadeIn 0.8s ease-out;
border: 1px solid rgba(255, 255, 255, 0.05);
box-shadow: 0 10px 30px rgba(0, 0, 0, 0.2);
}
```

```
.no-subjects i {
  font-size: 5rem;
  color: var(--light-purple);
  margin-bottom: 1.5rem;
  opacity: 0.6;
}
```

```
.no-subjects h2 {
  font-size: 2rem;
  margin-bottom: 1rem;
  color: white !important;
  text-shadow: 0 2px 4px rgba(0, 0, 0, 0.3);
}
```

```
.no-subjects p {
  font-size: 1.1rem;
  color: white !important;
  max-width: 400px;
  margin: 0 auto;
  font-weight: 400;
}
```

```
/* Анимации */
@keyframes fadeIn {
  from { opacity: 0; }
  to { opacity: 1; }
}
```

```
@keyframes fadeInDown {
  from {
    opacity: 0;
    transform: translateY(-20px);
  }
}
```

```
        to {
          opacity: 1;
          transform: translateY(0);
        }
      }

      @keyframes fadeInUp {
        from {
          opacity: 0;
          transform: translateY(20px);
        }
        to {
          opacity: 1;
          transform: translateY(0);
        }
      }

      /* Адаптивный дизайн */
      @media (max-width: 992px) {
        .subject-list {
          grid-template-columns: repeat(auto-fill, minmax(280px, 1fr));
          gap: 2rem;
        }
      }

      .hero-section h1 {
        font-size: 2.5rem;
      }

      @media (max-width: 768px) {
        .hero-section {
          padding: 3rem 1.5rem;
        }
      }

      .hero-section h1 {
        font-size: 2.2rem;
      }

      .subjects-container {
        padding: 2rem 1rem;
      }
```

```
.subject-list {  
    grid-template-columns: repeat(auto-fill, minmax(240px, 1fr));  
}  
  
.main-header .container {  
    padding: 0.8rem 1rem;  
}  
  
.subject-icon {  
    width: 70px;  
    height: 70px;  
}  
  
.subject-icon i {  
    font-size: 2.2rem;  
}  
  
.subject-title {  
    font-size: 1.4rem;  
}  
}  
  
@media (max-width: 576px) {  
    .hero-section h1 {  
        font-size: 1.8rem;  
    }  
  
.hero-section p {  
    font-size: 1rem;  
}  
  
.subject-list {  
    grid-template-columns: 1fr;  
}  
  
.card-header {  
    padding: 1.5rem 1rem;  
}  
  
.card-body {  
    padding: 1.2rem;  
}
```

```
.subject-icon {  
    width: 60px;  
    height: 60px;  
}  
  
.subject-icon i {  
    font-size: 2rem;  
}  
  
.subject-title {  
    font-size: 1.3rem;  
}  
  
.info-item {  
    padding: 0.8rem;  
}  
}  
  
/* Добавьте эти стили в конец файла AllSubjectPage.css */  
  
.subject-card-link {  
    display: block;  
    text-decoration: none;  
    color: inherit;  
    height: 100%;  
    width: 100%;  
    transition: transform 0.3s;  
    cursor: pointer;  
}  
  
.subject-card-link:hover {  
    transform: translateY(-5px);  
    text-decoration: none;  
}  
  
.subject-card-link:hover .card-content::before {  
    opacity: 0.7;  
}  
  
.subject-card-link:hover .subject-icon {  
    transform: scale(1.1) translateY(-5px);  
}
```

```
    box-shadow: 0 8px 20px rgba(0, 0, 0, 0.3),  
               inset 0 2px 5px rgba(0, 0, 0, 0.2);  
  }  
  
.subject-card-link:hover .subject-icon i {  
  transform: rotate(10deg);  
}  
  
.subject-card-link:hover .subject-title {  
  transform: translateY(-2px);  
  text-shadow: 0 4px 8px rgba(0, 0, 0, 0.4);  
}  
  
.subject-card-link:hover .info-item::before {  
  opacity: 1;  
}  
  
.subject-card-link:hover .info-item i {  
  transform: scale(1.2);  
  color: var(--accent-purple);  
}  
  
.subject-card-link:hover .info-item span {  
  color: white;  
}  
  
.subject-card-link:hover .view-subject {  
  transform: translateY(-3px);  
  box-shadow: 0 7px 15px rgba(0, 0, 0, 0.4);  
}  
  
.subject-card-link:hover .view-subject::before {  
  transform: translateX(100%);  
}  
  
.subject-card-link:hover .subject-status {  
  background: var(--accent-purple);  
  transform: translateY(-2px);  
  box-shadow: 0 4px 12px rgba(0, 0, 0, 0.3);  
}  
  
.subject-card-link:hover .card-content {
```

```
    box-shadow: 0 20px 35px rgba(0, 0, 0, 0.5),  
              0 10px 15px rgba(0, 0, 0, 0.4),  
              inset 0 1px 1px rgba(255, 255, 255, 0.15);  
    border-color: rgba(255, 255, 255, 0.1);  
}
```

## base.css

```
/* base.css - Основные стили для всего приложения */
```

```
:root {  
  /* Основная цветовая палитра */  
  --primary: #7743DB;  
  --primary-dark: #5E35B1;  
  --primary-light: #B39DDB;  
  --secondary: #3F51B5;  
  --secondary-dark: #303F9F;  
  --secondary-light: #C5CAE9;  
  
  /* Нейтральные цвета */  
  --text: #ffffff;  
  --text-light: rgba(255, 255, 255, 0.85);  
  --text-lighter: rgba(255, 255, 255, 0.7);  
  --bg-light: #F5F7FA;  
  --bg-white: #FFFFFF;  
  
  /* Функциональные цвета */  
  --success: #4CAF50;  
  --success-light: #E8F5E9;  
  --warning: #FFC107;  
  --warning-light: #FFF8E1;  
  --danger: #F44336;  
  --danger-light: #FFEBEE;  
  --info: #2196F3;  
  --info-light: #E3F2FD;  
  
  /* Оформление */  
  --border-radius: 8px;  
  --box-shadow: 0 2px 10px rgba(0, 0, 0, 0.05);  
  --box-shadow-hover: 0 5px 15px rgba(0, 0, 0, 0.1);  
  --transition: all 0.3s ease;  
}  
  
/* Общие стили */
```

```
* {  
    margin: 0;  
    padding: 0;  
    box-sizing: border-box;  
}  
  
body {  
    font-family: 'Roboto', 'Segoe UI', Arial, sans-serif;  
    font-size: 16px;  
    line-height: 1.6;  
    color: var(--text);  
    background-color: var(--bg-light);  
    -webkit-font-smoothing: antialiased;  
    -moz-osx-font-smoothing: grayscale;  
}  
  
/* Контейнеры */  
.container {  
    max-width: 1200px;  
    margin: 2rem auto;  
    padding: 0 1.5rem;  
}  
  
/* Типография */  
h1, h2, h3, h4, h5, h6 {  
    font-weight: 600;  
    margin-bottom: 1rem;  
    color: var(--text) !important;  
    line-height: 1.3;  
}  
  
h1 {  
    font-size: 2.5rem;  
    margin-bottom: 1.5rem;  
    color: var(--text) !important;  
}  
  
h2 {  
    font-size: 2rem;  
    margin-top: 2rem;  
    margin-bottom: 1rem;  
    border-bottom: 2px solid var(--primary-light);
```

```
padding-bottom: 0.5rem;  
}  
  
h3 {  
    font-size: 1.5rem;  
    margin-top: 1.5rem;  
}  
  
p {  
    margin-bottom: 1rem;  
}  
  
a {  
    color: var(--text) !important;  
    text-decoration: none;  
    transition: var(--transition);  
}  
  
a:hover {  
    color: var(--primary-light) !important;  
    text-decoration: underline;  
}  
  
/* Кнопки */  
.btn,  
button[type="submit"],  
input[type="submit"] {  
    display: inline-block;  
    padding: 0.75rem 1.5rem;  
    border-radius: var(--border-radius);  
    background-color: var(--primary);  
    color: var(--text) !important;  
    font-weight: 500;  
    text-align: center;  
    text-decoration: none;  
    cursor: pointer;  
    border: none;  
    transition: var(--transition);  
    box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);  
}  
  
.btn:hover,
```

```
button[type="submit"]:hover,  
input[type="submit"]:hover {  
    background-color: var(--primary-dark);  
    transform: translateY(-2px);  
    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.15);  
    text-decoration: none;  
}  
  
.btn-secondary {  
    background-color: var(--secondary);  
}  
  
.btn-secondary:hover {  
    background-color: var(--secondary-dark);  
}  
  
.btn-success {  
    background-color: var(--success);  
}  
  
.btn-success:hover {  
    background-color: var(--success);  
    filter: brightness(90%);  
}  
  
.btn-warning {  
    background-color: var(--warning);  
    color: var(--text);  
}  
  
.btn-warning:hover {  
    background-color: var(--warning);  
    filter: brightness(90%);  
}  
  
.btn-danger {  
    background-color: var(--danger);  
}  
  
.btn-danger:hover {  
    background-color: var(--danger);  
    filter: brightness(90%);  
}
```

```
}

.btn-outline {
    background-color: transparent;
    border: 2px solid var(--text);
    color: var(--text) !important;
}

.btn-outline:hover {
    color: var(--text) !important;
    background-color: var(--primary);
}

.btn-sm {
    padding: 0.5rem 1rem;
    font-size: 0.875rem;
}

.btn-lg {
    padding: 1rem 2rem;
    font-size: 1.125rem;
}

.btn-block {
    display: block;
    width: 100%;
}

.btn-icon {
    display: inline-flex;
    align-items: center;
    gap: 0.5rem;
}

/* Ввод данных */
input, select, textarea {
    width: 100%;
    padding: 0.75rem 1rem;
    border: 1px solid rgba(255, 255, 255, 0.1);
    border-radius: var(--border-radius);
    background-color: var(--card-bg);
    font-size: 1rem;
}
```

```
font-family: inherit;
transition: var(--transition);
color: var(--text) !important;
}

input:focus, select:focus, textarea:focus {
outline: none;
border-color: var(--primary);
box-shadow: 0 0 0 2px var(--primary-light);
}

textarea {
min-height: 120px;
resize: vertical;
}

.form-group {
margin-bottom: 1.5rem;
}

.form-group label {
display: block;
margin-bottom: 0.5rem;
font-weight: 500;
color: var(--text) !important;
}

/* Карточки */
.card {
background-color: var(--card-bg);
border-radius: var(--border-radius);
box-shadow: var(--box-shadow);
padding: 1.5rem;
margin-bottom: 1.5rem;
transition: var(--transition);
color: var(--text) !important;
}

.card:hover {
box-shadow: var(--box-shadow-hover);
transform: translateY(-5px);
}
```

```
.card-header {  
    /* margin: -1.5rem -1.5rem 1.5rem -1.5rem; */  
    padding: 1.5rem;  
    border-bottom: 1px solid rgba(255, 255, 255, 0.1);  
    border-radius: var(--border-radius) var(--border-radius) 0 0;  
    background-color: var(--card-bg);  
}  
  
.main-header .container {  
    max-width: 100% !important;  
    width: 100%;  
    padding: 0.8rem 2rem;  
    display: flex;  
    justify-content: space-between;  
    align-items: center;  
    margin: 0 !important;  
    background-color: transparent !important;  
    box-shadow: none !important;  
}  
  
.card-footer {  
    margin: 1.5rem -1.5rem -1.5rem -1.5rem;  
    padding: 1.5rem;  
    border-top: 1px solid rgba(255, 255, 255, 0.1);  
    border-radius: 0 0 var(--border-radius) var(--border-radius);  
    background-color: var(--card-bg);  
}  
  
/* Сетки */  
.grid {  
    display: grid;  
    grid-template-columns: repeat(auto-fill, minmax(300px, 1fr));  
    gap: 1.5rem;  
}  
  
.flex {  
    display: flex;  
}  
  
.flex-column {  
    flex-direction: column;
```

```
}

.justify-between {
  justify-content: space-between;
}

.justify-center {
  justify-content: center;
}

.items-center {
  align-items: center;
}

.gap-1 {
  gap: 0.5rem;
}

.gap-2 {
  gap: 1rem;
}

.gap-3 {
  gap: 1.5rem;
}

/* Утилиты */

.mt-1 { margin-top: 0.5rem; }
.mt-2 { margin-top: 1rem; }
.mt-3 { margin-top: 1.5rem; }
.mt-4 { margin-top: 2rem; }
.mt-5 { margin-top: 2.5rem; }

.mb-1 { margin-bottom: 0.5rem; }
.mb-2 { margin-bottom: 1rem; }
.mb-3 { margin-bottom: 1.5rem; }
.mb-4 { margin-bottom: 2rem; }
.mb-5 { margin-bottom: 2.5rem; }

.ml-1 { margin-left: 0.5rem; }
.ml-2 { margin-left: 1rem; }
.ml-3 { margin-left: 1.5rem; }
```

```
.mr-1 { margin-right: 0.5rem; }
.mr-2 { margin-right: 1rem; }
.mr-3 { margin-right: 1.5rem; }

.p-1 { padding: 0.5rem; }
.p-2 { padding: 1rem; }
.p-3 { padding: 1.5rem; }
.p-4 { padding: 2rem; }
.p-5 { padding: 2.5rem; }

.text-center { text-align: center; }
.text-right { text-align: right; }
.text-left { text-align: left; }
.text-sm { font-size: 0.875rem; }
.text-lg { font-size: 1.125rem; }
.text-xl { font-size: 1.25rem; }
.text-2xl { font-size: 1.5rem; }

.font-bold { font-weight: 700; }
.font-medium { font-weight: 500; }
.font-normal { font-weight: 400; }

.text-primary { color: var(--text) !important; }
.text-secondary { color: var(--text-light) !important; }
.text-success { color: var(--success); }
.text-warning { color: var(--warning); }
.text-danger { color: var(--danger); }
.text-info { color: var(--info); }
.text-light { color: var(--text-light) !important; }
.text-lighter { color: var(--text-lighter) !important; }

.bg-primary { background-color: var(--primary); }
.bg-primary-light { background-color: var(--primary-light); }
.bg-success-light { background-color: var(--success-light); }
.bg-warning-light { background-color: var(--warning-light); }
.bg-danger-light { background-color: var(--danger-light); }
.bg-info-light { background-color: var(--info-light); }

.rounded { border-radius: var(--border-radius); }
.shadow { box-shadow: var(--box-shadow); }
.shadow-hover { box-shadow: var(--box-shadow-hover); }
```

```
/* Анимации */
.fade-in {
    animation: fadeIn 0.3s ease-in;
}

@keyframes fadeIn {
    from { opacity: 0; }
    to { opacity: 1; }
}

.slide-in {
    animation: slideIn 0.3s ease-out;
}

@keyframes slideIn {
    from { transform: translateY(20px); opacity: 0; }
    to { transform: translateY(0); opacity: 1; }
}

/* Адаптивность */
@media (max-width: 768px) {
    h1 {
        font-size: 2rem;
    }

    h2 {
        font-size: 1.75rem;
    }

    .container {
        padding: 0 1rem;
    }

    .grid {
        grid-template-columns: 1fr;
    }

    .hidden-sm {
        display: none;
    }
}
```

```
/* Кастомные стили для компонентов */

.badge {
    display: inline-block;
    padding: 0.25rem 0.75rem;
    border-radius: 50px;
    font-size: 0.75rem;
    font-weight: 600;
    text-transform: uppercase;
    letter-spacing: 0.5px;
    color: var(--text) !important;
}

.badge-primary {
    background-color: var(--primary-light);
    color: var(--primary-dark);
}

.badge-success {
    background-color: var(--success-light);
    color: var(--success);
}

.badge-warning {
    background-color: var(--warning-light);
    color: var(--text);
}

.badge-danger {
    background-color: var(--danger-light);
    color: var(--danger);
}

.badge-info {
    background-color: var(--info-light);
    color: var(--info);
}

/* Уведомления */

.alert {
    padding: 1rem 1.5rem;
    border-radius: var(--border-radius);
```

```
margin-bottom: 1.5rem;
border-left: 4px solid transparent;
color: var(--text) !important;
}

.alert-success {
background-color: var(--success-light);
border-left-color: var(--success);
color: var(--success);
}

.alert-warning {
background-color: var(--warning-light);
border-left-color: var(--warning);
color: var(--text);
}

.alert-danger {
background-color: var(--danger-light);
border-left-color: var(--danger);
color: var(--danger);
}

.alert-info {
background-color: var(--info-light);
border-left-color: var(--info);
color: var(--info);
}

/* Таблицы */
table {
width: 100%;
border-collapse: collapse;
margin-bottom: 1.5rem;
border-radius: var(--border-radius);
overflow: hidden;
}

th, td {
padding: 0.75rem 1rem;
text-align: left;
border-bottom: 1px solid #eeeeee;
```

```
color: var(--text) !important;
}

th {
background-color: var(--card-bg);
font-weight: 600;
color: var(--text) !important;
}

tr:hover {
background-color: var(--card-hover);
}

tr:last-child td {
border-bottom: none;
}

/* Разное */

.divider {
height: 1px;
background-color: #eeeeee;
margin: 1.5rem 0;
}

.loader {
display: inline-block;
width: 25px;
height: 25px;
border: 3px solid rgba(0, 0, 0, 0.1);
border-radius: 50%;
border-top-color: var(--primary);
animation: spin 1s ease-in-out infinite;
}

@keyframes spin {
to { transform: rotate(360deg); }
}

/* Скрытие скролл-бара но сохранение функциональности */

.scrollbar-hide {
-ms-overflow-style: none; /* IE and Edge */
scrollbar-width: none; /* Firefox */
}
```

```
}

.scrollbar-hide::-webkit-scrollbar {
    display: none; /* Chrome, Safari, Opera */
}

/* Глобальные стили для текста */
*, *::before, *::after {
    color: var(--text) !important;
}

body {
    color: var(--text);
    background-color: var(--dark-purple);
}

/* Обновляем цвета заголовков */
h1, h2, h3, h4, h5, h6 {
    color: var(--text) !important;
}

h1 {
    color: var(--text) !important;
}

/* Обновляем цвета ссылок */
a {
    color: var(--text) !important;
}

a:hover {
    color: var(--primary-light) !important;
}

/* Обновляем цвета для форм */
input::placeholder,
textarea::placeholder {
    color: var(--text-lighter) !important;
}

/* Обновляем цвета для выпадающих списков */
select option {
    background-color: var(--card-bg);
```

```
color: var(--text) !important;  
}  
  
/* Добавляем стили для компонентов header */  
.user-info {  
    display: flex;  
    align-items: center;  
    gap: 1.5rem;  
}  
  
.user-details {  
    display: flex;  
    flex-direction: column;  
    align-items: flex-end;  
}  
  
.user-name {  
    font-weight: 500;  
    color: var(--text-light) !important;  
}  
  
.user-role {  
    font-size: 0.85rem;  
    color: var(--text-secondary) !important;  
}  
  
.logout-form {  
    margin: 0;  
}  
  
.logout-btn {  
    display: inline-flex;  
    align-items: center;  
    gap: 0.5rem;  
    padding: 0.5rem 1rem;  
    background-color: transparent;  
    border: 1px solid rgba(255, 255, 255, 0.1);  
    border-radius: var(--border-radius);  
    color: var(--text-light) !important;  
    font-size: 0.9rem;  
    cursor: pointer;  
    transition: all 0.3s ease;
```

```
}
```

```
.logout-btn:hover {  
    background-color: rgba(255, 255, 255, 0.1);  
    transform: translateY(-2px);  
}
```

```
.logout-btn i {  
    font-size: 1rem;  
    color: var(--text-light) !important;  
}
```

## CreateTest.css

```
/* Темная пурпурная тема для страницы теста */  
  
:root {  
    --dark-purple: #1a1225;  
    --medium-purple: #7743DB;  
    --light-purple: #B39DDB;  
    --accent-purple: #9D4EDD;  
    --text-light: #ffffff;  
    --text-secondary: rgba(255, 255, 255, 0.85);  
    --card-bg: #272134;  
    --card-hover: #362b48;  
    --gradient-bg: linear-gradient(135deg, #1a1225 0%, #3a1d69 100%);  
    --shadow-sm: 0 4px 6px rgba(0, 0, 0, 0.3);  
    --shadow-md: 0 6px 12px rgba(0, 0, 0, 0.4);  
    --shadow-lg: 0 10px 25px rgba(0, 0, 0, 0.5);  
    --transition-slow: all 0.4s cubic-bezier(0.165, 0.84, 0.44, 1);  
    --transition-fast: all 0.2s ease;  
    --border-radius: 12px;  
    --card-spacing: 1.75rem;  
  
    /* Функциональные цвета */  
    --success: #4CAF50;  
    --success-dark: #388E3C;  
    --success-light: rgba(76, 175, 80, 0.15);  
    --danger: #F44336;  
    --danger-dark: #D32F2F;  
    --danger-light: rgba(244, 67, 54, 0.15);  
    --warning: #FFC107;  
    --warning-dark: #FFA000;  
    --warning-light: rgba(255, 193, 7, 0.15);  
    --info: #2196F3;
```

```
--info-dark: #1976D2;
--info-light: rgba(33, 150, 243, 0.15);
}

/* Принудительное переопределение базовых стилей */
html {
    background-color: var(--dark-purple) !important;
}

body {
    background-color: var(--dark-purple) !important;
    color: var(--text-light) !important;
    font-family: 'Roboto', sans-serif;
    margin: 0;
    padding: 0;
    min-height: 100vh;
    width: 100%;
}

.container {
    max-width: 1200px;
    margin: 2rem auto;
    padding: 0 2rem;
    background-color: transparent !important;
    box-shadow: none !important;
}

/* Заголовок страницы */
h1 {
    color: var(--text-light) !important;
    text-align: center;
    margin-bottom: 2rem;
    font-size: 2.5rem;
    font-weight: 700;
    text-shadow: 0 2px 4px rgba(0, 0, 0, 0.3);
    position: relative;
    z-index: 1;
}

/* Форма создания теста */
.test-form {
    display: flex;
```

```
flex-direction: column;
gap: 20px;
background: var(--gradient-bg);
border-radius: var(--border-radius);
padding: 2rem;
box-shadow: var(--shadow-md);
position: relative;
overflow: hidden;
animation: fadeIn 0.3s ease-out;
z-index: 0;
}
```

```
.test-form::before {
  content: "";
  position: absolute;
  top: 0;
  left: 0;
  right: 0;
  bottom: 0;
  background: url('data:image/svg+xml;utf8,<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 100 100" preserveAspectRatio="none"><path fill="none" stroke="rgba(255,255,255,0.05)" stroke-width="1" d="M0,0 L100,100 M100,0 L0,100"/></svg>');
  background-size: 20px 20px;
  opacity: 0.3;
  z-index: -1;
}
```

```
.test-info-section,
.grading-section {
  background-color: var(--card-bg);
  border-radius: var(--border-radius);
  padding: 2rem;
  margin-bottom: 2rem;
  border: 1px solid rgba(255, 255, 255, 0.1);
}
```

```
.section-title {
  color: var(--text-light);
  font-size: 1.4rem;
  font-weight: 600;
  margin-bottom: 1.5rem;
  padding-bottom: 1rem;
```

```
border-bottom: 2px solid var(--accent-purple);  
}  
  
/* Обновляем стили для основных полей формы */  
.test-info-section .form-group {  
    position: relative;  
    margin-bottom: 1.5rem;  
    display: flex;  
    flex-direction: column;  
    gap: 0.5rem;  
    background: transparent !important; /* Убираем фоновый цвет */  
    padding: 0 !important; /* Убираем лишние отступы */  
    border: none !important; /* Убираем границу */  
}  
  
.test-info-section .form-group label {  
    color: var(--text-light) !important;  
    font-size: 1.1rem;  
    font-weight: 500;  
    margin-bottom: 0.5rem;  
}  
  
.test-info-section .form-group input[type="text"],  
.test-info-section .form-group input[type="number"],  
.test-info-section .form-group textarea {  
    width: 100%;  
    padding: 0.8rem 1rem;  
    background-color: rgba(0, 0, 0, 0.2) !important;  
    border: 1px solid rgba(255, 255, 255, 0.1);  
    border-radius: var(--border-radius);  
    color: var(--text-light) !important;  
    font-size: 1rem;  
    z-index: 1;  
}  
  
/* Обновляем стили для критериев оценивания */  
.grading-section {  
    position: relative;  
    z-index: 1;  
}  
  
.grading-criteria-container {
```

```
display: flex;
flex-direction: column;
gap: 1rem;
width: 100%;

}

.grading-row {
  display: flex;
  align-items: center;
  padding: 1.5rem;
  background-color: rgba(0, 0, 0, 0.2);
  border-radius: var(--border-radius);
  border: 1px solid rgba(255, 255, 255, 0.1);
  margin-bottom: 1rem;
}

.grade-label {
  min-width: 120px;
  font-weight: 600;
  color: var(--text-light);
  margin-right: 2rem;
}

.points-range {
  display: flex;
  align-items: center;
  gap: 1rem;
  flex-grow: 1;
}

.range-input {
  display: flex;
  align-items: center;
  gap: 1rem;
  flex-grow: 1;
}

.range-input input[type="number"] {
  width: 100px;
  padding: 0.8rem;
  text-align: center;
  background-color: rgba(0, 0, 0, 0.2);
```

```
border: 1px solid rgba(255, 255, 255, 0.1);
border-radius: var(--border-radius);
color: var(--text-light);
font-size: 1rem;
}

.range-separator {
color: var(--text-light);
font-weight: 500;
font-size: 1.2rem;
margin: 0 1rem;
}

.points-label {
color: var(--text-light);
margin-left: 1rem;
min-width: 60px;
}

/* Стили для фокуса полей ввода в критериях */
.range-input input[type="number"]:focus {
outline: none;
border-color: var(--accent-purple);
box-shadow: 0 0 0 2px rgba(157, 78, 221, 0.3);
background-color: rgba(0, 0, 0, 0.3);
}

/* Адаптивность для критериев оценивания */
@media (max-width: 768px) {
.grading-row {
flex-direction: column;
align-items: flex-start;
gap: 1rem;
padding: 1rem;
}

.grade-label {
margin-bottom: 0.5rem;
}

.points-range {
width: 100%;
```

```
flex-wrap: wrap;
}

.range-input {
  width: 100%;
  justify-content: space-between;
}

.range-input input[type="number"] {
  width: calc(50% - 1rem);
}

.points-label {
  width: 100%;
  text-align: center;
  margin-top: 0.5rem;
}

/* Убираем конфликтующие стили */
.form-group:hover {
  transform: none !important;
  background-color: transparent !important;
  box-shadow: none !important;
}

/* Обновляем стили фокуса */
.test-info-section .form-group input:focus,
.test-info-section .form-group textarea:focus,
.grading-section input:focus {
  outline: none;
  border-color: var(--accent-purple);
  box-shadow: 0 0 0 2px rgba(157, 78, 221, 0.3);
  background-color: rgba(0, 0, 0, 0.3) !important;
}

/* Обновленные стили для блока вопроса */
.question-block {
  background-color: var(--card-bg);
  border: 1px solid rgba(255, 255, 255, 0.1);
  border-radius: var(--border-radius);
  padding: 2rem;
```

```
margin-bottom: 2rem;  
box-shadow: var(--shadow-md);  
transition: var(--transition-fast);  
position: relative;  
-webkit-animation: fadeIn 0.3s ease-out;  
-moz-animation: fadeIn 0.3s ease-out;  
animation: fadeIn 0.3s ease-out;  
}
```

```
.question-block:hover {  
    transform: translateY(-3px);  
    background-color: var(--card-hover);  
    box-shadow: var(--shadow-lg);  
}
```

```
.question-block h3 {  
    color: var(--text-light) !important;  
    margin: 0 0 1.5rem 0;  
    font-size: 1.4rem;  
    font-weight: 600;  
    padding-bottom: 1rem;  
    border-bottom: 2px solid var(--accent-purple);  
    display: flex;  
    align-items: center;  
    justify-content: space-between;  
}
```

/\* Стили для формы внутри вопроса \*/

```
.question-block .form-group {  
    background-color: rgba(0, 0, 0, 0.2);  
    padding: 1.5rem;  
    border-radius: var(--border-radius);  
    margin-bottom: 1.5rem;  
    border: 1px solid rgba(255, 255, 255, 0.05);  
    transition: all 0.2s ease;  
}
```

```
.question-block .form-group label {  
    display: block;  
    margin-bottom: 0.8rem;  
    color: var(--text-light) !important;  
    font-weight: 500;
```

```
    font-size: 1.1rem;
}

.question-block input[type="text"],
.question-block input[type="number"],
.question-block textarea,
.question-block select {
    width: 100%;
    padding: 0.8rem 1rem;
    background-color: rgba(0, 0, 0, 0.3);
    border: 1px solid rgba(255, 255, 255, 0.1);
    border-radius: 8px;
    color: var(--text-light) !important;
    font-size: 1rem;
    transition: all 0.2s ease;
}

.question-block input[type="text"]:focus,
.question-block input[type="number"]:focus,
.question-block textarea:focus,
.question-block select:focus {
    border-color: var(--accent-purple);
    box-shadow: 0 0 0 2px rgba(157, 78, 221, 0.3);
    outline: none;
}

/* Стили для группы ответов */
.answers-group {
    display: flex;
    flex-direction: column;
    gap: 1rem;
}

/* Стили для контейнера варианта ответа */
.answers-group .form-group {
    position: relative;
    display: flex;
    flex-direction: row;
    align-items: center;
    padding: 1rem;
    gap: 1rem;
    margin-bottom: 0.5rem;
```

```
background-color: rgba(0, 0, 0, 0.2);
border-radius: var(--border-radius);
border: 1px solid rgba(255, 255, 255, 0.05);
}

/* Стили для текстового поля ответа */
.answers-group .form-group input[type="text"] {
flex: 1;
width: calc(100% - 60px); /* Оставляем место для радио/чекбокса */
margin-right: 50px; /* Место для радио/чекбокса */
}

/* Стили для радио и чекбоксов */
.answers-group .form-group input[type="radio"],
.answers-group .form-group input[type="checkbox"] {
appearance: none;
-webkit-appearance: none;
width: 24px;
height: 24px;
border: 2px solid var(--accent-purple);
background-color: rgba(0, 0, 0, 0.3);
cursor: pointer;
position: absolute;
right: 1rem;
top: 50%;
transform: translateY(-50%);
margin: 0;
padding: 0;
}

/* Стили для радиокнопок */
.answers-group .form-group input[type="radio"] {
border-radius: 50%;
}

/* Стили для чекбоксов */
.answers-group .form-group input[type="checkbox"] {
border-radius: 4px;
}

/* Стили для отмеченного состояния */
.answers-group .form-group input[type="radio"]:checked,
```

```
.answers-group .form-group input[type="checkbox"]:checked {  
    background-color: var(--accent-purple);  
    border-color: var(--light-purple);  
    box-shadow: 0 0 8px rgba(157, 78, 221, 0.6);  
}  
  
/* Добавляем галочку для чекбокса */  
.answers-group .form-group input[type="checkbox"]:checked::after {  
    content: "✓";  
    position: absolute;  
    color: white;  
    font-size: 16px;  
    font-weight: bold;  
    top: 50%;  
    left: 50%;  
    transform: translate(-50%, -50%);  
}  
  
/* Добавляем точку для радио */  
.answers-group .form-group input[type="radio"]:checked::after {  
    content: "";  
    position: absolute;  
    width: 12px;  
    height: 12px;  
    border-radius: 50%;  
    background-color: white;  
    top: 50%;  
    left: 50%;  
    transform: translate(-50%, -50%);  
}  
  
/* Стили для метки варианта ответа */  
.answers-group .form-group label {  
    min-width: 100px;  
    font-weight: 500;  
}  
  
/* Эффект при наведении на вариант ответа */  
.answers-group .form-group:hover {  
    background-color: rgba(157, 78, 221, 0.15);  
    transform: translateX(5px);  
    transition: all 0.2s ease;
```

```
}

/* Стили фокуса для радио и чекбоксов */
.answers-group .form-group input[type="radio"]:focus,
.answers-group .form-group input[type="checkbox"]:focus {
    outline: none;
    box-shadow: 0 0 0 3px rgba(157, 78, 221, 0.4);
}

/* Адаптивность для мобильных устройств */
@media (max-width: 768px) {
    .answers-group .form-group {
        padding: 0.8rem;
    }

    .answers-group .form-group input[type="text"] {
        width: calc(100% - 40px);
        margin-right: 35px;
    }

    .answers-group .form-group input[type="radio"],
    .answers-group .form-group input[type="checkbox"] {
        width: 20px;
        height: 20px;
        right: 0.8rem;
    }
}

/* Обновленные стили для кнопок */
.add-question-btn,
.submit-btn,
.remove-btn {
    display: inline-block;
    padding: 1rem 2rem;
    border: none;
    border-radius: var(--border-radius);
    font-size: 1.1rem;
    font-weight: 600;
    cursor: pointer;
    transition: all 0.3s ease;
    text-align: center;
    margin: 1rem 0;
}
```

```
color: var(--text-light) !important;
box-shadow: var(--shadow-sm);
position: relative;
overflow: hidden;
z-index: 1;
}

.add-question-btn {
background-color: var(--accent-purple);
width: 100%;
}

.submit-btn {
background-color: var(--success);
width: 100%;
margin-top: 2rem;
}

.remove-btn {
background-color: var(--danger);
position: absolute;
top: 1rem;
right: 1rem;
padding: 0.8rem 1.5rem;
font-size: 0.9rem;
}

/* Эффекты при наведении */
.add-question-btn:hover,
.submit-btn:hover,
.remove-btn:hover {
transform: translateY(-3px);
box-shadow: var(--shadow-md);
opacity: 0.9;
}

/* Эффекты при нажатии */
.add-question-btn:active,
.submit-btn:active,
.remove-btn:active {
transform: translateY(1px);
box-shadow: var(--shadow-sm);
```

```
}

/* Состояние фокуса */
.add-question-btn:focus,
.submit-btn:focus,
.remove-btn:focus {
    outline: none;
    box-shadow: 0 0 0 3px rgba(157, 78, 221, 0.4);
}

/* Отключенное состояние */
.add-question-btn:disabled,
.submit-btn:disabled,
.remove-btn:disabled {
    background-color: var(--card-bg);
    cursor: not-allowed;
    opacity: 0.7;
}

/* Стили для установления порядка */
.order-number {
    display: inline-flex;
    align-items: center;
    justify-content: center;
    width: 24px;
    height: 24px;
    background-color: var(--accent-purple);
    color: white;
    border-radius: 50%;
    margin-right: 1rem;
}

/* Адаптивность */
@media (max-width: 768px) {
    .question-block {
        padding: 1.5rem;
        box-shadow: var(--shadow-sm);
    }
}

.matching-pair {
    grid-template-columns: 1fr;
}
```

```
.remove-btn {  
    position: relative;  
    top: auto;  
    right: auto;  
    width: 100%;  
    margin-top: 1rem;  
}  
  
.container {  
    padding: 0 1rem;  
}  
  
h1 {  
    font-size: 2rem;  
}  
  
.test-form {  
    padding: 1.5rem;  
}  
  
.grading-row {  
    flex-direction: column;  
    align-items: stretch;  
}  
  
.points-range {  
    flex-wrap: wrap;  
}  
}  
  
#gradingCriteria {  
    font-family: monospace;  
    white-space: pre-wrap;  
}  
  
/* Анимации */  
@keyframes fadeIn {  
    from { opacity: 0; transform: translateY(10px); }  
    to { opacity: 1; transform: translateY(0); }  
}
```

```
.question-block {  
    animation: fadeIn 0.3s ease-out;  
}  
  
/* Стили для состояния наведения и фокуса */  
.form-group:focus-within label {  
    color: #740978;  
}  
  
.error-message {  
    display: none;  
    color: #fff;  
    background-color: var(--danger-light);  
    border: 1px solid var(--danger);  
    border-radius: var(--border-radius);  
    padding: 1rem;  
    margin-top: 1rem;  
    font-size: 0.9em;  
}  
  
/* Удаляем старые стили, которые больше не нужны */  
.remove-criteria,  
.add-criteria {  
    display: none;  
}  
  
/* Анимация появления */  
.criterion-group {  
    animation: slideIn 0.3s ease-out forwards;  
    opacity: 0;  
    transform: translateY(20px);  
    background-color: rgba(0, 0, 0, 0.2);  
    color: #fff;  
}  
  
@keyframes slideIn {  
    to {  
        opacity: 1;  
        transform: translateY(0);  
    }  
}
```

```

/* Добавляем задержку для последовательной анимации */
.criterion-group:nth-child(1) { animation-delay: 0.1s; }
.criterion-group:nth-child(2) { animation-delay: 0.2s; }
.criterion-group:nth-child(3) { animation-delay: 0.3s; }

/* Обновляем стили для состояния фокуса */
.form-group:focus-within label {
    color: #fff !important;
}

/* Обновляем стили для placeholder */
input::placeholder,
textarea::placeholder {
    color: rgba(255, 255, 255, 0.5) !important;
}

/* Обновляем стили для текста внутри вопросов */
.question-block input[type="text"],
.question-block input[type="number"],
.question-block textarea,
.question-block select {
    color: #fff !important;
}

/* Обновляем стили для меток радио и чекбоксов */
.form-group label {
    color: #fff !important;
}

/* Обновляем стили для текста в сопоставлениях */
.matching-pair input {
    color: #fff !important;
    background-color: rgba(0, 0, 0, 0.2);
}

/* Обновляем стили для текста в установке порядка */
.order-item {
    color: #fff !important;
}

/* Принудительно устанавливаем белый цвет для всех текстовых элементов */
*, *::before, *::after {

```

```
color: #fff !important;
}

/* Обновляем стили для всех заголовков */
h1, h2, h3, h4, h5, h6 {
    color: #fff !important;
}

/* Обновляем стили для всех параграфов */
p {
    color: #fff !important;
}

/* Обновляем стили для всех span элементов */
span {
    color: #fff !important;
}

/* Обновляем стили для всех input элементов */
input {
    color: #fff !important;
}

/* Обновляем стили для всех select элементов */
select {
    color: #fff !important;
}

/* Обновляем стили для всех textarea элементов */
textarea {
    width: 100%;
    padding: 12px;
    border: 1px solid rgba(255, 255, 255, 0.1);
    border-radius: var(--border-radius);
    background-color: rgba(0, 0, 0, 0.2) !important;
    color: #fff !important;
    font-size: 1rem;
    transition: var(--transition-fast);
    resize: vertical;
    min-height: 120px;
}
```

```
textarea:focus {
    outline: none;
    border-color: var(--accent-purple);
    box-shadow: 0 0 0 3px rgba(157, 78, 221, 0.3);
    background-color: rgba(0, 0, 0, 0.3) !important;
}

/* Принудительно переопределяем стили для textarea в форме теста */
.test-form textarea {
    background-color: rgba(0, 0, 0, 0.2) !important;
    color: #fff !important;
    border: 1px solid rgba(255, 255, 255, 0.1);
}

.test-form textarea:focus {
    background-color: rgba(0, 0, 0, 0.3) !important;
}

/* Обновляем стили для всех полей ввода в форме теста */
.test-form input,
.test-form textarea,
.test-form select {
    background-color: rgba(0, 0, 0, 0.2) !important;
    color: #fff !important;
}

/* Стили для placeholder в textarea */
textarea::placeholder {
    color: rgba(255, 255, 255, 0.5) !important;
}

/* Обновляем стили для всех label элементов */
label {
    color: #fff !important;
}

select option {
    background-color: var(--card-bg);
    color: #fff;
}

/* Улучшение доступности */
```

```

.form-group input:focus,
.form-group textarea:focus,
.form-group select:focus {
    outline: 2px solid var(--accent-purple);
    outline-offset: 2px;
}

/* Улучшение контрастности */
.form-group label {
    color: var(--text-light) !important;
    font-weight: 600;
}

.question-header {
    display: flex;
    justify-content: space-between;
    align-items: center;
    margin-bottom: 1rem;
}

.question-header h3 {
    margin: 0;
}

.question-number {
    color: var(--text);
    font-weight: bold;
}

```

## GroupCRUD.css

```

/* GroupCRUD.css */

:root {
    --dark-purple: #1a1225;
    --medium-purple: #7743DB;
    --light-purple: #B39DDB;
    --accent-purple: #9D4EDD;
    --text-light: #ffffff;
    --text-secondary: rgba(255, 255, 255, 0.85);
    --card-bg: #272134;
    --card-hover: #362b48;
    --gradient-bg: linear-gradient(135deg, #1a1225 0%, #3a1d69 100%);
    --shadow-sm: 0 4px 6px rgba(0, 0, 0, 0.3);
    --shadow-md: 0 6px 12px rgba(0, 0, 0, 0.4);
}

```

```

--shadow-lg: 0 10px 25px rgba(0, 0, 0, 0.5);
--border-radius: 12px;
}

/* Глобальные стили */
*, *::before, *::after {
    color: var(--text-light) !important;
}

body {
    font-family: 'Roboto', sans-serif;
    background-color: var(--dark-purple) !important;
    margin: 0;
    min-height: 100vh;
}

.container {
    max-width: 1200px;
    margin: 2rem auto;
    padding: 2rem;
    background: var(--gradient-bg) !important;
    border-radius: var(--border-radius);
    box-shadow: var(--shadow-md);
    position: relative;
    overflow: hidden;
    animation: fadeIn 0.5s ease-out forwards;
}

.container::before {
    content: "";
    position: absolute;
    top: 0;
    left: 0;
    right: 0;
    bottom: 0;
    background: url('data:image/svg+xml;utf8,<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 100 100" preserveAspectRatio="none"><path fill="none" stroke="rgba(255,255,255,0.05)" stroke-width="1" d="M0,0 L100,100 M100,0 L0,100"/></svg>');
    background-size: 20px 20px;
    opacity: 0.3;
    z-index: 0;
}

```

```
h1, h2 {  
    text-align: center;  
    color: var(--text-light) !important;  
    font-weight: 700;  
    margin-bottom: 2rem;  
    text-shadow: 0 2px 4px rgba(0, 0, 0, 0.3);  
    position: relative;  
    z-index: 1;  
}  
  
h1 {  
    font-size: 2.5rem;  
}
```

```
h2 {  
    font-size: 2rem;  
}
```

```
/* Форма создания группы */
```

```
.crud-form {  
    display: flex;  
    flex-direction: column;  
    gap: 1rem;  
    position: relative;  
    z-index: 1;  
    max-width: 600px;  
    margin: 0 auto 2rem;  
}
```

```
.crud-form input {  
    padding: 1rem;  
    background-color: var(--card-bg);  
    border: 1px solid rgba(255, 255, 255, 0.1);  
    border-radius: var(--border-radius);  
    font-size: 1rem;  
    transition: all 0.3s ease;  
}
```

```
.crud-form input:focus {  
    outline: none;  
    border-color: var(--accent-purple);
```

```
    box-shadow: 0 0 0 2px rgba(157, 78, 221, 0.3);
}

.crud-form button,
.delete-button {
    padding: 1rem;
    background-color: var(--medium-purple);
    color: var(--text-light);
    border: none;
    border-radius: var(--border-radius);
    font-size: 1rem;
    font-weight: 500;
    cursor: pointer;
    transition: all 0.3s ease;
}

.crud-form button:hover {
    background-color: var(--accent-purple);
    transform: translateY(-2px);
    box-shadow: var(--shadow-md);
}

.delete-button {
    background-color: #dc3545;
}

.delete-button:hover {
    background-color: #c82333;
    transform: translateY(-2px);
    box-shadow: var(--shadow-md);
}

/* Список групп */
.group-list {
    display: grid;
    grid-template-columns: repeat(auto-fill, minmax(300px, 1fr));
    gap: 1.5rem;
    position: relative;
    z-index: 1;
}

.group-card {
```

```
background-color: var(--card-bg);
border: 1px solid rgba(255, 255, 255, 0.1);
border-radius: var(--border-radius);
padding: 1.5rem;
box-shadow: var(--shadow-sm);
transition: all 0.3s ease;
}
```

```
.group-card:hover {
    transform: translateY(-5px);
    box-shadow: var(--shadow-lg);
    background-color: var(--card-hover);
}
```

```
.group-info {
    margin-bottom: 1.5rem;
}
```

```
.group-info h3 {
    margin: 0;
    font-size: 1.2rem;
    font-weight: 500;
}
```

```
.crud-actions {
    display: flex;
    flex-direction: column;
    gap: 1rem;
}
```

```
/* Кнопка "Назад" */
.back-button {
    position: relative;
    display: inline-flex;
    align-items: center;
    gap: 0.75rem;
    padding: 0.75rem 1.5rem;
    background-color: var(--medium-purple);
    color: var(--text-light) !important;
    border-radius: var(--border-radius);
    text-decoration: none;
    font-weight: 500;
```

```
    transition: all 0.3s ease;
    margin-bottom: 1.5rem;
    border: 1px solid rgba(255, 255, 255, 0.1);
    box-shadow: var(--shadow-sm);
}
```

```
.back-button:hover {
    background-color: var(--accent-purple);
    transform: translateY(-2px);
    box-shadow: var(--shadow-md);
    text-decoration: none;
}
```

```
.back-button i {
    font-size: 1.1rem;
}
```

```
/* Анимация появления */
@keyframes fadeIn {
    from {
        opacity: 0;
        transform: translateY(20px);
    }
    to {
        opacity: 1;
        transform: translateY(0);
    }
}
```

```
/* Адаптивность */
@media (max-width: 768px) {
    .container {
        margin: 1rem;
        padding: 1.5rem;
    }
}
```

```
h1 {
    font-size: 2rem;
}
```

```
h2 {
    font-size: 1.5rem;
```

```
    }

    .group-list {
        grid-template-columns: 1fr;
    }
}

/* Обновляем стили для секций */

.header-section {
    position: relative;
    margin-bottom: 2rem;
}

.create-section {
    margin-bottom: 2rem;
}

.groups-section {
    position: relative;
}

/* Обновляем стили для форм */

.input-group {
    display: flex;
    gap: 1rem;
    width: 100%;
}

.input-group input {
    flex: 1;
    min-width: 0;
}

.input-group button {
    white-space: nowrap;
}

/* Обновляем стили для карточек групп */

.group-card {
    background-color: var(--card-bg);
    border: 1px solid rgba(255, 255, 255, 0.1);
    border-radius: var(--border-radius);
}
```

```
padding: 1.5rem;
box-shadow: var(--shadow-sm);
transition: all 0.3s ease;
}

.group-card:hover {
    transform: translateY(-5px);
    box-shadow: var(--shadow-lg);
    background-color: var(--card-hover);
}

.group-info {
    margin-bottom: 1.5rem;
}

.group-info h3 {
    margin: 0;
    font-size: 1.2rem;
    font-weight: 500;
    color: var(--text-light) !important;
}

/* Обновляем стили для форм действий */
.crud-form {
    margin-bottom: 1rem;
}

.delete-form {
    margin-top: 0.5rem;
}

.delete-button {
    width: 100%;
    background-color: #dc3545;
    color: var(--text-light) !important;
    border: none;
    padding: 0.75rem;
    border-radius: var(--border-radius);
    transition: all 0.3s ease;
}

.delete-button:hover {
```

```
background-color: #c82333;
transform: translateY(-2px);
}

/* Добавляем стили для заголовков */
h1, h2 {
    color: var(--text-light) !important;
    text-align: center;
    margin-bottom: 2rem;
}

h1 {
    font-size: 2.25rem;
    font-weight: 700;
}

h2 {
    font-size: 1.75rem;
    font-weight: 600;
    margin-top: 0;
}

/* Обновляем стили для input полей */
input {
    background-color: rgba(255, 255, 255, 0.1);
    border: 1px solid rgba(255, 255, 255, 0.2);
    color: var(--text-light) !important;
    padding: 0.75rem 1rem;
}

input:focus {
    outline: none;
    border-color: var(--accent-purple);
    box-shadow: 0 0 0 2px rgba(157, 78, 221, 0.3);
}

/* Добавляем отзывчивость */
@media (max-width: 768px) {
    .input-group {
        flex-direction: column;
    }
}
```

```
.input-group button {  
    width: 100%;  
}  
  
}
```

```
.container {  
    margin: 1rem;  
    padding: 1rem;  
}  
}
```

## LectionCRUD.css

```
/* Современный адаптивный стиль для страницы предмета - темная тема */
```

```
:root {  
    --dark-purple: #1a1225;  
    --medium-purple: #7743DB;  
    --light-purple: #B39DDB;  
    --accent-purple: #9D4EDD;  
    --text-light: #ffffff;  
    --text-secondary: rgba(255, 255, 255, 0.85);  
    --card-bg: #272134;  
    --card-hover: #362b48;  
    --gradient-bg: linear-gradient(135deg, #1a1225 0%, #3a1d69 100%);  
    --shadow-sm: 0 4px 6px rgba(0, 0, 0, 0.3);  
    --shadow-md: 0 6px 12px rgba(0, 0, 0, 0.4);  
    --shadow-lg: 0 10px 25px rgba(0, 0, 0, 0.5);  
    --transition-slow: all 0.4s cubic-bezier(0.165, 0.84, 0.44, 1);  
    --transition-fast: all 0.2s ease;  
    --border-radius: 12px;  
    --card-spacing: 1.75rem;  
}
```

```
/* Принудительное переопределение базовых стилей */
```

```
html {  
    background-color: var(--dark-purple) !important;  
}
```

```
body {  
    background-color: var(--dark-purple) !important;  
    color: var(--text-light) !important;  
    font-family: 'Roboto', sans-serif;  
    margin: 0;  
    padding: 0;  
    min-height: 100vh;
```

```
width: 100%;  
overflow-x: hidden;  
}  
  
/* Основной контейнер страницы */  
.container {  
    max-width: 800px;  
    margin: 2rem auto;  
    padding: 2rem;  
    background-color: var(--card-bg);  
    border-radius: var(--border-radius);  
    box-shadow: var(--shadow-md);  
    border: 1px solid rgba(255, 255, 255, 0.05);  
}  
  
/* Заголовок */  
h1 {  
    font-size: 2.8rem;  
    font-weight: 700;  
    margin-bottom: 2rem;  
    color: white !important;  
    text-shadow: 0 2px 4px rgba(0, 0, 0, 0.3);  
    text-align: center;  
    background: var(--gradient-bg);  
    padding: 2rem;  
    border-radius: var(--border-radius);  
    box-shadow: var(--shadow-md);  
    position: relative;  
    overflow: hidden;  
}  
  
/* Форма */  
.form-group {  
    margin-bottom: 1.5rem;  
    background-color: var(--card-bg);  
    padding: 1.5rem;  
    border-radius: var(--border-radius);  
    border: 1px solid rgba(255, 255, 255, 0.05);  
    transition: var(--transition-fast);  
}  
  
.form-group:hover {
```

```
    box-shadow: var(--shadow-lg);
    transform: translateY(-2px);
}

label {
    display: block;
    margin-bottom: 0.8rem;
    color: var(--text-light);
    font-weight: 500;
    font-size: 1.1rem;
}

/* Обновленные стили для полей ввода */
input[type="text"],
input[type="url"],
textarea {
    width: 100%;
    padding: 0.8rem 1rem;
    background: linear-gradient(145deg, #2d1f3a, #231630);
    border: 1px solid rgba(255, 255, 255, 0.05);
    border-radius: var(--border-radius);
    color: var(--text-light);
    font-size: 1rem;
    transition: var(--transition-fast);
    box-shadow: var(--shadow-sm);
    box-sizing: border-box;
    margin: 0;
}

textarea {
    min-height: 150px;
    resize: vertical;
    line-height: 1.5;
    padding: 1rem;
}

input[type="text"]:focus,
input[type="url"]:focus,
textarea:focus {
    outline: none;
    border-color: var(--accent-purple);
    box-shadow: 0 0 0 2px rgba(157, 78, 221, 0.2);
```

```
        transform: translateY(-2px);
    }

/* Кнопка отправки */
.submit-button {
    display: inline-flex;
    align-items: center;
    gap: 0.5rem;
    width: 100%;
    padding: 1rem 1.2rem;
    background-color: var(--accent-purple);
    color: white !important;
    text-decoration: none;
    border-radius: var(--border-radius);
    font-size: 1.1rem;
    font-weight: 500;
    transition: all 0.3s ease;
    border: 1px solid rgba(255, 255, 255, 0.1);
    box-shadow: var(--shadow-sm);
    cursor: pointer;
    justify-content: center;
}

.submit-button:hover {
    background-color: var(--medium-purple);
    transform: translateY(-2px);
    box-shadow: var(--shadow-md);
}

.submit-button i {
    font-size: 1.1rem;
}

/* Сообщение об ошибке */
.error-message {
    color: #ff6b6b;
    text-align: center;
    margin-top: 1.5rem;
    padding: 1.5rem;
    background-color: rgba(255, 107, 107, 0.1);
    border-radius: var(--border-radius);
    border: 1px solid rgba(255, 107, 107, 0.2);
```

```
display: flex;
align-items: center;
justify-content: center;
gap: 0.5rem;
font-weight: 500;
}

/* Плейсхолдеры */
::placeholder {
    color: rgba(255, 255, 255, 0.4);
}

/* Стилизация скроллбара */
::-webkit-scrollbar {
    width: 8px;
}

::-webkit-scrollbar-track {
    background: var(--dark-purple);
}

::-webkit-scrollbar-thumb {
    background: var(--medium-purple);
    border-radius: 4px;
}

::-webkit-scrollbar-thumb:hover {
    background: var(--accent-purple);
}

/* Адаптивный дизайн */
@media (max-width: 992px) {
    .container {
        margin: 1rem;
    }

    h1 {
        font-size: 2.4rem;
        padding: 1.8rem;
    }
}
```

```

@media (max-width: 768px) {
    .container {
        padding: 1.5rem;
    }

    h1 {
        font-size: 2rem;
        padding: 1.5rem;
    }

    .form-group {
        padding: 1.2rem;
    }
}

@media (max-width: 576px) {
    .container {
        margin: 0.5rem;
        padding: 1rem;
    }

    h1 {
        font-size: 1.8rem;
        padding: 1.2rem;
    }

    .form-group {
        padding: 1rem;
    }

    input[type="text"],
    input[type="url"],
    textarea {
        padding: 0.8rem;
    }
}

```

## LectionMaterial.css

```

/* Современный адаптивный стиль для страницы лекции - темная тема */

:root {
    --dark-purple: #1a1225;
    --medium-purple: #7743DB;
    --light-purple: #B39DDB;
}

```

```
--accent-purple: #9D4EDD;  
--text-light: #ffffff;  
--text-secondary: rgba(255, 255, 255, 0.85);  
--card-bg: #272134;  
--card-hover: #362b48;  
--gradient-bg: linear-gradient(135deg, #1a1225 0%, #3a1d69 100%);  
--shadow-sm: 0 4px 6px rgba(0, 0, 0, 0.3);  
--shadow-md: 0 6px 12px rgba(0, 0, 0, 0.4);  
--border-radius: 12px;  
--danger: #dc3545;  
--danger-hover: #c82333;  
}
```

```
html, body {  
background-color: var(--dark-purple) !important;  
color: var(--text-light) !important;  
font-family: 'Roboto', sans-serif;  
margin: 0;  
padding: 0;  
min-height: 100vh;  
}
```

```
.container {  
max-width: 1200px;  
margin: 2rem auto;  
padding: 0 2rem;  
background-color: transparent !important;  
box-shadow: none !important;  
}
```

```
/* Заголовок */  
h1 {  
font-size: 2.5rem;  
font-weight: 700;  
text-align: center;  
color: var(--text-light) !important;  
margin-bottom: 2rem;  
padding: 1.5rem;  
background: var(--gradient-bg);  
border-radius: var(--border-radius);  
box-shadow: var(--shadow-md);  
}
```

```
/* Основной блок с информацией */
.details {
    background-color: var(--card-bg);
    border-radius: var(--border-radius);
    padding: 2rem;
    margin-bottom: 2rem;
    box-shadow: var(--shadow-md);
    border: 1px solid rgba(255, 255, 255, 0.05);
}

.details p {
    margin: 1rem 0;
    color: var(--text-light);
    font-size: 1.1rem;
}

.details strong {
    color: var(--accent-purple);
    font-weight: 600;
    margin-right: 0.5rem;
}

/* Информация о файле */
.file-info {
    background-color: rgba(255, 255, 255, 0.05);
    padding: 1.5rem;
    border-radius: var(--border-radius);
    margin: 1.5rem 0;
    display: flex;
    align-items: center;
    flex-wrap: wrap;
    gap: 1rem;
}

.file-link {
    color: var(--accent-purple);
    text-decoration: none;
    display: inline-flex;
    align-items: center;
    gap: 0.5rem;
    transition: all 0.3s ease;
```

```
    font-weight: 500;
}

.file-link:hover {
    color: var(--light-purple);
    transform: translateY(-2px);
}

.file-icon {
    font-size: 1.5rem;
}

/* Группа кнопок */
.button-group {
    display: flex;
    gap: 1rem;
    margin: 1rem 0 2rem;
    align-items: center;
    flex-wrap: wrap;
}

/* Кнопка "Назад" */
.back-button {
    display: inline-flex;
    align-items: center;
    gap: 0.5rem;
    padding: 0.7rem 1.2rem;
    background: var(--dark-purple);
    color: white !important;
    text-decoration: none;
    border-radius: var(--border-radius);
    font-size: 1rem;
    font-weight: 500;
    transition: all 0.3s ease;
    border: 1px solid rgba(255, 255, 255, 0.1);
    box-shadow: var(--shadow-sm);
}

.back-button:hover {
    background: var(--card-hover);
    transform: translateY(-2px);
    box-shadow: var(--shadow-md);
```

```
}
```

```
/* Кнопка редактирования */  
.edit-button {  
    display: inline-flex;  
    align-items: center;  
    gap: 0.5rem;  
    padding: 0.7rem 1.2rem;  
    background: var(--medium-purple);  
    color: white !important;  
    text-decoration: none;  
    border-radius: var(--border-radius);  
    font-size: 1rem;  
    font-weight: 500;  
    transition: all 0.3s ease;  
    border: 1px solid rgba(255, 255, 255, 0.1);  
    box-shadow: var(--shadow-sm);  
}  
  
}
```

```
.edit-button:hover {  
    background: var(--accent-purple);  
    transform: translateY(-2px);  
    box-shadow: var(--shadow-md);  
}  
  
}
```

```
/* Кнопка удаления */  
.delete-button {  
    display: inline-flex;  
    align-items: center;  
    gap: 0.5rem;  
    padding: 0.7rem 1.2rem;  
    background-color: var(--danger);  
    color: white !important;  
    border: none;  
    border-radius: var(--border-radius);  
    font-size: 1rem;  
    font-weight: 500;  
    cursor: pointer;  
    transition: all 0.3s ease;  
    text-decoration: none;  
    border: 1px solid rgba(255, 255, 255, 0.1);  
    box-shadow: var(--shadow-sm);  
}
```

```
}

.delete-button:hover {
    background-color: var(--danger-hover);
    transform: translateY(-2px);
    box-shadow: var(--shadow-md);
}

/* Адаптивность */
@media (max-width: 768px) {
    .container {
        padding: 0 1rem;
    }

    h1 {
        font-size: 2rem;
        padding: 1.2rem;
    }

    .details {
        padding: 1.5rem;
    }

    .button-group {
        flex-direction: column;
        width: 100%;
    }

    .back-button,
    .edit-button,
    .delete-button {
        width: 100%;
        justify-content: center;
    }
}

/* Стили для формы редактирования */
.edit-form-container {
    background-color: var(--card-bg);
    border-radius: var(--border-radius);
    padding: 2rem;
    margin-bottom: 2rem;
}
```

```
    box-shadow: var(--shadow-md);
    border: 1px solid rgba(255, 255, 255, 0.05);
}

.edit-form {
    display: flex;
    flex-direction: column;
    gap: 1.5rem;
}

.form-group {
    display: flex;
    flex-direction: column;
    gap: 0.5rem;
}

.form-group label {
    color: var(--text-light);
    font-size: 1.1rem;
    font-weight: 500;
    display: flex;
    align-items: center;
    gap: 0.5rem;
}

.form-group label i {
    color: var(--accent-purple);
    width: 20px;
}

.form-group input,
.form-group textarea {
    background-color: rgba(255, 255, 255, 0.05);
    border: 1px solid rgba(255, 255, 255, 0.1);
    border-radius: var(--border-radius);
    padding: 0.8rem 1rem;
    color: var(--text-light);
    font-size: 1rem;
    transition: all 0.3s ease;
    width: 100%;
    box-sizing: border-box;
}
```

```
.form-group textarea {
    min-height: 150px;
    resize: vertical;
    line-height: 1.5;
}

.form-group input:focus,
.form-group textarea:focus {
    outline: none;
    border-color: var(--accent-purple);
    box-shadow: 0 0 0 2px rgba(157, 78, 221, 0.2);
}

/* Кнопка сохранения */
.save-button {
    display: inline-flex;
    align-items: center;
    gap: 0.5rem;
    padding: 0.7rem 1.2rem;
    background: var(--medium-purple);
    color: white !important;
    border: none;
    border-radius: var(--border-radius);
    font-size: 1rem;
    font-weight: 500;
    cursor: pointer;
    transition: all 0.3s ease;
    text-decoration: none;
    border: 1px solid rgba(255, 255, 255, 0.1);
    box-shadow: var(--shadow-sm);
}

.save-button:hover {
    background: var(--accent-purple);
    transform: translateY(-2px);
    box-shadow: var(--shadow-md);
}

/* Сообщение об ошибке */
.error-message {
    background-color: rgba(220, 53, 69, 0.1);
```

```
color: #ff6b6b;
padding: 1rem 1.5rem;
border-radius: var(--border-radius);
margin-bottom: 1.5rem;
display: flex;
align-items: center;
gap: 0.5rem;
border-left: 4px solid #ff6b6b;
}

.error-message i {
color: #ff6b6b;
}

/* Адаптивность для формы */
@media (max-width: 768px) {
.edit-form-container {
padding: 1.5rem;
}

.form-group input,
.form-group textarea {
font-size: 16px; /* Предотвращает зум на iOS */
}

.button-group {
flex-direction: column;
width: 100%;
}

.save-button {
width: 100%;
justify-content: center;
}
}
```

## Logreg.css

```
/* logreg.css */
:root {
--dark-purple: #1a1225;
--medium-purple: #7743DB;
--light-purple: #B39DDB;
--accent-purple: #9D4EDD;
```

```
--text-light: #ffffff;  
--text-secondary: rgba(255, 255, 255, 0.85);  
--card-bg: #272134;  
--card-hover: #362b48;  
--gradient-bg: linear-gradient(135deg, #1a1225 0%, #3a1d69 100%);  
--shadow-sm: 0 4px 6px rgba(0, 0, 0, 0.3);  
--shadow-md: 0 6px 12px rgba(0, 0, 0, 0.4);  
--shadow-lg: 0 10px 25px rgba(0, 0, 0, 0.5);  
--border-radius: 12px;  
}  
  
body {
```

```
    font-family: 'Roboto', sans-serif;  
    margin: 0;  
    padding: 0;  
    min-height: 100vh;  
    display: flex;  
    justify-content: center;  
    align-items: center;  
    background-color: var(--dark-purple);  
    color: var(--text-light);  
}
```

```
.auth-container {  
    width: 100%;  
    max-width: 450px;  
    margin: 2rem;  
    padding: 2.5rem;  
    background: var(--gradient-bg);  
    border-radius: var(--border-radius);  
    box-shadow: var(--shadow-lg);  
    position: relative;  
    overflow: hidden;  
    animation: fadeIn 0.5s ease-out;  
    box-sizing: border-box;  
    z-index: 1;  
}
```

```
.auth-container::before {  
    content: "";  
    position: absolute;  
    top: 0;
```

```
    left: 0;
    right: 0;
    bottom: 0;
    background: url('data:image/svg+xml;utf8,<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 100 100" preserveAspectRatio="none"><path fill="none" stroke="rgba(255,255,255,0.05)" stroke-width="1" d="M0,0 L100,100 M100,0 L0,100"/></svg>');
    background-size: 20px 20px;
    opacity: 0.3;
    z-index: 0;
    pointer-events: none;
}

.header {
    text-align: center;
    margin-bottom: 2rem;
    position: relative;
    z-index: 1;
}

.logo {
    font-size: 2.5rem;
    font-weight: 700;
    color: var(--text-light);
    margin-bottom: 1rem;
    display: flex;
    align-items: center;
    justify-content: center;
}

.logo i {
    margin-right: 0.75rem;
    font-size: 2.75rem;
}

h1 {
    color: var(--text-light);
    font-size: 2rem;
    margin-bottom: 1.5rem;
    text-align: center;
}

.subtitle {
```

```
    color: var(--text-secondary);  
    font-size: 1.1rem;  
    margin-bottom: 2rem;  
}
```

```
form {  
  position: relative;  
  z-index: 2;  
  display: flex;  
  flex-direction: column;  
  gap: 1.5rem;  
}
```

```
.form-group {  
  margin-bottom: 1rem;  
  width: 100%;  
}
```

```
label {  
  display: block;  
  margin-bottom: 0.5rem;  
  color: var(--text-light);  
  font-weight: 500;  
}
```

```
.input-group {  
  position: relative;  
  width: 100%;  
}
```

```
.input-group i {  
  position: absolute;  
  left: 1rem;  
  top: 50%;  
  transform: translateY(-50%);  
  color: var(--text-secondary);  
  font-size: 1.1rem;  
}
```

```
input[type="text"],  
input[type="password"],  
input[type="email"] {
```

```
width: 100%;  
padding: 0.75rem 1rem 0.75rem 2.75rem;  
background-color: rgba(255, 255, 255, 0.1);  
border: 1px solid rgba(255, 255, 255, 0.2);  
border-radius: var(--border-radius);  
color: var(--text-light);  
font-size: 1rem;  
transition: all 0.3s ease;  
box-sizing: border-box;  
}
```

```
input:focus {  
outline: none;  
border-color: var(--accent-purple);  
box-shadow: 0 0 0 2px rgba(157, 78, 221, 0.2);  
background-color: rgba(255, 255, 255, 0.15);  
}
```

```
input::placeholder {  
color: var(--text-secondary);  
}
```

```
button {  
width: 100%;  
padding: 1rem;  
background-color: var(--medium-purple);  
color: var(--text-light);  
border: none;  
border-radius: var(--border-radius);  
font-size: 1.1rem;  
font-weight: 500;  
cursor: pointer;  
transition: all 0.3s ease;  
display: flex;  
align-items: center;  
justify-content: center;  
gap: 0.75rem;  
}
```

```
button:hover {  
background-color: var(--accent-purple);  
transform: translateY(-2px);
```

```
    box-shadow: var(--shadow-md);  
}  
  
button i {  
    font-size: 1.2rem;  
}  
  
.divider {  
    display: flex;  
    align-items: center;  
    text-align: center;  
    margin: 2rem 0;  
    color: var(--text-secondary);  
}  
  
.divider::before,  
.divider::after {  
    content: "";  
    flex: 1;  
    border-bottom: 1px solid rgba(255, 255, 255, 0.2);  
}  
  
.divider::before {  
    margin-right: 1rem;  
}  
  
.divider::after {  
    margin-left: 1rem;  
}  
  
.auth-links {  
    text-align: center;  
    margin-top: 1.5rem;  
    color: var(--text-secondary);  
    position: relative;  
    z-index: 2;  
}  
  
.auth-links a {  
    display: inline-block;  
    color: var(--accent-purple) !important;  
    text-decoration: none;  
}
```

```
    font-weight: 500;
    transition: all 0.3s ease;
    padding: 0.25rem 0.5rem;
    border-radius: var(--border-radius);
    position: relative;
    z-index: 3;
    cursor: pointer;
}

.auth-links a:hover {
    color: var(--light-purple) !important;
    text-decoration: underline;
    transform: translateY(-1px);
}

.error-message {
    background-color: rgba(220, 53, 69, 0.1);
    color: #ff6b6b;
    padding: 1rem;
    border-radius: var(--border-radius);
    margin-bottom: 1.5rem;
    border: 1px solid rgba(220, 53, 69, 0.2);
    text-align: center;
}

@keyframes fadeIn {
    from {
        opacity: 0;
        transform: translateY(20px);
    }
    to {
        opacity: 1;
        transform: translateY(0);
    }
}

@media (max-width: 768px) {
    .auth-container {
        margin: 1rem;
        padding: 1.5rem;
        max-width: calc(100% - 2rem);
    }
}
```

```
input[type="text"],  
input[type="password"],  
input[type="email"] {  
    font-size: 16px; /* Предотвращает масштабирование на iOS */  
}  
  
.logo {  
    font-size: 2rem;  
}  
  
.logo i {  
    font-size: 2.25rem;  
}  
  
h1 {  
    font-size: 1.75rem;  
}  
}  
  
/* Добавляем новые стили для страницы входа */  
.login-form {  
    display: flex;  
    flex-direction: column;  
    gap: 1.5rem;  
}  
  
.remember-me {  
    display: none;  
}  
  
.button-container {  
    margin-top: 0.5rem;  
}  
  
.login-button {  
    width: 100%;  
    padding: 0.75rem;  
    background-color: var(--medium-purple);  
    color: var(--text-light);  
    border: none;  
    border-radius: var(--border-radius);
```

```
    font-weight: 500;
    font-size: 1rem;
    cursor: pointer;
    transition: all 0.3s ease;
    display: flex;
    align-items: center;
    justify-content: center;
    gap: 0.5rem;
}
```

```
.login-button:hover {
    background-color: var(--accent-purple);
    transform: translateY(-2px);
    box-shadow: var(--shadow-md);
}
```

```
.login-button i {
    font-size: 1.1rem;
}
```

```
/* Обновляем стили для чекбокса */
input[type="checkbox"] {
    appearance: none;
    -webkit-appearance: none;
    width: 18px;
    height: 18px;
    background-color: rgba(255, 255, 255, 0.1);
    border: 1px solid rgba(255, 255, 255, 0.2);
    border-radius: 4px;
    display: inline-flex;
    align-items: center;
    justify-content: center;
    cursor: pointer;
    transition: all 0.3s ease;
}
```

```
input[type="checkbox"]:checked {
    background-color: var(--accent-purple);
    border-color: var(--accent-purple);
}
```

```
input[type="checkbox"]:checked::before {
```

```

content: '✓';
color: var(--text-light);
font-size: 12px;
font-weight: bold;
}

input[type="checkbox"]:hover {
background-color: rgba(255, 255, 255, 0.15);
}

input[type="checkbox"]:checked:hover {
background-color: var(--medium-purple);
}

/* Добавляем box-sizing для всех элементов */
*, *::before, *::after {
box-sizing: border-box;
}

/* Убираем возможные конфликты со стилями */
* {
pointer-events: auto;
}

```

## PracticalWork.css

```

/* Стили для страницы практических работ - согласованные с общим дизайном */
:root {
/* Основные цвета */
--dark-purple: #1a1225;
--medium-purple: #7743DB;
--light-purple: #B39DDB;
--accent-purple: #9D4EDD;
--primary: #9d4edd;
--primary-light: #c77dff;
--primary-dark: #7b2cbf;
--text-color: #ffffff !important;
--text-secondary: rgba(255, 255, 255, 0.85) !important;
--text-light: #ffffff !important;
--card-bg: rgba(39, 33, 52, 0.95);
--card-hover: #362b48;
--gradient-bg: linear-gradient(135deg, #1a1225 0%, #3a1d69 100%);
--shadow-sm: 0 4px 6px rgba(0, 0, 0, 0.3);
--shadow-md: 0 6px 12px rgba(0, 0, 0, 0.4);

```

```

/* Статусы */
--success: #28a745;
--warning: #ffc107;
--danger: #dc3545;
--info: #17a2b8;

/* Визуальные элементы */
--border-radius-lg: 16px;
--border-radius: 12px;
--border-radius-sm: 8px;
--shadow-soft: 0 10px 30px rgba(0, 0, 0, 0.15);
--shadow-strong: 0 15px 35px rgba(0, 0, 0, 0.2);
--glass-effect: rgba(255, 255, 255, 0.08);

/* Анимации */
--transition-fast: 0.2s ease;
--transition-medium: 0.3s ease;

/* Градиенты */
--gradient-primary: linear-gradient(135deg, #9d4edd, #7b2cbf);
--gradient-dark: linear-gradient(135deg, #1a1225, #2d1c3b);
}

.main-header .container {
margin: 0 !important;
max-width: 100% !important;
}

.main-header .container {
max-width: 100% !important;
width: 100%;
padding: 0.8rem 2rem;
display: flex;
justify-content: space-between;
align-items: center;
margin: 0 !important;
background-color: transparent !important;
box-shadow: none !important;
}

/* Принудительное переопределение базовых стилей - согласованно с другими страницами */

```

```
html, body {  
    background-color: var(--dark-purple) !important;  
    color: var(--text-color, #ffffff) !important;  
    font-family: 'Roboto', sans-serif;  
    line-height: 1.6;  
    margin: 0;  
    padding: 0;  
}  
  
/* Основной контейнер */  
.container {  
    max-width: 1200px;  
    margin: 2rem auto;  
    padding: 0 2rem;  
    background-color: transparent !important;  
    box-shadow: none !important;  
}  
  
/* Заголовок страницы */  
.container h1 {  
    font-size: 2.5rem;  
    font-weight: 700;  
    text-align: center;  
    color: var(--text-color);  
    margin-bottom: 2rem;  
    padding: 1.5rem;  
    position: relative;  
}  
  
/* Убираем декоративные элементы из заголовка */  
.container h1::before, .container h1::after {  
    display: none;  
}  
  
.container h2 {  
    font-size: 1.8rem;  
    font-weight: 600;  
    color: var(--text-color);  
    margin: 2rem 0 1.5rem;  
}  
  
/* Упрощаем стиль подзаголовков */
```

```
.container h2::before {  
    display: none;  
}  
  
/* Карточки и панели */  
.details, .completed-work, .student-table-container,  
.student-work-details, .export-actions, .warning-banner,  
.success-banner, .info-banner, .error-banner,  
.grade-form, .grade-result, .quick-grade-section {  
    background-color: var(--card-bg);  
    border-radius: var(--border-radius);  
    padding: 1.5rem;  
    margin-bottom: 1.5rem;  
    box-shadow: var(--shadow-sm);  
    color: var(--text-color);  
    position: relative;  
    border: 1px solid rgba(255, 255, 255, 0.05);  
}  
  
/* Убираем анимацию при наведении */  
.details:hover, .completed-work:hover, .student-table-container:hover,  
.export-actions:hover {  
    transform: none;  
    box-shadow: var(--shadow-sm);  
}  
  
/* Убираем цветную полосу слева */  
.details::before, .completed-work::before,  
.export-actions::before {  
    display: none;  
}  
  
/* Заголовки в карточках */  
.completed-work h3, .student-work-details h4, .quick-grade-section h4 {  
    color: var(--text-color);  
    font-size: 1.4rem;  
    margin-top: 0;  
    margin-bottom: 1.5rem;  
    padding-bottom: 0.8rem;  
    border-bottom: 1px solid rgba(255, 255, 255, 0.1);  
}
```

```
/* Информация о файле */
.file-info {
    background-color: rgba(255, 255, 255, 0.05);
    padding: 1rem;
    border-radius: var(--border-radius);
    margin-top: 1rem;
    display: flex;
    align-items: center;
    flex-wrap: wrap;
    gap: 0.5rem;
    border: 1px solid rgba(255, 255, 255, 0.1);
}

/* Ссылки на файлы */
.file-link, a[href$=".pdf"], a[href$=".doc"], a[href$=".docx"],
a[href$=".zip"], a[href$=".rar"] {
    color: var(--accent-purple);
    text-decoration: none;
    font-weight: 500;
    display: inline-flex;
    align-items: center;
    gap: 0.5rem;
    transition: all 0.2s ease;
}

.file-link:hover, a[href$=".pdf"]:hover, a[href$=".doc"]:hover,
a[href$=".docx"]:hover, a[href$=".zip"]:hover, a[href$=".rar"]:hover {
    color: var(--light-purple);
    text-decoration: underline;
}

.file-icon {
    font-size: 1.2rem;
}

/* Кнопки */
.back-button, .active-button, .export-button,
.toggle-details-btn, .grade-button,
button[type="submit"], .submit-btn, .grade-submit-btn {
    background: var(--medium-purple);
    color: white;
    font-weight: 600;
}
```

```
font-size: 1rem;
padding: 0.7rem 1.2rem;
border: none;
border-radius: 8px;
cursor: pointer;
display: inline-flex;
align-items: center;
justify-content: center;
gap: 0.5rem;
transition: all 0.2s ease;
text-decoration: none;
}

/* Упрощаем эффект при наведении */
.back-button:hover, .active-button:hover, .export-button:hover,
.toggle-details-btn:hover, .grade-button:hover,
button[type="submit"]:hover, .submit-btn:hover, .grade-submit-btn:hover {
background: var(--accent-purple);
transform: none;
}

/* Убираем анимацию подсветки */
.back-button::before, .active-button::before, .export-button::before,
.toggle-details-btn::before, .grade-button::before,
button[type="submit"]::before, .submit-btn::before, .grade-submit-btn::before {
display: none;
}

/* Кнопка назад */
.back-button {
margin: 1rem 0 2rem;
background: var(--dark-purple);
border: 1px solid rgba(255, 255, 255, 0.1);
}

/* Таблица студентов */
.student-table-container {
padding: 0;
overflow: hidden;
border-radius: var(--border-radius);
}
```

```
.student-table {  
    width: 100%;  
    border-collapse: collapse;  
}  
  
.student-table th {  
    background: var(--medium-purple);  
    color: white;  
    padding: 1rem;  
    text-align: left;  
    font-weight: 600;  
    font-size: 1rem;  
}  
  
.student-table td {  
    padding: 1rem;  
    border-bottom: 1px solid rgba(255, 255, 255, 0.1);  
    color: var(--text-color);  
}  
  
.student-details {  
    background-color: rgba(255, 255, 255, 0.03);  
}  
  
/* Отображение статусов */  
.status-badge, .grade-pending, .grade-2, .grade-3-5 {  
    display: inline-flex;  
    align-items: center;  
    justify-content: center;  
    padding: 0.35rem 1rem;  
    border-radius: 30px;  
    font-size: 0.85rem;  
    font-weight: 600;  
    min-width: 100px;  
}  
  
.status-passed {  
    background-color: rgba(40, 167, 69, 0.15);  
    color: var(--success);  
    border: 1px solid rgba(40, 167, 69, 0.3);  
}
```

```
.status-pending {  
    background-color: rgba(255, 193, 7, 0.15);  
    color: var(--warning);  
    border: 1px solid rgba(255, 193, 7, 0.3);  
}  
  
.grade-pending {  
    background-color: rgba(108, 117, 125, 0.15);  
    color: var(--text-secondary);  
    border: 1px solid rgba(108, 117, 125, 0.3);  
}  
  
.grade-2 {  
    background-color: rgba(220, 53, 69, 0.15);  
    color: var(--danger);  
    border: 1px solid rgba(220, 53, 69, 0.3);  
}  
  
.grade-3-5 {  
    background-color: rgba(40, 167, 69, 0.15);  
    color: var(--success);  
    border: 1px solid rgba(40, 167, 69, 0.3);  
}  
  
/* Информационные баннеры */  
.warning-banner, .success-banner, .info-banner, .error-banner {  
    padding: 1.5rem 1.5rem 1.5rem 4rem;  
    margin: 1.5rem 0;  
    border-radius: var(--border-radius-sm);  
    position: relative;  
}  
  
.warning-banner::before, .success-banner::before,  
.info-banner::before, .error-banner::before {  
    font-family: 'Font Awesome 5 Free';  
    font-weight: 900;  
    position: absolute;  
    left: 1.5rem;  
    top: 50%;  
    transform: translateY(-50%);  
    font-size: 1.5rem;  
}
```

```
.warning-banner {  
    background-color: rgba(255, 193, 7, 0.1);  
    border-left: 4px solid var(--warning);  
}  
.warning-banner::before {  
    content: '\f071'; /* иконка предупреждения */  
    color: var(--warning);  
}  
  
.success-banner {  
    background-color: rgba(40, 167, 69, 0.1);  
    border-left: 4px solid var(--success);  
}  
.success-banner::before {  
    content: '\f00c'; /* иконка галочки */  
    color: var(--success);  
}  
  
.info-banner {  
    background-color: rgba(23, 162, 184, 0.1);  
    border-left: 4px solid var(--info);  
}  
.info-banner::before {  
    content: '\f129'; /* иконка информации */  
    color: var(--info);  
}  
  
.error-banner {  
    background-color: rgba(220, 53, 69, 0.1);  
    border-left: 4px solid var(--danger);  
}  
.error-banner::before {  
    content: '\f00d'; /* иконка крестика */  
    color: var(--danger);  
}  
  
/* Формы */  
.grade-form {  
    background-color: rgba(255, 255, 255, 0.03);  
    border-radius: var(--border-radius);  
    padding: 1.2rem;  
}
```

```
margin-top: 1rem;
border: 1px solid rgba(255, 255, 255, 0.1);
}

.grade-form-group {
display: flex;
align-items: center;
flex-wrap: wrap;
gap: 1rem;
}

.grade-form label {
color: var(--text-secondary);
font-weight: 500;
}

.grade-form select,
.grade-form input[type="text"],
.grade-form textarea {
padding: 0.8rem 1rem;
border: 1px solid rgba(0, 0, 0, 0.1);
border-radius: var(--border-radius-sm);
font-size: 1rem;
transition: all var(--transition-fast);
min-width: 200px;
}

.grade-form select:focus,
.grade-form input[type="text"]:focus,
.grade-form textarea:focus {
outline: none;
border-color: var(--primary);
box-shadow: 0 0 0 3px rgba(157, 78, 221, 0.2);
}

/* Индикатор попыток */
.attempt-counter {
display: flex;
align-items: center;
gap: 0.8rem;
margin: 1.5rem 0;
}
```

```
.attempt-counter-label {  
    font-weight: 500;  
}  
  
.attempt-circle {  
    width: 32px;  
    height: 32px;  
    border-radius: 50%;  
    display: flex;  
    align-items: center;  
    justify-content: center;  
    font-size: 0.9rem;  
    font-weight: 600;  
    transition: transform var(--transition-fast);  
}  
  
.attempt-used {  
    background: var(--gradient-primary);  
    color: white;  
    box-shadow: 0 3px 6px rgba(0, 0, 0, 0.1);  
}  
  
.attempt-available {  
    background-color: rgba(108, 117, 125, 0.1);  
    color: var(--text-secondary);  
    border: 1px solid rgba(108, 117, 125, 0.3);  
}  
  
.attempt-used:hover,  
.attempt-available:hover {  
    transform: scale(1.1);  
}  
  
/* Работы студентов */  
.student-work-details {  
    padding: 1.8rem;  
}  
  
.work-header {  
    display: flex;  
    justify-content: space-between;
```

```
    align-items: center;
    margin-bottom: 1.2rem;
    padding-bottom: 0.8rem;
    border-bottom: 1px dashed rgba(0, 0, 0, 0.1);
}
```

```
.attempt-number {
    font-weight: 600;
    color: var(--primary);
    padding: 0.3rem 0.8rem;
    background-color: rgba(157, 78, 221, 0.1);
    border-radius: 20px;
    font-size: 0.85rem;
}
```

```
.work-date {
    color: var(--text-secondary);
    font-size: 0.9rem;
}
```

```
.grade-result {
    margin: 1.5rem 0;
    padding: 1.5rem;
    border-radius: var(--border-radius-sm);
    border: 1px solid rgba(0, 0, 0, 0.05);
}
```

```
.grade-result h5 {
    margin-top: 0;
    font-size: 1.2rem;
    color: var(--primary-dark);
    margin-bottom: 1rem;
}
```

```
/* Экспорт данных */
.export-actions {
    display: flex;
    justify-content: center;
    gap: 1.5rem;
    padding: 1.5rem;
    flex-wrap: wrap;
    background-color: var(--card-bg);
```

```
}

.export-button {
    padding: 1rem 1.5rem;
    background: var(--gradient-primary);
    min-width: 220px;
    font-size: 1.05rem;
}

/* График (chart.js) */
canvas {
    max-width: 100%;
    margin: 2.5rem auto;
    background-color: var(--card-bg);
    padding: 1.5rem;
    border-radius: var(--border-radius);
    box-shadow: var(--shadow-soft);
    border: 1px solid rgba(255, 255, 255, 0.2);
}

/* Отладочная информация */
div[style*="background-color: #ff8f9fa"] {
    background-color: var(--card-bg) !important;
    border-radius: var(--border-radius-sm);
    padding: 1.5rem;
    margin-top: 2rem;
    border: 1px solid rgba(0, 0, 0, 0.1);
    box-shadow: var(--shadow-soft);
}

/* Анимации */
@keyframes fadeIn {
    from { opacity: 0; }
    to { opacity: 1; }
}

/* Стиль для кнопки выхода - соответствует стилю на других страницах */
.logout-btn {
    background-color: var(--medium-purple);
    color: white;
    padding: 0.5rem 1rem;
    border-radius: 6px;
}
```

```
text-decoration: none;
display: inline-flex;
align-items: center;
gap: 0.5rem;
transition: var(--transition-fast);
}

.logout-btn:hover {
background-color: var(--accent-purple);
transform: translateY(-2px);
text-decoration: none;
}

/* Адаптивность */

@media (max-width: 992px) {
.container {
padding: 0 1.5rem;
}
.container h1 {
font-size: 2rem;
padding: 1.5rem;
}
}

@media (max-width: 768px) {
.container {
padding: 0 1rem;
margin: 1rem auto;
}
.container h1 {
font-size: 1.8rem;
padding: 1.2rem;
margin-bottom: 2rem;
}
}

.details, .completed-work, .student-table-container,
.student-work-details, .export-actions {
padding: 1.5rem;
}
```

```
.student-table th, .student-table td {  
    padding: 1rem 0.75rem;  
    font-size: 0.9rem;  
}  
  
.grade-form-group {  
    flex-direction: column;  
    align-items: flex-start;  
}  
  
.grade-form select,  
.grade-form input[type="text"] {  
    width: 100%;  
    min-width: auto;  
}  
  
.export-actions {  
    flex-direction: column;  
}  
  
.attempt-counter {  
    flex-wrap: wrap;  
}  
}  
  
 @media (max-width: 576px) {  
    .container h1 {  
        font-size: 1.5rem;  
    }  
  
    .container h2 {  
        font-size: 1.4rem;  
    }  
  
.back-button, .active-button, .export-button,  
.toggle-details-btn, .grade-button {  
    width: 100%;  
    justify-content: center;  
    margin-bottom: 0.5rem;  
}  
  
.student-work-details {
```

```

padding: 1.2rem;
}

.work-header {
  flex-direction: column;
  align-items: flex-start;
  gap: 0.5rem;
}
}

/* Общие стили для текста на странице */
body, p, h1, h2, h3, h4, h5, h6, span, div, li, a, button, input, textarea, label {
  color: var(--text-color, #ffffff) !important;
}

.task-text, .description-text, .work-content, .work-details, .work-header, .work-title, .work-instructions {
  color: var(--text-color, #ffffff) !important;
}

/* Для элементов с установленным темным цветом текста */
.dark-text, .task-title, .completion-status, .work-status, .deadline, .submission-info, .grade-info {
  color: var(--text-color, #ffffff) !important;
}

/* Making sure headers are clearly visible */
.content-header h1, .content-header h2, .section-title, .practical-title {
  color: var(--text-color);
  text-shadow: 0 1px 3px rgba(0, 0, 0, 0.3); /* Adding shadow for better contrast */
}

/* Targeting specific containers and important text */
.practical-info p, .practical-description, .work-details, .student-info, .completed-works-list, .work-card {
  color: var(--text-color);
}

/* Ensuring form labels and important instructions are visible */
label, .form-group span, .instructions, .upload-note {
  color: var(--text-secondary);
}

/* Making sure links are visible */
a, a:visited {

```

```

color: #b39ddb; /* Light purple for links */
}

a:hover {
  color: #ffffff; /* White on hover for better visibility */
}

/* Improving visibility of table text if present */
table {
  color: var(--text-color);
}

/* Ensuring status and grading information is visible */
.status-label, .grade-label, .attempt-number {
  color: var(--text-color);
  font-weight: 500; /* Making it slightly bolder */
}

/* Targeting specific panel and card elements */
.panel-content, .card-body, .info-panel, .work-submission {
  color: var(--text-color);
}

/* Группа кнопок */
.button-group {
  display: flex;
  gap: 1rem;
  margin: 1rem 0 2rem;
  align-items: center;
  flex-wrap: wrap;
}

/* Стили для формы удаления */
.delete-form {
  margin: 0;
}

/* Кнопка удаления */
.delete-button {
  display: inline-flex;
  align-items: center;
  gap: 0.5rem;
}

```

```
padding: 0.7rem 1.2rem;  
background-color: #dc3545;  
color: white !important;  
border: none;  
border-radius: var(--border-radius);  
font-size: 1rem;  
font-weight: 500;  
cursor: pointer;  
transition: all 0.3s ease;  
text-decoration: none;  
border: 1px solid rgba(255, 255, 255, 0.1);  
box-shadow: var(--shadow-sm);  
}
```

```
.delete-button:hover {  
background-color: #c82333;  
transform: translateY(-2px);  
box-shadow: var(--shadow-md);  
}
```

/\* Обновляем стиль кнопки "Назад" для соответствия \*/

```
.back-button {  
display: inline-flex;  
align-items: center;  
gap: 0.5rem;  
padding: 0.7rem 1.2rem;  
background: var(--dark-purple);  
color: white !important;  
text-decoration: none;  
border-radius: var(--border-radius);  
font-size: 1rem;  
font-weight: 500;  
transition: all 0.3s ease;  
border: 1px solid rgba(255, 255, 255, 0.1);  
box-shadow: var(--shadow-sm);  
}
```

```
.back-button:hover {  
background: var(--card-hover);  
transform: translateY(-2px);  
box-shadow: var(--shadow-md);  
text-decoration: none;
```

```

}

/* Адаптивность для мобильных устройств */

@media (max-width: 576px) {

    .button-group {
        flex-direction: column;
        width: 100%;
    }

    .back-button,
    .delete-button {
        width: 100%;
        justify-content: center;
    }
}

```

## SubjectCRUD.css

```

:root {

    --dark-purple: #1a1225;
    --medium-purple: #7743DB;
    --light-purple: #B39DDB;
    --accent-purple: #9D4EDD;
    --text-light: #ffffff;
    --text-secondary: rgba(255, 255, 255, 0.85);
    --card-bg: #272134;
    --card-hover: #362b48;
    --gradient-bg: linear-gradient(135deg, #1a1225 0%, #3a1d69 100%);
    --shadow-sm: 0 4px 6px rgba(0, 0, 0, 0.3);
    --shadow-md: 0 6px 12px rgba(0, 0, 0, 0.4);
    --shadow-lg: 0 10px 25px rgba(0, 0, 0, 0.5);
    --border-radius: 12px;
}

```

```

/* Глобальные стили */

*, *::before, *::after {
    color: var(--text-light) !important;
}

```

```

body {
    font-family: 'Roboto', sans-serif;
    background-color: var(--dark-purple) !important;
    margin: 0;
    min-height: 100vh;
}

```

```
}
```

```
/* Контейнер */
```

```
.container {  
    max-width: 1200px;  
    margin: 2rem auto;  
    padding: 2rem;  
    background: var(--gradient-bg) !important;  
    border-radius: var(--border-radius);  
    box-shadow: var(--shadow-md);  
    position: relative;  
    overflow: visible;  
}
```

```
.container::before {
```

```
    content: "";  
    position: absolute;  
    top: 0;  
    left: 0;  
    right: 0;  
    bottom: 0;  
    background: url('data:image/svg+xml;utf8,<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 100 100" preserveAspectRatio="none"><path fill="none" stroke="rgba(255,255,255,0.05)" stroke-width="1" d="M0,0 L100,100 M100,0 L0,100"/></svg>');  
    background-size: 20px 20px;  
    opacity: 0.3;  
    z-index: 0;  
}
```

```
/* Заголовки */
```

```
h1, h2, h3 {  
    color: var(--text-light) !important;  
    text-align: center;  
    margin-bottom: 2rem;  
    position: relative;  
    z-index: 1;  
}
```

```
h1 {
```

```
    font-size: 2.25rem;  
    font-weight: 700;  
}
```

```
h2 {  
    font-size: 1.75rem;  
    font-weight: 600;  
}  
  
/* Добавляем стили для секции с заголовком и кнопкой назад */  
.header-section {  
    position: relative;  
    margin-bottom: 2rem;  
    z-index: 2;  
}  
  
/* Форма создания предмета */  
.crud-form {  
    background-color: var(--card-bg);  
    padding: 2rem;  
    border-radius: var(--border-radius);  
    margin-bottom: 2rem;  
    position: relative;  
    z-index: 1;  
    border: 1px solid rgba(255, 255, 255, 0.1);  
}  
  
.crud-form label {  
    display: block;  
    margin-bottom: 0.5rem;  
    color: var(--text-light) !important;  
    font-weight: 500;  
}  
  
.crud-form input,  
.crud-form select {  
    width: 100%;  
    padding: 0.75rem;  
    background-color: rgba(255, 255, 255, 0.05);  
    border: 1px solid rgba(255, 255, 255, 0.1);  
    border-radius: var(--border-radius);  
    color: var(--text-light) !important;  
    margin-bottom: 1rem;  
}
```

```
.crud-form input:focus,  
.crud-form select:focus {  
    outline: none;  
    border-color: var(--accent-purple);  
    box-shadow: 0 0 0 2px rgba(157, 78, 221, 0.2);  
}  
  
/* Кнопки */  
button {  
    padding: 0.75rem 1.5rem;  
    background-color: var(--medium-purple);  
    color: var(--text-light) !important;  
    border: none;  
    border-radius: var(--border-radius);  
    cursor: pointer;  
    transition: all 0.3s ease;  
    font-weight: 500;  
    width: 100%;  
}  
  
button:hover {  
    background-color: var(--accent-purple);  
    transform: translateY(-2px);  
    box-shadow: var(--shadow-md);  
}  
  
.delete-button {  
    background-color: #dc3545;  
    margin-top: 0.5rem;  
}  
  
.delete-button:hover {  
    background-color: #c82333;  
}  
  
/* Список предметов */  
.subject-list {  
    display: grid;  
    grid-template-columns: repeat(auto-fill, minmax(300px, 1fr));  
    gap: 1.5rem;  
    position: relative;  
    z-index: 1;  
}
```

```
}

.subject-card {
    background-color: var(--card-bg);
    border: 1px solid rgba(255, 255, 255, 0.1);
    border-radius: var(--border-radius);
    padding: 1.5rem;
    transition: all 0.3s ease;
}

.subject-card:hover {
    transform: translateY(-5px);
    box-shadow: var(--shadow-lg);
    background-color: var(--card-hover);
}

.subject-card h3 {
    font-size: 1.2rem;
    margin-bottom: 1rem;
    text-align: left;
}

.subject-card p {
    margin: 0.5rem 0;
    color: var(--text-secondary) !important;
}

.subject-card strong {
    color: var(--text-light) !important;
    font-weight: 500;
}

/* Обновляем стили для кнопки назад */
.back-button {
    display: inline-flex;
    align-items: center;
    gap: 0.75rem;
    padding: 0.75rem 1.5rem;
    background-color: var(--medium-purple);
    color: var(--text-light) !important;
    border-radius: var(--border-radius);
    text-decoration: none;
}
```

```
font-weight: 500;
transition: all 0.3s ease;
border: 1px solid rgba(255, 255, 255, 0.1);
margin-bottom: 1rem;
width: auto; /* Убираем width: 100% для кнопки назад */
}

.back-button:hover {
background-color: var(--accent-purple);
transform: translateY(-2px);
box-shadow: var(--shadow-md);
text-decoration: none;
}

.back-button i {
font-size: 1.1rem;
}

/* Действия с предметом */
.crud-actions {
display: flex;
gap: 1rem;
margin-top: 1rem;
}

.crud-actions a.button {
padding: 0.75rem 1.5rem;
background-color: var(--medium-purple);
color: var(--text-light) !important;
border-radius: var(--border-radius);
text-decoration: none;
text-align: center;
flex: 1;
transition: all 0.3s ease;
}

.crud-actions a.button:hover {
background-color: var(--accent-purple);
transform: translateY(-2px);
box-shadow: var(--shadow-md);
}
```

```

.crud-actions form {
    flex: 1;
}

/* Адаптивность */

@media (max-width: 768px) {
    .container {
        margin: 1rem;
        padding: 1rem;
    }

    .crud-form {
        padding: 1rem;
    }

    .subject-list {
        grid-template-columns: 1fr;
    }
}

.crud-actions {
    flex-direction: column;
}
}

/* Выпадающие списки */

select {
    appearance: none;
    background-image: url("data:image/svg+xml; charset=UTF-8, %3csvg
xmlns='http://www.w3.org/2000/svg' viewBox='0 0 24 24' fill='white'%3e%3cpath d='M7 10l5 5 5z'%3e%3c/svg%3e");
    background-repeat: no-repeat;
    background-position: right 0.7rem center;
    background-size: 1.5em;
    padding-right: 2.5rem;
}

```

## SubjectPage.css

```

/* Современный адаптивный стиль для страницы предмета - темная тема */

:root {
    --dark-purple: #1a1225;
    --medium-purple: #7743DB;
    --light-purple: #B39DDB;
    --accent-purple: #9D4EDD;
}

```

```
--text-light: #ffffff;  
--text-secondary: rgba(255, 255, 255, 0.85);  
--card-bg: #272134;  
--card-hover: #362b48;  
--gradient-bg: linear-gradient(135deg, #1a1225 0%, #3a1d69 100%);  
--shadow-sm: 0 4px 6px rgba(0, 0, 0, 0.3);  
--shadow-md: 0 6px 12px rgba(0, 0, 0, 0.4);  
--shadow-lg: 0 10px 25px rgba(0, 0, 0, 0.5);  
--transition-slow: all 0.4s cubic-bezier(0.165, 0.84, 0.44, 1);  
--transition-fast: all 0.2s ease;  
--border-radius: 12px;  
--card-spacing: 1.75rem;  
}
```

/\* Принудительное переопределение базовых стилей \*/

```
html {  
background-color: var(--dark-purple) !important;  
}
```

```
body {  
background-color: var(--dark-purple) !important;  
color: var(--text-light) !important;  
font-family: 'Roboto', sans-serif;  
margin: 0;  
padding: 0;  
min-height: 100vh;  
width: 100%;  
overflow-x: hidden; /* Предотвращает горизонтальный скролл */  
}
```

/\* Корректировка стилей header - такие же, как на странице предметов \*/

```
.main-header {  
background-color: var(--dark-purple) !important;  
width: 100%;  
box-shadow: var(--shadow-md);  
position: relative;  
z-index: 100;  
border-bottom: 1px solid rgba(255, 255, 255, 0.1);  
}
```

```
.main-header .container {  
max-width: 100% !important;
```

```
width: 100%;  
padding: 0.8rem 2rem;  
display: flex;  
justify-content: space-between;  
align-items: center;  
margin: 0 !important;  
background-color: transparent !important;  
box-shadow: none !important;  
}
```

/\* Основной контейнер страницы \*/

```
.container {  
max-width: 1200px;  
margin: 0 auto;  
padding: 2rem;  
background-color: transparent !important;  
box-shadow: none !important;  
}
```

/\* Заголовок предмета \*/

```
h1 {  
font-size: 2.8rem;  
font-weight: 700;  
margin-bottom: 1.5rem;  
color: white !important;  
text-shadow: 0 2px 4px rgba(0, 0, 0, 0.3);  
text-align: center;  
background: var(--gradient-bg);  
padding: 2rem;  
border-radius: var(--border-radius);  
box-shadow: var(--shadow-md);  
margin-top: 2rem;  
position: relative;  
overflow: hidden;  
}
```

```
h1::before {  
content: "";  
position: absolute;  
top: 0;  
left: 0;  
right: 0;
```

```
bottom: 0;  
background: url('data:image/svg+xml;utf8,<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 100 100" preserveAspectRatio="none"><path fill="none" stroke="rgba(255,255,255,0.05)" stroke-width="1" d="M0,0 L100,100 M100,0 L0,100"/></svg>');  
background-size: 20px 20px;  
opacity: 0.3;  
z-index: 0;  
}  
  
/* Информация о предмете */  
.subject-info {  
background-color: var(--card-bg);  
padding: 1.5rem;  
border-radius: var(--border-radius);  
margin-bottom: 2rem;  
display: flex;  
flex-wrap: wrap;  
justify-content: space-between;  
border: 1px solid rgba(255, 255, 255, 0.05);  
box-shadow: var(--shadow-sm);  
}  
  
.subject-info p {  
margin: 0.5rem 0;  
font-size: 1.1rem;  
flex: 1;  
min-width: 250px;  
color: white !important;  
font-weight: 400;  
}  
  
.subject-info strong {  
color: var(--accent-purple);  
margin-right: 0.5rem;  
font-weight: 600;  
}  
  
/* Секции контента (Лекции, Практические работы, Тесты) */  
.content-section {  
margin-bottom: 2.5rem;  
background-color: var(--card-bg);  
border-radius: var(--border-radius);
```

```
padding: 1.5rem;
box-shadow: var(--shadow-md);
border: 1px solid rgba(255, 255, 255, 0.05);
transition: var(--transition-fast);
}

.content-section:hover {
  box-shadow: var(--shadow-lg);
  transform: translateY(-5px);
}

.content-section h2 {
  color: white !important;
  font-size: 1.8rem;
  font-weight: 600;
  margin-top: 0;
  margin-bottom: 1.5rem;
  padding-bottom: 0.5rem;
  border-bottom: 2px solid var(--accent-purple);
  position: relative;
}

/* Список контента */
.content-list {
  list-style-type: none;
  padding: 0;
  margin: 0;
  display: grid;
  grid-template-columns: repeat(auto-fill, minmax(300px, 1fr));
  gap: 1rem;
}

.content-list li {
  transition: var(--transition-fast);
  border-radius: var(--border-radius);
  overflow: hidden;
}

.content-list li a {
  display: block;
  padding: 1.2rem;
  background: linear-gradient(145deg, #2d1f3a, #231630);
}
```

```
color: white !important;  
text-decoration: none;  
border-radius: var(--border-radius);  
transition: var(--transition-fast);  
box-shadow: 0 4px 10px rgba(0, 0, 0, 0.3);  
border: 1px solid rgba(255, 255, 255, 0.05);  
position: relative;  
overflow: hidden;  
z-index: 1;  
font-weight: 400;  
}
```

```
.content-list li a::before {  
    content: " ";  
    position: absolute;  
    top: 0;  
    left: 0;  
    width: 100%;  
    height: 100%;  
    background: linear-gradient(90deg,  
        transparent,  
        rgba(255, 255, 255, 0.1),  
        transparent  
    );  
    transform: translateX(-100%);  
    transition: transform 0.6s;  
    z-index: -1;  
}
```

```
.content-list li a:hover {  
    background: var(--card-hover);  
    color: white !important;  
    transform: translateY(-3px);  
    box-shadow: 0 7px 15px rgba(0, 0, 0, 0.4);  
    text-shadow: 0 0 5px rgba(255, 255, 255, 0.3);  
}
```

```
.content-list li a:hover::before {  
    transform: translateX(100%);  
}
```

```
.content-list li a::after {
```

```
content: "→";
position: absolute;
right: 1.2rem;
top: 50%;
transform: translateY(-50%);
opacity: 0;
transition: all 0.3s ease;
}

.content-list li a:hover::after {
    opacity: 1;
    right: 1rem;
}

/* Сообщение об отсутствии элементов */
.content-section p {
    color: white !important;
    font-style: italic;
    text-align: center;
    padding: 2rem;
    background-color: rgba(0, 0, 0, 0.1);
    border-radius: var(--border-radius);
}

/* Добавляем дополнительное свечение для иконок */
.content-list li a i {
    color: var(--accent-purple);
    margin-right: 10px;
    font-size: 1.1rem;
    transition: all 0.3s;
}

.content-list li a:hover i {
    color: white;
    transform: scale(1.1);
    text-shadow: 0 0 10px rgba(255, 255, 255, 0.5);
}

/* Адаптивный дизайн */
@media (max-width: 992px) {
    h1 {
        font-size: 2.4rem;
```

```
padding: 1.8rem;
}

.content-list {
  grid-template-columns: repeat(auto-fill, minmax(250px, 1fr));
}
}

@media (max-width: 768px) {
  h1 {
    font-size: 2rem;
    padding: 1.5rem;
  }

  .container {
    padding: 1.5rem;
  }

  .subject-info {
    flex-direction: column;
  }

  .content-list {
    grid-template-columns: 1fr;
  }

  .content-section h2 {
    font-size: 1.5rem;
  }
}

@media (max-width: 576px) {
  h1 {
    font-size: 1.8rem;
    padding: 1.2rem;
  }

  .container {
    padding: 1rem;
  }

  .content-section {
```

```
padding: 1rem;
}

.content-list li a {
    padding: 1rem;
}
}

/* Добавляем стили для заголовка секции с кнопкой */
.section-header {
    display: flex;
    justify-content: space-between;
    align-items: center;
    margin-bottom: 1.5rem;
}

.section-header h2 {
    margin: 0;
}

/* Стили для кнопки добавления */
.add-content-btn {
    display: inline-flex;
    align-items: center;
    gap: 0.5rem;
    padding: 0.6rem 1.2rem;
    background-color: var(--accent-purple);
    color: white !important;
    text-decoration: none;
    border-radius: var(--border-radius);
    font-size: 0.9rem;
    font-weight: 500;
    transition: all 0.3s ease;
    border: 1px solid rgba(255, 255, 255, 0.1);
    box-shadow: var(--shadow-sm);
}

.add-content-btn:hover {
    background-color: var(--medium-purple);
    transform: translateY(-2px);
    box-shadow: var(--shadow-md);
    text-decoration: none;
}
```

```
}

.add-content-btn i {
    font-size: 0.9rem;
}

/* Адаптивность для мобильных устройств */
@media (max-width: 768px) {
    .section-header {
        flex-direction: column;
        gap: 1rem;
        align-items: flex-start;
    }

    .add-content-btn {
        width: 100%;
        justify-content: center;
    }
}

.button-group {
    display: flex;
    justify-content: space-between;
    align-items: center;
    margin-bottom: 1.5rem;
}

.back-button,
.add-content-btn {
    display: inline-flex;
    align-items: center;
    gap: 0.5rem;
    padding: 0.6rem 1.2rem;
    background-color: #7743DB;
    color: #ffffff !important;
    text-decoration: none;
    border-radius: 8px;
    font-size: 0.9rem;
    font-weight: 500;
    transition: all 0.3s ease;
}
```

```
border: 1px solid rgba(255, 255, 255, 0.1);
box-shadow: 0 4px 10px rgba(0, 0, 0, 0.3);
}

.back-button:hover,
.add-content-btn:hover {
background-color: #5E35B1;
transform: translateY(-2px);
box-shadow: 0 6px 15px rgba(0, 0, 0, 0.4);
text-decoration: none;
}

/* Новые стили */

.button-group {
display: flex;
justify-content: flex-start;
align-items: center;
margin-bottom: 1.5rem;
}

.back-button {
margin-right: auto;
}
```

## TestAttempts.css

```
body {
font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
background-color: #f5f7fa;
margin: 0;
padding: 0;
color: #333;
}

.container {
max-width: 1200px;
margin: 2rem auto;
padding: 0 1rem;
}

h1 {
color: #740978;
font-size: 2rem;
margin-bottom: 2rem;
```

```
    text-align: center;
}

h2 {
    color: #444;
    font-size: 1.5rem;
    margin: 1.5rem 0;
}

.test-info, .grading-criteria, .statistics, .students-results {
    background: white;
    border-radius: 8px;
    padding: 1.5rem;
    margin-bottom: 2rem;
    box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
}

.grading-criteria .criteria-content {
    background: #f8f9fa;
    padding: 1rem;
    border-radius: 4px;
    font-family: monospace;
    white-space: pre-wrap;
}

.stats-grid {
    display: grid;
    grid-template-columns: repeat(auto-fit, minmax(200px, 1fr));
    gap: 1.5rem;
    margin-top: 1rem;
}

.stat-card {
    background: #f8f9fa;
    padding: 1.5rem;
    border-radius: 6px;
    text-align: center;
    transition: transform 0.2s;
}

.stat-card:hover {
    transform: translateY(-2px);
```

```
}
```

```
.stat-card h3 {  
    color: #666;  
    font-size: 1rem;  
    margin: 0 0 0.5rem 0;  
}
```

```
.stat-value {  
    color: #740978;  
    font-size: 2rem;  
    font-weight: bold;  
    margin: 0;  
}
```

```
.students-list {  
    display: grid;  
    grid-template-columns: repeat(auto-fit, minmax(300px, 1fr));  
    gap: 1.5rem;  
}
```

```
.student-card {  
    background: white;  
    border-radius: 8px;  
    padding: 1.5rem;  
    box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);  
}
```

```
.student-header {  
    display: flex;  
    justify-content: space-between;  
    align-items: center;  
    margin-bottom: 1rem;  
    padding-bottom: 0.5rem;  
    border-bottom: 1px solid #eee;  
}
```

```
.student-header h3 {  
    margin: 0;  
    color: #444;  
}
```

```
.attempt-count {  
    color: #666;  
    font-size: 0.9rem;  
}  
  
.attempts-list {  
    display: flex;  
    flex-direction: column;  
    gap: 1rem;  
}  
  
.attempt-item {  
    background: #f8f9fa;  
    border-radius: 6px;  
    padding: 1rem;  
}  
  
.attempt-info {  
    display: flex;  
    justify-content: space-between;  
    margin-bottom: 0.5rem;  
}  
  
.attempt-number {  
    font-weight: bold;  
    color: #740978;  
}  
  
.attempt-date {  
    color: #666;  
    font-size: 0.9rem;  
}  
  
.attempt-details {  
    display: grid;  
    grid-template-columns: repeat(auto-fit, minmax(150px, 1fr));  
    gap: 0.5rem;  
}  
  
.attempt-details p {  
    margin: 0.25rem 0;  
}
```

```
.status-completed {  
    color: #28a745;  
}  
  
.status-in-progress {  
    color: #ffc107;  
}  
  
.actions {  
    display: flex;  
    justify-content: center;  
    gap: 1rem;  
    margin-top: 2rem;  
}  
  
.back-btn, .export-btn {  
    padding: 0.75rem 1.5rem;  
    border-radius: 4px;  
    font-weight: 500;  
    text-decoration: none;  
    transition: all 0.2s;  
    cursor: pointer;  
}  
  
.back-btn {  
    background-color: #6c757d;  
    color: white;  
    border: none;  
}  
  
.export-btn {  
    background-color: #740978;  
    color: white;  
    border: none;  
}  
  
.back-btn:hover, .export-btn:hover {  
    transform: translateY(-2px);  
    box-shadow: 0 2px 4px rgba(0, 0, 0, 0.2);  
}
```

```
.no-attempts {
    text-align: center;
    color: #666;
    font-style: italic;
    padding: 2rem;
}

@media (max-width: 768px) {
    .container {
        margin: 1rem auto;
    }

    .stats-grid {
        grid-template-columns: 1fr;
    }

    .students-list {
        grid-template-columns: 1fr;
    }

    .attempt-details {
        grid-template-columns: 1fr;
    }

    .actions {
        flex-direction: column;
        align-items: stretch;
    }

    .back-btn, .export-btn {
        text-align: center;
    }
}

.criteria-table {
    width: 100%;
    border-collapse: collapse;
    margin: 15px 0;
}

.criteria-table th, .criteria-table td {
    padding: 10px;
}
```

```
text-align: center;
border: 1px solid #ddd;
}

.criteria-table th {
background-color: #f2f2f2;
font-weight: bold;
}

.criteria-table tr:hover {
background-color: #f5f5f5;
}

.grade-2-row {
background-color: rgba(220, 53, 69, 0.1);
}

.grade-2-row:hover {
background-color: rgba(220, 53, 69, 0.2);
}

/* Стили для оценок в таблице критериев */
.criteria-table strong[data-grade] {
padding: 3px 8px;
border-radius: 4px;
display: inline-block;
min-width: 20px;
text-align: center;
font-weight: bold;
}

.criteria-table strong[data-grade="5"] {
background-color: #28a745;
color: white;
}

.criteria-table strong[data-grade="4"] {
background-color: #17a2b8;
color: white;
}

.criteria-table strong[data-grade="3"] {
```

```
background-color: #ffc107;
color: #333;
}

.criteria-table strong[data-grade="2"] {
background-color: #dc3545;
color: white;
}
```

## TestEditor.css

```
/* Стили для счетчика баллов */

.score-counter {
position: fixed;
top: 150px;
right: 20px;
width: 200px;
background-color: var(--card-bg);
padding: 1.5rem;
border-radius: var(--border-radius);
box-shadow: var(--shadow-md);
z-index: 1000;
transition: var(--transition-fast);
border: 1px solid rgba(255, 255, 255, 0.1);
}

.score-counter:hover {
box-shadow: var(--shadow-lg);
transform: translateY(-3px);
background-color: var(--card-hover);
}

.score-counter h3 {
color: var(--text-light) !important;
margin-top: 0;
margin-bottom: 1rem;
font-size: 1.3rem;
font-weight: 600;
border-bottom: 2px solid var(--accent-purple);
padding-bottom: 0.7rem;
text-align: center;
}

.score-counter-info {
```

```
margin-bottom: 1rem;
}

.total-score {
  font-size: 2.5rem;
  font-weight: bold;
  color: var(--text-light);
  text-align: center;
  display: block;
  margin: 1rem 0;
  text-shadow: 0 2px 4px rgba(0, 0, 0, 0.3);
}

.score-counter p {
  margin: 0.5rem 0;
  font-size: 0.95rem;
  color: var(--text-secondary);
}

.score-counter-valid,
.score-counter-invalid {
  display: none;
}

/* Кнопка скрытия счетчика */
.toggle-counter {
  position: fixed;
  top: 110px;
  right: 20px;
  width: 40px;
  height: 40px;
  background-color: var(--accent-purple);
  color: var(--text-light);
  border-radius: 50%;
  display: flex;
  justify-content: center;
  align-items: center;
  cursor: pointer;
  box-shadow: var(--shadow-sm);
  z-index: 1001;
  font-size: 1.5rem;
  transition: var(--transition-fast);
}
```

```
border: 1px solid rgba(255, 255, 255, 0.1);
}

.toggle-counter:hover {
background-color: var(--medium-purple);
transform: rotate(180deg);
box-shadow: var(--shadow-md);
}

/* Скрытый счетчик */
.score-counter.hidden {
transform: translateX(200px);
opacity: 0;
}

/* Адаптивность */
@media (max-width: 768px) {
.score-counter {
width: 180px;
padding: 1rem;
}
}

.total-score {
font-size: 2rem;
}

/* Дополнительные стили для улучшения читаемости */
.score-counter span {
color: var(--text-light);
}

.score-counter label {
color: var(--text-light);
}

.score-counter input {
background-color: rgba(255, 255, 255, 0.1);
border: 1px solid rgba(255, 255, 255, 0.2);
color: var(--text-light);
}
```

```
.score-counter input:focus {  
    border-color: var(--accent-purple);  
    background-color: rgba(255, 255, 255, 0.15);  
}
```

## TestMaterial.css

```
/* Темная пурпурная тема для страницы теста */  
  
:root {  
    --dark-purple: #1a1225;  
    --medium-purple: #7743DB;  
    --light-purple: #B39DDB;  
    --accent-purple: #9D4EDD;  
    --text-light: #ffffff;  
    --text-secondary: rgba(255, 255, 255, 0.85);  
    --card-bg: #272134;  
    --card-hover: #362b48;  
    --gradient-bg: linear-gradient(135deg, #1a1225 0%, #3a1d69 100%);  
    --shadow-sm: 0 4px 6px rgba(0, 0, 0, 0.3);  
    --shadow-md: 0 6px 12px rgba(0, 0, 0, 0.4);  
    --shadow-lg: 0 10px 25px rgba(0, 0, 0, 0.5);  
    --transition-slow: all 0.4s cubic-bezier(0.165, 0.84, 0.44, 1);  
    --transition-fast: all 0.2s ease;  
    --border-radius: 12px;  
    --card-spacing: 1.75rem;  
  
    /* Функциональные цвета */  
    --success: #4CAF50;  
    --success-dark: #388E3C;  
    --success-light: rgba(76, 175, 80, 0.15);  
    --danger: #F44336;  
    --danger-dark: #D32F2F;  
    --danger-light: rgba(244, 67, 54, 0.15);  
    --warning: #FFC107;  
    --warning-dark: #FFA000;  
    --warning-light: rgba(255, 193, 7, 0.15);  
    --info: #2196F3;  
    --info-dark: #1976D2;  
    --info-light: rgba(33, 150, 243, 0.15);  
}  
  
/* Принудительное переопределение базовых стилей */  
html {  
    background-color: var(--dark-purple) !important;
```

```
}
```

```
body {  
    background-color: var(--dark-purple) !important;  
    color: var(--text-light) !important;  
    font-family: 'Roboto', sans-serif;  
    margin: 0;  
    padding: 0;  
    min-height: 100vh;  
    width: 100%;  
    overflow-x: hidden; /* Предотвращает горизонтальный скролл */  
}
```

```
.container {  
    max-width: 1200px;  
    margin: 2rem auto;  
    padding: 0 1.5rem;  
    background-color: transparent !important;  
    box-shadow: none !important;  
}
```

```
/* Заголовок страницы */
```

```
.page-header {  
    background: var(--gradient-bg);  
    border-radius: var(--border-radius);  
    padding: 2rem;  
    margin-bottom: 2rem;  
    box-shadow: var(--shadow-md);  
    position: relative;  
    overflow: hidden;  
}
```

```
.page-header::before {  
    content: "";  
    position: absolute;  
    top: 0;  
    left: 0;  
    right: 0;  
    bottom: 0;  
    background: url('data:image/svg+xml;utf8,<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 100 100" preserveAspectRatio="none"><path fill="none" stroke="rgba(255,255,255,0.05)" stroke-width="1" d="M0,0 L100,100 M100,0 L0,100"/></svg>');
```

```
background-size: 20px 20px;
opacity: 0.3;
z-index: 0;
}

.header-content {
  position: relative;
  z-index: 1;
}

.page-header h1 {
  color: var(--text-light) !important;
  margin: 0;
  font-size: 2.2rem;
  font-weight: 700;
  text-shadow: 0 2px 4px rgba(0, 0, 0, 0.3);
}

.description {
  color: var(--text-light) !important;
  font-size: 1.1rem;
  margin: 1rem 0;
}

.meta-info {
  display: flex;
  flex-wrap: wrap;
  gap: 1.5rem;
  margin-top: 1rem;
  color: var(--text-light) !important;
}

.meta-item {
  display: flex;
  align-items: center;
  gap: 0.5rem;
  font-size: 1rem;
  color: var(--text-light) !important;
}

.meta-item i {
  color: var(--text-light) !important;
```

```
    font-size: 1.2rem;  
}  
  
/* Информация о тесте */  
.test-info {  
    background-color: var(--card-bg);  
    border-radius: var(--border-radius);  
    padding: 1.5rem;  
    margin-bottom: 2rem;  
    box-shadow: var(--shadow-md);  
    border: 1px solid rgba(255, 255, 255, 0.05);  
    transition: var(--transition-fast);  
}  
  
.test-info:hover {  
    box-shadow: var(--shadow-lg);  
    transform: translateY(-5px);  
}  
  
.test-info h2 {  
    color: var(--text-light) !important;  
    font-size: 1.6rem;  
    margin-top: 0;  
    margin-bottom: 1.5rem;  
    font-weight: 600;  
    padding-bottom: 0.5rem;  
    border-bottom: 2px solid var(--accent-purple);  
    position: relative;  
}  
  
.test-info h2::before {  
    content: "";  
    position: absolute;  
    width: 30px;  
    height: 30px;  
    right: 0;  
    top: 0;  
    background-color: var(--accent-purple);  
    opacity: 0.2;  
    border-radius: 50%;  
}
```

```
.test-info p {
    margin: 0.8rem 0;
    font-size: 1.05rem;
    color: var(--text-light) !important;
}

.test-status {
    margin: 1.5rem 0;
    padding: 1rem 1.2rem;
    border-radius: var(--border-radius);
    font-weight: 500;
    font-size: 1.05rem;
    color: var(--text-light) !important;
}

.test-status.passed {
    background-color: var(--success-light);
    color: var(--text-light) !important;
    border-left: 4px solid var(--success);
}

.test-status.failed {
    background-color: var(--danger-light);
    color: var(--text-light) !important;
    border-left: 4px solid var(--danger);
}

.test-status.attempts {
    background-color: var(--warning-light);
    color: var(--text-light) !important;
    border-left: 4px solid var(--warning);
}

/* Секция с вопросами */
.questions-preview {
    margin-top: 2rem;
}

.question-card {
    background-color: var(--card-bg);
    border-radius: var(--border-radius);
    padding: 1.5rem;
}
```

```
    margin-bottom: 1.5rem;  
    box-shadow: var(--shadow-sm);  
    transition: var(--transition-fast);  
    border: 1px solid rgba(255, 255, 255, 0.05);  
}
```

```
.question-card:hover {  
    transform: translateY(-3px);  
    box-shadow: var(--shadow-md);  
    background-color: var(--card-hover);  
}
```

```
.question-card h3 {  
    color: var(--text-light) !important;  
    margin-top: 0;  
    margin-bottom: 1rem;  
    font-size: 1.3rem;  
    font-weight: 600;  
    display: flex;  
    align-items: center;  
    gap: 0.5rem;  
}
```

```
.question-card h3 i {  
    font-size: 1.2rem;  
    color: var(--text-light) !important;  
}
```

```
.question-text {  
    font-size: 1.1rem;  
    margin-bottom: 1rem;  
    line-height: 1.5;  
    color: var(--text-light) !important;  
}
```

```
.question-type {  
    display: inline-block;  
    padding: 0.3rem 0.8rem;  
    background-color: rgba(157, 78, 221, 0.2);  
    color: var(--text-light) !important;  
    border-radius: 50px;  
    font-size: 0.85rem;
```

```
font-weight: 500;
margin-bottom: 0.8rem;
}

.question-points {
  display: inline-block;
  padding: 0.3rem 0.8rem;
  background-color: rgba(255, 255, 255, 0.1);
  color: var(--text-light) !important;
  border-radius: 50px;
  font-size: 0.85rem;
  font-weight: 500;
  margin-left: 0.5rem;
  margin-bottom: 0.8rem;
}

.answers {
  margin-top: 1rem;
  background-color: rgba(0, 0, 0, 0.2);
  padding: 1rem;
  border-radius: var(--border-radius);
}

.answer {
  margin: 0.7rem 0;
  padding: 0.5rem 0.8rem;
  border-radius: 6px;
  background-color: rgba(255, 255, 255, 0.05);
  border-left: 3px solid rgba(255, 255, 255, 0.2);
  color: var(--text-light) !important;
}

.answer.correct {
  border-left-color: var(--success);
  background-color: rgba(76, 175, 80, 0.1);
  color: var(--text-light) !important;
}

/* Критерии оценивания */
.grading-criteria {
  background-color: var(--card-bg);
  border-radius: var(--border-radius);
```

```
padding: 1.5rem;
margin: 2rem 0;
box-shadow: var(--shadow-md);
border: 1px solid rgba(255, 255, 255, 0.05);
transition: var(--transition-fast);
}

.grading-criteria:hover {
  box-shadow: var(--shadow-lg);
  transform: translateY(-5px);
}

.grading-criteria h2 {
  color: var(--text-light) !important;
  font-size: 1.6rem;
  margin-top: 0;
  margin-bottom: 1.5rem;
  font-weight: 600;
  padding-bottom: 0.5rem;
  border-bottom: 2px solid var(--accent-purple);
  position: relative;
}

.criteria-table {
  width: 100%;
  border-collapse: collapse;
  margin: 1rem 0;
  border-radius: 8px;
  overflow: hidden;
  box-shadow: 0 0 0 1px rgba(255, 255, 255, 0.1);
}

.criteria-table th,
.criteria-table td {
  padding: 1rem;
  text-align: center;
  border-bottom: 1px solid rgba(255, 255, 255, 0.1);
  color: var(--text-light) !important;
}

.criteria-table th {
  background-color: rgba(157, 78, 221, 0.2);
```

```
color: var(--text-light);
font-weight: 600;
}

.criteria-table tr:nth-child(odd) {
background-color: rgba(0, 0, 0, 0.2);
}

.criteria-table tr:hover {
background-color: rgba(157, 78, 221, 0.1);
}

.criteria-table strong[data-grade] {
display: inline-block;
min-width: 2rem;
padding: 0.3rem 0.6rem;
border-radius: 4px;
font-weight: 600;
text-align: center;
}

.criteria-table strong[data-grade="5"] {
background-color: var(--success);
color: white;
}

.criteria-table strong[data-grade="4"] {
background-color: var(--info);
color: white;
}

.criteria-table strong[data-grade="3"] {
background-color: var(--warning);
color: var(--dark-purple);
}

.criteria-table strong[data-grade="2"] {
background-color: var(--danger);
color: white;
}

.grade-2-row {
```

```
background-color: rgba(244, 67, 54, 0.15) !important;  
}  
  
.grade-2-row:hover {  
background-color: rgba(244, 67, 54, 0.25) !important;  
}  
  
/* Кнопки действий */  
.teacher-controls,  
.student-controls {  
display: flex;  
flex-wrap: wrap;  
justify-content: center;  
gap: 1rem;  
margin: 2rem 0;  
}  
  
.action-btn,  
.edit-btn,  
.delete-btn,  
.back-btn,  
.start-btn,  
.view-attempts-button,  
.view-results-button {  
display: inline-flex;  
align-items: center;  
justify-content: center;  
gap: 0.5rem;  
padding: 0.8rem 1.5rem;  
border-radius: 8px;  
text-decoration: none;  
font-weight: 500;  
font-size: 1rem;  
transition: all 0.2s;  
border: none;  
cursor: pointer;  
box-shadow: var(--shadow-sm);  
}  
  
.action-btn i,  
.edit-btn i,  
.delete-btn i,
```

```
.back-btn i,  
.start-btn i {  
    font-size: 1.1rem;  
}  
  
.edit-btn {  
    background-color: var(--accent-purple);  
    color: white;  
}  
  
.edit-btn:hover {  
    background-color: #8637c6;  
    transform: translateY(-3px);  
    box-shadow: var(--shadow-md);  
}  
  
.delete-btn {  
    background-color: var(--danger);  
    color: white;  
}  
  
.delete-btn:hover {  
    background-color: var(--danger-dark);  
    transform: translateY(-3px);  
    box-shadow: var(--shadow-md);  
}  
  
.back-btn {  
    background-color: rgba(255, 255, 255, 0.1);  
    color: var(--text-light);  
}  
  
.back-btn:hover {  
    background-color: rgba(255, 255, 255, 0.2);  
    transform: translateY(-3px);  
    box-shadow: var(--shadow-md);  
}  
  
.start-btn {  
    background-color: var(--success);  
    color: white;  
    font-size: 1.1rem;
```

```
    font-weight: 600;
    padding: 1rem 2rem;
}

.start-btn:hover {
    background-color: var(--success-dark);
    transform: translateY(-3px);
    box-shadow: var(--shadow-md);
}

.start-btn.disabled {
    background-color: rgba(255, 255, 255, 0.1);
    opacity: 0.7;
    cursor: not-allowed;
    transform: none !important;
    box-shadow: none !important;
    color: rgba(255, 255, 255, 0.5);
}

.view-attempts-button {
    background-color: var(--info);
    color: white;
}

.view-attempts-button:hover {
    background-color: var(--info-dark);
    transform: translateY(-3px);
    box-shadow: var(--shadow-md);
}

.view-results-button {
    background-color: var(--success);
    color: white;
}

.view-results-button:hover {
    background-color: var(--success-dark);
    transform: translateY(-3px);
    box-shadow: var(--shadow-md);
}

/* Шаблоны для результатов типа matching и order */
```

```
.matching-pair {  
    display: flex;  
    align-items: center;  
    margin: 0.8rem 0;  
    background-color: rgba(255, 255, 255, 0.05);  
    padding: 0.8rem;  
    border-radius: 6px;  
}
```

```
.matching-left,  
.matching-right {  
    flex: 1;  
    padding: 0.5rem;  
    color: var(--text-light) !important;  
}
```

```
.matching-arrow {  
    margin: 0 1rem;  
    color: var(--accent-purple);  
    font-size: 1.2rem;  
}
```

```
.order-items {  
    margin-top: 1rem;  
}
```

```
.order-item {  
    background-color: rgba(255, 255, 255, 0.05);  
    padding: 0.8rem 1rem;  
    margin: 0.5rem 0;  
    border-radius: 6px;  
    border-left: 3px solid var(--accent-purple);  
    display: flex;  
    align-items: center;  
    color: var(--text-light) !important;  
}
```

```
.order-number {  
    display: inline-flex;  
    align-items: center;  
    justify-content: center;  
    width: 2rem;
```

```
height: 2rem;  
background-color: var(--accent-purple);  
color: white;  
border-radius: 50%;  
margin-right: 1rem;  
font-weight: bold;  
}  
  
/* Адаптивность */  
@media (max-width: 768px) {  
    .container {  
        padding: 0 1rem;  
        margin: 1rem auto;  
    }  
  
    .page-header {  
        padding: 1.5rem;  
    }  
  
    .page-header h1 {  
        font-size: 1.8rem;  
    }  
  
    .meta-info {  
        flex-direction: column;  
        gap: 0.8rem;  
    }  
  
.teacher-controls,  
.student-controls {  
    flex-direction: column;  
}  
  
.action-btn,  
.edit-btn,  
.delete-btn,  
.back-btn,  
.start-btn,  
.view-attempts-button,  
.view-results-button {  
    width: 100%;  
    text-align: center;
```

```
}
```

```
}
```

## TestProcess.css

```
/* Темная пурпурная тема для страницы теста */
```

```
:root {
```

```
--dark-purple: #1a1225;
```

```
--medium-purple: #7743DB;
```

```
--light-purple: #B39DDB;
```

```
--accent-purple: #9D4EDD;
```

```
--text-light: #ffffff;
```

```
--text-secondary: rgba(255, 255, 255, 0.85);
```

```
--card-bg: #272134;
```

```
--card-hover: #362b48;
```

```
--gradient-bg: linear-gradient(135deg, #1a1225 0%, #3a1d69 100%);
```

```
--shadow-sm: 0 4px 6px rgba(0, 0, 0, 0.3);
```

```
--shadow-md: 0 6px 12px rgba(0, 0, 0, 0.4);
```

```
--shadow-lg: 0 10px 25px rgba(0, 0, 0, 0.5);
```

```
--transition-slow: all 0.4s cubic-bezier(0.165, 0.84, 0.44, 1);
```

```
--transition-fast: all 0.2s ease;
```

```
--border-radius: 12px;
```

```
--card-spacing: 1.75rem;
```

```
/* Функциональные цвета */
```

```
--success: #4CAF50;
```

```
--success-dark: #388E3C;
```

```
--success-light: rgba(76, 175, 80, 0.15);
```

```
--danger: #F44336;
```

```
--danger-dark: #D32F2F;
```

```
--danger-light: rgba(244, 67, 54, 0.15);
```

```
--warning: #FFC107;
```

```
--warning-dark: #FFA000;
```

```
--warning-light: rgba(255, 193, 7, 0.15);
```

```
--info: #2196F3;
```

```
--info-dark: #1976D2;
```

```
--info-light: rgba(33, 150, 243, 0.15);
```

```
}
```

```
/* Принудительное переопределение базовых стилей */
```

```
html {
```

```
background-color: var(--dark-purple) !important;
```

```
}
```

```
body {  
    background-color: var(--dark-purple) !important;  
    color: var(--text-light) !important;  
    font-family: 'Roboto', sans-serif;  
    margin: 0;  
    padding: 0;  
    min-height: 100vh;  
    width: 100%;  
    overflow-x: hidden; /* Предотвращает горизонтальный скролл */  
}
```

```
.container {  
    max-width: 1200px;  
    margin: 2rem auto;  
    padding: 0 1.5rem;  
    background-color: transparent !important;  
    box-shadow: none !important;  
}
```

/\* Верхняя часть с заголовком и таймером \*/

```
.test-header {  
    display: flex;  
    justify-content: space-between;  
    align-items: center;  
    margin-bottom: 30px;  
    padding: 1.5rem 2rem;  
    background: var(--gradient-bg);  
    border-radius: var(--border-radius);  
    box-shadow: var(--shadow-md);  
    position: relative;  
    overflow: hidden;  
}
```

```
.test-header::before {  
    content: "";  
    position: absolute;  
    top: 0;  
    left: 0;  
    right: 0;  
    bottom: 0;
```

```
background: url('data:image/svg+xml;utf8,<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 100 100" preserveAspectRatio="none"><path fill="none" stroke="rgba(255,255,255,0.05)" stroke-width="1" d="M0,0 L100,100 M100,0 L0,100"/></svg>');
background-size: 20px 20px;
opacity: 0.3;
z-index: 0;
}

.test-header h1 {
color: var(--text-light) !important;
margin: 0;
font-size: 1.8rem;
font-weight: 700;
text-shadow: 0 2px 4px rgba(0, 0, 0, 0.3);
position: relative;
z-index: 1;
}

.timer {
font-size: 1.2em;
font-weight: bold;
color: var(--warning);
padding: 10px 20px;
background-color: rgba(0, 0, 0, 0.3);
border-radius: var(--border-radius);
box-shadow: var(--shadow-sm);
position: relative;
z-index: 1;
border: 1px solid rgba(255, 193, 7, 0.3);
}

.timer::before {
content: "\f017"; /* Иконка часов Font Awesome */
font-family: 'Font Awesome 5 Free';
margin-right: 8px;
}

/* Вопросы */
.questions {
margin-bottom: 30px;
display: flex;
flex-direction: column;
```

```
    gap: 1.5rem;
    overflow: visible;
}

.question-card {
    background-color: var(--card-bg);
    border: 1px solid rgba(255, 255, 255, 0.05);
    border-radius: var(--border-radius);
    padding: 1.8rem;
    margin-bottom: 0.5rem;
    box-shadow: var(--shadow-sm);
    transition: var(--transition-fast);
    position: relative;
    overflow: visible;
}

.question-card:hover {
    transform: translateY(-3px);
    background-color: var(--card-hover);
    box-shadow: var(--shadow-md);
}

.question-card h3 {
    color: var(--accent-purple) !important;
    margin-top: 0;
    margin-bottom: 1rem;
    font-size: 1.4rem;
    font-weight: 600;
    border-bottom: 1px solid rgba(157, 78, 221, 0.3);
    padding-bottom: 0.7rem;
}

.question-text {
    font-size: 1.1rem;
    margin-bottom: 1.5rem;
    line-height: 1.5;
    color: var(--text-light);
}

.question-points {
    color: var(--text-secondary);
    font-size: 0.9em;
```

```
margin-bottom: 15px;  
display: inline-block;  
padding: 0.3rem 0.8rem;  
background-color: rgba(255, 255, 255, 0.1);  
border-radius: 50px;  
}  
  
/* Улучшенные стили для вариантов ответов */  
.answer-option {  
margin: 15px 0;  
background-color: rgba(255, 255, 255, 0.03);  
padding: 12px 18px;  
border-radius: 10px;  
border-left: 3px solid var(--medium-purple);  
transition: all 0.2s ease;  
position: relative;  
display: flex;  
align-items: center;  
cursor: pointer;  
}  
  
.answer-option:hover {  
background-color: rgba(157, 78, 221, 0.15);  
transform: translateX(5px);  
box-shadow: var(--shadow-sm);  
}  
  
.answer-option label {  
margin-left: 12px;  
cursor: pointer;  
color: var(--text-light);  
font-size: 1.05rem;  
font-weight: 400;  
flex: 1;  
padding: 3px 0;  
}  
  
/* Кастомные стили для радио-кнопок */  
.answer-option input[type="radio"],  
.answer-option input[type="checkbox"] {  
appearance: none;  
-webkit-appearance: none;
```

```
width: 22px;  
height: 22px;  
border: 2px solid var(--accent-purple);  
background-color: rgba(0, 0, 0, 0.3);  
position: relative;  
cursor: pointer;  
transition: all 0.2s;  
}  
  
.answer-option input[type="radio"] {  
border-radius: 50%;  
}  
  
.answer-option input[type="checkbox"] {  
border-radius: 5px;  
}  
  
.answer-option input[type="radio"]:checked,  
.answer-option input[type="checkbox"]:checked {  
background-color: var(--accent-purple);  
border-color: var(--light-purple);  
box-shadow: 0 0 8px rgba(157, 78, 221, 0.6);  
}  
  
.answer-option input[type="radio"]:checked::after {  
content: "";  
position: absolute;  
width: 10px;  
height: 10px;  
border-radius: 50%;  
background-color: white;  
top: 50%;  
left: 50%;  
transform: translate(-50%, -50%);  
}  
  
.answer-option input[type="checkbox"]:checked::after {  
content: "✓";  
position: absolute;  
color: white;  
font-size: 16px;  
font-weight: bold;
```

```
    top: 50%;  
    left: 50%;  
    transform: translate(-50%, -50%);  
}  
  
.answer-option input:focus {  
    outline: none;  
    box-shadow: 0 0 0 3px rgba(157, 78, 221, 0.4);  
}  
  
/* Контейнер, когда выбран ответ */  
.answer-option:has(input:checked) {  
    background-color: rgba(157, 78, 221, 0.2);  
    border-left-width: 5px;  
    padding-left: 16px;  
}  
  
/* Улучшенные стили для текстовых полей */  
.answer-input {  
    margin: 20px 0;  
}  
  
.answer-input input[type="text"] {  
    width: 100%;  
    padding: 14px 16px;  
    border: 2px solid rgba(157, 78, 221, 0.4);  
    border-radius: var(--border-radius);  
    background-color: rgba(0, 0, 0, 0.2);  
    color: var(--text-light);  
    font-size: 1.05rem;  
    transition: all 0.3s;  
    box-shadow: inset 0 2px 5px rgba(0, 0, 0, 0.2);  
}  
  
.answer-input input[type="text"]::placeholder {  
    color: rgba(255, 255, 255, 0.5);  
}  
  
.answer-input input[type="text"]:hover {  
    border-color: var(--accent-purple);  
    background-color: rgba(0, 0, 0, 0.25);  
}
```

```
.answer-input input[type="text"]:focus {
    border-color: var(--accent-purple);
    box-shadow: 0 0 0 3px rgba(157, 78, 221, 0.3), inset 0 2px 5px rgba(0, 0, 0, 0.1);
    outline: none;
    background-color: rgba(0, 0, 0, 0.3);
}

/* Полностью переработанные стили для вопросов на сопоставление */
.matching-container {
    margin: 20px 0;
    padding: 20px;
    border: 1px solid rgba(255, 255, 255, 0.1);
    border-radius: var(--border-radius);
    background-color: rgba(0, 0, 0, 0.15);
    position: relative;
    overflow: visible;
}

.matching-instructions {
    margin-bottom: 20px;
    color: var(--text-secondary);
    font-size: 0.95rem;
    background-color: rgba(157, 78, 221, 0.1);
    padding: 12px;
    border-radius: 8px;
    border-left: 3px solid var(--accent-purple);
}

.matching-area {
    display: flex;
    position: relative;
    margin: 20px 0;
    gap: 100px; /* Увеличиваем расстояние между колонками */
    min-height: 300px;
    justify-content: center;
    align-items: flex-start;
    padding: 0 20px; /* Добавляем отступы по бокам */
}

.matching-left-column,
.matching-right-column {
```

```
flex: 0 0 40%; /* Немного уменьшаем ширину колонок */
display: flex;
flex-direction: column;
gap: 15px;
position: relative;
z-index: 2;
max-width: 250px; /* Уменьшаем максимальную ширину */
}
```

```
.matching-canvas-container {
  position: absolute;
  top: 0;
  left: 0;
  right: 0;
  bottom: 0;
  z-index: 1;
  min-height: 300px; /* Уменьшаем минимальную высоту */
}
```

```
.matching-canvas {
  width: 100%;
  height: 100%;
  position: absolute;
  top: 0;
  left: 0;
  pointer-events: none;
}
```

```
/* Элементы сопоставления */
.matching-item {
  background-color: var(--card-bg);
  border: 1px solid rgba(255, 255, 255, 0.08);
  border-radius: 10px;
  padding: 12px 25px; /* Уменьшаем padding */
  margin: 4px 0;
  display: flex;
  align-items: center;
  min-height: 25px;
  position: relative;
  box-shadow: var(--shadow-sm);
  transition: var(--transition-fast);
}
```

```
.left-item {
    padding-right: 30px; /* Уменьшаем отступ справа */
    margin-right: 0;
    border-right: none;
    text-align: right;
}

.right-item {
    padding-left: 30px; /* Уменьшаем отступ слева */
    margin-left: 0;
    border-left: none;
    text-align: left;
}

/* Точки соединения */
.connection-point {
    width: 12px;
    height: 12px;
    border-radius: 50%;
    border: 2px solid #9D4EDD;
    cursor: pointer;
    transition: all 0.3s ease;
}

.connection-point.active {
    background-color: #9D4EDD;
    transform: scale(1.2);
}

.connection-point.connected {
    background-color: #4CAF50;
    border-color: #4CAF50;
}

/* Текст элементов */
.item-text {
    font-size: 0.95rem; /* Немного уменьшаем размер текста */
    color: var(--text-light);
    flex-grow: 1;
    padding: 5px 0;
    line-height: 1.2;
}
```

```
    user-select: none;
}

/* Контролы и счетчик */
.matching-controls {
    display: flex;
    justify-content: space-between;
    align-items: center;
    margin-top: 20px;
    padding: 10px 15px;
    background-color: rgba(0, 0, 0, 0.2);
    border-radius: 8px;
}

.reset-connections-btn {
    background-color: var(--danger);
    color: white;
    border: none;
    padding: 8px 15px;
    border-radius: 6px;
    cursor: pointer;
    font-weight: 500;
    transition: background-color 0.2s;
}

.reset-connections-btn:hover {
    background-color: var(--danger-dark);
}

.connection-counter {
    background-color: rgba(255, 255, 255, 0.1);
    color: var(--text-light);
    padding: 6px 12px;
    border-radius: 20px;
    font-weight: 500;
}

/* Линии соединения */
.connection-line {
    stroke: var(--accent-purple);
    stroke-width: 3;
    fill: none;
```

```
stroke-linecap: round;
filter: drop-shadow(0 1px 3px rgba(0, 0, 0, 0.3));
transition: all 0.3s ease;
}

.connection-line.connected {
  stroke: var(--success);
  stroke-width: 3;
}

/* Инструкции по сопоставлению */
.matching-instructions {
  background-color: rgba(0, 0, 0, 0.15);
  border-left: 3px solid var(--medium-purple);
  color: var(--text-secondary);
  padding: 10px 15px;
  margin-bottom: 20px;
  border-radius: 6px;
  font-size: 0.9rem;
}

/* Специфический стиль для корректного отображения сопоставлений с учётом текущего HTML */
.question-card .matching-area {
  margin-top: 30px;
  min-height: 200px;
  position: relative;
}

/* Предупреждение когда не все связи установлены */
.matching-container.warning {
  border: 2px solid var(--warning);
  background-color: rgba(255, 193, 7, 0.05);
  animation: warningPulse 2s infinite;
}

/* Стили для вопросов на установление порядка */
.order-container {
  margin: 20px 0;
  padding: 20px;
  border: 1px solid rgba(255, 255, 255, 0.1);
  border-radius: var(--border-radius);
  background-color: rgba(0, 0, 0, 0.15);
```

```
}

.order-instructions {
  margin-bottom: 20px;
  color: var(--text-secondary);
  font-size: 0.95rem;
  background-color: rgba(157, 78, 221, 0.1);
  padding: 12px;
  border-radius: 8px;
  border-left: 3px solid var(--accent-purple);
}

.order-items-container {
  display: flex;
  flex-direction: column;
  gap: 10px;
  min-height: 50px;
  padding: 5px;
}

.order-item {
  display: flex;
  align-items: center;
  background-color: var(--card-bg);
  border: 1px solid rgba(255, 255, 255, 0.08);
  border-radius: 8px;
  padding: 12px 15px;
  cursor: grab;
  user-select: none;
  position: relative;
  box-shadow: 0 2px 4px rgba(0, 0, 0, 0.2);
  transition: all 0.2s ease;
}

.order-item:hover {
  background-color: var(--card-hover);
  transform: translateX(3px);
}

.order-item:active {
  cursor: grabbing;
  box-shadow: 0 4px 8px rgba(0, 0, 0, 0.3);
```

```
}

.order-item.dragging {
  opacity: 0.7;
  box-shadow: 0 8px 16px rgba(0, 0, 0, 0.4);
  background-color: rgba(157, 78, 221, 0.2);
  border: 1px dashed var(--accent-purple);
  z-index: 1000;
}

.order-item.placeholder {
  background-color: rgba(157, 78, 221, 0.08);
  border: 2px dashed var(--accent-purple);
  box-shadow: none;
}

.order-number {
  display: flex;
  justify-content: center;
  align-items: center;
  width: 28px;
  height: 28px;
  background-color: var(--accent-purple);
  color: white;
  border-radius: 50%;
  font-weight: bold;
  margin-right: 15px;
  flex-shrink: 0;
}

.order-text {
  flex: 1;
  font-size: 1rem;
  color: var(--text-light);
  line-height: 1.4;
}

.order-handle {
  display: flex;
  justify-content: center;
  align-items: center;
  width: 24px;
}
```

```
height: 24px;  
margin-left: 10px;  
color: rgba(255, 255, 255, 0.6);  
font-size: 22px;  
cursor: grab;  
border-radius: 4px;  
transition: background-color 0.15s ease;  
}  
  
.order-handle:hover {  
background-color: rgba(255, 255, 255, 0.1);  
color: var(--text-light);  
}  
  
.order-item:active .order-handle {  
cursor: grabbing;  
}  
  
.order-controls {  
display: flex;  
justify-content: flex-end;  
margin-top: 15px;  
}  
  
.reset-order-btn {  
background-color: rgba(255, 255, 255, 0.1);  
color: var(--text-light);  
border: none;  
border-radius: 6px;  
padding: 8px 15px;  
font-size: 0.9rem;  
cursor: pointer;  
transition: all 0.2s ease;  
}  
  
.reset-order-btn:hover {  
background-color: rgba(255, 255, 255, 0.2);  
transform: translateY(-2px);  
}  
  
.reset-order-btn:active {  
transform: translateY(0);
```

```
}

.order-hidden-inputs {
    display: none;
}

.order-container.warning {
    border: 1px solid var(--warning);
    background-color: rgba(255, 193, 7, 0.05);
}

/* Небольшие уточнения для области сопоставления */
.matching-hidden-inputs {
    display: none;
}

/* Медиа-запросы для мобильных устройств */
@media (max-width: 768px) {
    .container {
        padding: 0 1rem;
        margin: 1rem auto;
    }

    .test-header {
        flex-direction: column;
        gap: 15px;
        padding: 1.2rem;
    }

    .test-header h1 {
        font-size: 1.5rem;
        text-align: center;
    }

    .timer {
        width: 100%;
        text-align: center;
    }

    .question-card {
        padding: 1.2rem;
    }
}
```

```
.matching-area {  
    flex-direction: column;  
    gap: 20px;  
}  
  
.matching-left-column,  
.matching-right-column {  
    flex: 0 0 auto;  
    width: 100%;  
}  
  
.matching-canvas-container {  
    display: none; /* Скрываем canvas на мобильных */  
}  
  
.matching-item {  
    margin: 5px 0;  
}  
  
.left-item, .right-item {  
    margin: 5px 0;  
}  
  
.submit-btn {  
    width: 100%;  
    padding: 12px 20px;  
}  
  
.order-item {  
    padding: 10px;  
}  
  
.order-number {  
    width: 24px;  
    height: 24px;  
    font-size: 0.9rem;  
    margin-right: 10px;  
}  
  
.order-text {  
    font-size: 0.9rem;  
}
```

```
    }

.order-handle {
    width: 20px;
    height: 20px;
}

}
```

## TestResult.css

```
/* Темная пурпурная тема для страницы результатов теста */

:root {
    --dark-purple: #1a1225;
    --medium-purple: #7743DB;
    --light-purple: #B39DDB;
    --accent-purple: #9D4EDD;
    --text-light: #ffffff;
    --text-bright: #ffffff;
    --text-secondary: rgba(255, 255, 255, 0.9);
    --card-bg: #272134;
    --card-hover: #362b48;
    --gradient-bg: linear-gradient(135deg, #1a1225 0%, #3a1d69 100%);
    --shadow-sm: 0 4px 6px rgba(0, 0, 0, 0.3);
    --shadow-md: 0 6px 12px rgba(0, 0, 0, 0.4);
    --shadow-lg: 0 10px 25px rgba(0, 0, 0, 0.5);
    --transition-slow: all 0.4s cubic-bezier(0.165, 0.84, 0.44, 1);
    --transition-fast: all 0.2s ease;
    --border-radius: 12px;
    --card-spacing: 1.75rem;
}

/* Функциональные цвета с повышенной яркостью */

--success: #4CAF50;
--success-dark: #388E3C;
--success-light: rgba(76, 175, 80, 0.15);
--success-text: #C8E6C9;
--danger: #F44336;
--danger-dark: #D32F2F;
--danger-light: rgba(244, 67, 54, 0.15);
--danger-text: #FFCDD2;
--warning: #FFC107;
--warning-dark: #FFA000;
--warning-light: rgba(255, 193, 7, 0.15);
--warning-text: #FFECB3;
--info: #2196F3;
```

```

--info-dark: #1976D2;
--info-light: rgba(33, 150, 243, 0.15);
--info-text: #B3E5FC;
}

/* Принудительное переопределение базовых стилей */
html {
    background-color: var(--dark-purple) !important;
}

body {
    font-family: 'Roboto', sans-serif;
    background-color: var(--dark-purple) !important;
    background-image: radial-gradient(circle at top right, rgba(121, 68, 154, 0.13), transparent),
                      radial-gradient(circle at bottom left, rgba(41, 10, 89, 0.13), transparent);
    color: var(--text-light) !important;
    margin: 0;
    padding: 0;
    min-height: 100vh;
    width: 100%;
    overflow-x: hidden;
}

.container {
    max-width: 1000px;
    margin: 2rem auto;
    padding: 0 1.5rem;
    background-color: transparent !important;
    box-shadow: none !important;
    color: var(--text-light);
}

h1 {
    color: var(--text-bright) !important;
    text-align: center;
    margin-bottom: 30px;
    font-size: 2.2rem;
    font-weight: 700;
    text-shadow: 0 2px 4px rgba(0, 0, 0, 0.5);
    position: relative;
    padding-bottom: 0.8rem;
    letter-spacing: 0.5px;
}

```

```
}
```

```
h1::after {  
    content: "";  
    position: absolute;  
    bottom: 0;  
    left: 50%;  
    transform: translateX(-50%);  
    width: 120px;  
    height: 3px;  
    background-color: var(--accent-purple);  
    border-radius: 3px;  
}
```

```
h2 {  
    color: var(--text-bright) !important;  
    font-size: 1.5rem;  
    font-weight: 600;  
    margin-top: 0;  
    margin-bottom: 1.5rem;  
    padding-bottom: 0.7rem;  
    position: relative;  
    letter-spacing: 0.3px;  
    text-shadow: 0 1px 2px rgba(0, 0, 0, 0.4);  
}
```

```
h2::after {  
    content: "";  
    position: absolute;  
    bottom: 0;  
    left: 0;  
    width: 70px;  
    height: 3px;  
    background-color: var(--accent-purple);  
    border-radius: 3px;  
}
```

```
/* Единый стиль для всех карточек */  
.results-card, .grading-criteria, .previous-attempts {  
    background-color: var(--card-bg);  
    border-radius: var(--border-radius);  
    padding: 1.8rem;
```

```

margin-bottom: 2rem;
box-shadow: var(--shadow-md);
border: 1px solid rgba(255, 255, 255, 0.05);
transition: var(--transition-fast);
position: relative;
overflow: hidden;
color: var(--text-light);
}

.results-card::before, .grading-criteria::before, .previous-attempts::before {
content: "";
position: absolute;
top: 0;
left: 0;
width: 100%;
height: 100%;
background: url('data:image/svg+xml;utf8,<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 100 100" preserveAspectRatio="none"><path fill="none" stroke="rgba(255,255,255,0.03)" stroke-width="1" d="M0,0 L100,100 M100,0 L0,100"/></svg>');
background-size: 20px 20px;
opacity: 0.3;
z-index: 0;
}

.results-card:hover, .grading-criteria:hover, .previous-attempts:hover {
transform: translateY(-5px);
box-shadow: var(--shadow-lg);
background-color: var(--card-hover);
}

/* Содержимое карточек */

.score-info {
display: flex;
flex-direction: column;
gap: 20px;
position: relative;
z-index: 1;
}

.score, .grade, .completion-time {
display: flex;
align-items: center;

```

```
    font-size: 1.15rem;
    color: var(--text-light);
    line-height: 1.6;
}

.score::before, .grade::before, .completion-time::before {
    font-family: 'Font Awesome 5 Free';
    font-weight: 900;
    margin-right: 12px;
    font-size: 1.2rem;
    color: var(--light-purple);
    width: 25px;
    text-align: center;
}

.score::before { content: "\f005"; /* звезда */ }
.grade::before { content: "\f091"; /* трофей */ }
.completion-time::before { content: "\f017"; /* часы */ }

.score strong, .completion-time strong {
    color: var(--text-bright);
    font-weight: 600;
    margin-left: 5px;
}

.grade strong {
    font-size: 1.3rem;
    padding: 6px 12px;
    border-radius: 6px;
    display: inline-block;
    min-width: 30px;
    text-align: center;
    margin-left: 5px;
    box-shadow: var(--shadow-sm);
}

/* Стили для разных оценок с повышенной контрастностью */
.grade strong[data-grade="5"] {
    background-color: var(--success);
    color: white !important;
    text-shadow: 0 1px 1px rgba(0, 0, 0, 0.3);
}
```

```
.grade strong[data-grade="4"] {  
    background-color: var(--info);  
    color: white !important;  
    text-shadow: 0 1px 1px rgba(0, 0, 0, 0.3);  
}  
  
.grade strong[data-grade="3"] {  
    background-color: var(--warning);  
    color: var(--dark-purple) !important;  
    font-weight: 700;  
}  
  
.grade strong[data-grade="2"] {  
    background-color: var(--danger);  
    color: white !important;  
    text-shadow: 0 1px 1px rgba(0, 0, 0, 0.3);  
}  
  
/* Информация о пересдаче со светлым текстом */  
.retake-info {  
    margin-top: 25px;  
    padding: 18px;  
    border-radius: 10px;  
    font-weight: 500;  
    font-size: 1.05rem;  
    position: relative;  
    box-shadow: var(--shadow-sm);  
}  
  
.retake-info p {  
    margin: 0;  
    line-height: 1.5;  
}  
  
.retake-info::before {  
    font-family: 'Font Awesome 5 Free';  
    font-weight: 900;  
    margin-right: 12px;  
    font-size: 1.2rem;  
}
```

```
.retake-info.success {  
    background-color: #1e392a; /* Темно-зеленый фон */  
    color: var(--success-text); /* Светлый зеленый текст */  
    border-left: 4px solid var(--success);  
    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.3); /* Добавление тени для выделения */  
}  
  
.retake-info.success::before {  
    content: "\f058"; /* галочка в круге */  
    color: var(--success); /* Сохраняем зеленый цвет для иконки */  
}  
  
.retake-info.success p {  
    color: #c8ebc9; /* Светло-зеленый текст для лучшей видимости */  
    font-weight: 500; /* Немного жирнее для лучшей читаемости */  
}  
  
.retake-info.warning {  
    background-color: var(--card-hover);  
    color: var(--warning-text);  
    border-left: 4px solid var(--warning);  
}  
  
.retake-info.warning::before {  
    content: "\f071"; /* предупреждение */  
    color: var(--warning);  
}  
  
.retake-info.danger {  
    /* background-color: var(--danger-light); */  
    color: var(--danger-text);  
    border-left: 4px solid var(--danger);  
}  
  
.retake-info.danger::before {  
    content: "\f057"; /* X в круге */  
    color: var(--danger);  
}  
  
.retake-btn {  
    display: inline-flex;  
    align-items: center;
```

```
margin-top: 15px;  
padding: 10px 20px;  
background-color: var(--warning);  
color: var(--dark-purple);  
text-decoration: none;  
border-radius: 8px;  
font-weight: 700;  
transition: all 0.2s;  
box-shadow: var(--shadow-sm);  
}
```

```
.retake-btn::before {  
content: "\f2f1"; /* иконка обновления */  
font-family: 'Font Awesome 5 Free';  
font-weight: 900;  
margin-right: 10px;  
}
```

```
.retake-btn:hover {  
background-color: var(--warning-dark);  
transform: translateY(-3px);  
box-shadow: var(--shadow-md);  
}
```

```
/* Таблицы со светлым текстом */  
table {  
width: 100%;  
border-collapse: separate;  
border-spacing: 0;  
margin-top: 20px;  
border-radius: 10px;  
overflow: hidden;  
box-shadow: var(--shadow-sm);  
color: var(--text-light);  
}
```

```
table th, table td {  
padding: 15px;  
text-align: left;  
border-bottom: 1px solid rgba(255, 255, 255, 0.07);  
position: relative;  
}
```

```
table th {  
    background-color: rgba(157, 78, 221, 0.2);  
    color: var(--text-bright);  
    font-weight: 600;  
    text-transform: uppercase;  
    font-size: 0.85rem;  
    letter-spacing: 0.5px;  
}  
  
table tr:last-child td {  
    border-bottom: none;  
}
```

```
table tr:nth-child(odd) {  
    background-color: rgba(0, 0, 0, 0.2);  
}
```

```
table tr:hover {  
    background-color: rgba(157, 78, 221, 0.1);  
}
```

```
table td {  
    color: var(--text-light);  
}
```

```
table td strong[data-grade] {  
    display: inline-flex;  
    align-items: center;  
    justify-content: center;  
    min-width: 35px;  
    min-height: 35px;  
    padding: 5px;  
    border-radius: 50%;  
    font-weight: 700;  
    box-shadow: var(--shadow-sm);  
}
```

```
.criteria-table {  
    margin-top: 20px;  
    overflow-x: auto;  
    position: relative;
```

```
        z-index: 1;
        padding-bottom: 5px;
    }

.criteria-table table {
    width: 100%;
    margin-bottom: 0;
    box-shadow: 0 0 0 1px rgba(255, 255, 255, 0.1);
}

.criteria-table th {
    text-align: center;
    padding: 15px 20px;
}

.criteria-table td {
    text-align: center;
    padding: 12px 20px;
    border: 1px solid rgba(255, 255, 255, 0.07);
}

.criteria-table strong[data-grade] {
    display: inline-flex;
    align-items: center;
    justify-content: center;
    width: 40px;
    height: 40px;
    border-radius: 50%;
    font-weight: 700;
    font-size: 1.2rem;
    box-shadow: var(--shadow-sm);
}

.grade-2-row {
    background-color: rgba(220, 53, 69, 0.1) !important;
}

.grade-2-row:hover {
    background-color: rgba(220, 53, 69, 0.2) !important;
}

.grade-2-row td {
```

```
color: var(--danger-text) !important;
}

/* Кнопки со светлым текстом */
.actions {
    display: flex;
    flex-wrap: wrap;
    gap: 1.2rem;
    justify-content: center;
    margin-top: 40px;
}

.actions a {
    display: inline-flex;
    align-items: center;
    padding: 12px 25px;
    border-radius: 10px;
    text-decoration: none;
    font-weight: 600;
    transition: all 0.25s;
    box-shadow: var(--shadow-sm);
    letter-spacing: 0.3px;
}

.actions a::before {
    font-family: 'Font Awesome 5 Free';
    font-weight: 900;
    margin-right: 10px;
}

.actions a:first-child {
    background-color: rgba(255, 255, 255, 0.1);
    color: var(--text-bright) !important;
}

.actions a:first-child::before {
    content: "\f060"; /* стрелка влево */
}

.actions a:first-child:hover {
    background-color: rgba(255, 255, 255, 0.2);
    transform: translateY(-3px);
}
```

```
    box-shadow: var(--shadow-md);  
}  
  
.button {  
    background-color: var(--accent-purple) !important;  
    color: white !important;  
}  
  
.button::before {  
    content: "\f0a8" !important; /* стрелка назад */  
}  
  
.button:hover {  
    background-color: #8637c6 !important;  
    transform: translateY(-3px);  
    box-shadow: var(--shadow-md);  
}  
  
/* Адаптивность для мобильных устройств */  
@media (max-width: 768px) {  
    .container {  
        padding: 0 1rem;  
        margin: 1rem auto;  
    }  
  
    h1 {  
        font-size: 1.8rem;  
    }  
  
    h2 {  
        font-size: 1.4rem;  
    }  
  
.results-card, .grading-criteria, .previous-attempts {  
    padding: 1.5rem;  
}  
  
.score-info {  
    gap: 15px;  
}  
  
.score, .grade, .completion-time {
```

```
    font-size: 1.05rem;
```

```
}
```

```
.retake-info {  
    padding: 15px;  
    font-size: 0.95rem;  
}
```

```
.actions {  
    flex-direction: column;  
    gap: 1rem;  
}
```

```
.actions a {  
    width: 100%;  
    justify-content: center;  
}
```

```
table {  
    display: block;  
    overflow-x: auto;  
}
```

```
table th, table td {  
    padding: 12px 10px;  
}
```

```
.criteria-table th, .criteria-table td {  
    padding: 10px;  
}  
}
```

## TestStudentResults.css

```
body {  
    font-family: 'Roboto', sans-serif;  
    background-color: var(--dark-purple);  
    margin: 0;  
    padding: 0;  
    color: var(--text-light);  
}
```

```
.container {  
    max-width: 1200px;
```

```
margin: 2rem auto;
padding: 0 1.5rem;
}

h1, h2, h3, h4, p {
  color: var(--text-light);
}

h2 {
  font-size: 1.5rem;
  margin: 1.5rem 0;
  border-bottom: 2px solid var(--accent-purple);
  padding-bottom: 0.5rem;
}

/* Карточки и контейнеры */
.statistics-card, .test-info, .grading-criteria, .results-summary, .students-results {
  background-color: var(--card-bg);
  border-radius: var(--border-radius);
  padding: 1.5rem;
  margin-bottom: 2rem;
  box-shadow: var(--shadow-md);
}

/* Статистика */
.stats-grid {
  display: grid;
  grid-template-columns: repeat(auto-fit, minmax(200px, 1fr));
  gap: 1.5rem;
  margin-top: 1rem;
}

.stat-item {
  background-color: rgba(255, 255, 255, 0.05);
  padding: 1.5rem;
  border-radius: var(--border-radius);
  text-align: center;
  color: var(--text-light);
}

.stat-value {
  font-size: 2.5rem;
```

```
    font-weight: bold;
    color: var(--text-light);
}

.stat-label {
    color: var(--text-secondary);
}

/* Карточки студентов */
.student-card {
    background-color: var(--card-bg);
    border-radius: var(--border-radius);
    padding: 1.5rem;
    margin-bottom: 1rem;
    transition: var(--transition-fast);
    color: var(--text-light);
}

.student-card:hover {
    transform: translateY(-2px);
    background-color: var(--card-hover);
}

.student-card h3 {
    color: var(--text-light) !important;
}

/* Таблица попыток */
.attempts-table {
    width: 100%;
    border-collapse: collapse;
    margin-top: 1rem;
    background-color: var(--card-bg);
    border-radius: var(--border-radius);
    overflow: hidden;
}

.attempts-table th,
.attempts-table td {
    padding: 1rem;
    text-align: left;
    border-bottom: 1px solid rgba(255, 255, 255, 0.1);
}
```

```

    color: var(--text-light) !important;
}

/* Специальные стили для колонок даты и времени */
.attempts-table td:nth-child(2), /* Колонка даты */
.attempts-table td:nth-child(3) { /* Колонка времени */
    font-family: 'Consolas', monospace;
    white-space: nowrap;
    color: var(--text-light) !important;
    background-color: rgba(255, 255, 255, 0.05);
}

/* Заголовки таблицы */
.attempts-table thead th {
    background-color: rgba(255, 255, 255, 0.1);
    font-weight: 600;
    padding: 1rem;
    color: var(--text-light) !important;
    text-transform: uppercase;
    font-size: 0.9rem;
    letter-spacing: 0.5px;
}

/* Стили для строк при наведении */
.attempts-table tbody tr:hover {
    background-color: rgba(255, 255, 255, 0.05);
}

/* Стили для пустых значений */
.attempts-table td.no-data {
    color: rgba(255, 255, 255, 0.5) !important;
    font-style: italic;
}

/* Статусы и оценки */
.grade-cell.passed {
    color: var(--success);
}

.grade-cell.failed {
    color: var(--danger);
}

```

```
.no-attempts {  
    color: var(--text-secondary);  
    font-style: italic;  
}  
  
/* Кнопки */  
.button {  
    display: inline-block;  
    padding: 0.75rem 1.5rem;  
    background-color: var(--accent-purple);  
    color: var(--text-light);  
    text-decoration: none;  
    border-radius: var(--border-radius);  
    font-weight: 500;  
    transition: var(--transition-fast);  
}  
  
.button:hover {  
    background-color: var(--medium-purple);  
    transform: translateY(-2px);  
    box-shadow: var(--shadow-md);  
}  
  
/* Пустое состояние */  
.empty-state {  
    text-align: center;  
    padding: 3rem;  
    background-color: var(--card-bg);  
    border-radius: var(--border-radius);  
    color: var(--text-light);  
}  
  
.empty-state i {  
    font-size: 3rem;  
    color: var(--accent-purple);  
    margin-bottom: 1rem;  
}  
  
/* Информация о попытках */  
.attempt-info {  
    display: flex;
```

```
gap: 1.5rem;
flex-wrap: wrap;
color: var(--text-light);
}

.attempt-info span {
  display: flex;
  align-items: center;
  gap: 0.5rem;
  color: var(--text-light);
}

.attempts-count, .best-score {
  color: var(--text-light);
}

/* Адаптивность */
@media (max-width: 768px) {
  .container {
    padding: 0 1rem;
  }
}

.student-info {
  flex-direction: column;
  align-items: flex-start;
}

.attempt-info {
  flex-direction: column;
  gap: 0.5rem;
}

.attempts-table {
  font-size: 0.9rem;
}

/* Стили для таблицы критериев оценивания */
.criteria-table {
  width: 100%;
  border-collapse: collapse;
  margin: 15px 0;
}
```

```
}

.criteria-table th, .criteria-table td {
    padding: 10px;
    text-align: center;
    border: 1px solid #ddd;
}

.criteria-table th {
    background-color: #f2f2f2;
    font-weight: bold;
}

.criteria-table tr:hover {
    background-color: #f5f5f5;
}

.grade-2-row {
    background-color: rgba(220, 53, 69, 0.1);
}

.grade-2-row:hover {
    background-color: rgba(220, 53, 69, 0.2);
}

/* Стили для оценок в таблице критериев */
.criteria-table strong[data-grade], .results-table strong[data-grade] {
    padding: 3px 8px;
    border-radius: 4px;
    display: inline-block;
    min-width: 20px;
    text-align: center;
    font-weight: bold;
}

strong[data-grade="5"] {
    background-color: #28a745;
    color: white;
}

strong[data-grade="4"] {
    background-color: #17a2b8;
```

```
    color: white;  
}  
  
strong[data-grade="3"] {  
    background-color: #ffc107;  
    color: #333;  
}  
  
strong[data-grade="2"] {  
    background-color: #dc3545;  
    color: white;  
}  
  
/* Стили для сводки результатов */  
.summary-cards {  
    display: flex;  
    justify-content: space-between;  
    flex-wrap: wrap;  
    gap: 1rem;  
    margin-top: 1rem;  
}  
  
.summary-card {  
    flex: 1;  
    min-width: 200px;  
    background-color: var(--card-bg);  
    border-radius: 8px;  
    padding: 1.5rem;  
    text-align: center;  
    box-shadow: 0 2px 4px rgba(0, 0, 0, 0.05);  
    transition: transform 0.2s;  
    color: var(--text-light);  
}  
  
.summary-card:hover {  
    transform: translateY(-5px);  
    box-shadow: 0 5px 15px rgba(0, 0, 0, 0.1);  
}  
  
.summary-value {  
    font-size: 2.5rem;  
    font-weight: bold;  
}
```

```
color: var(--text-light);
margin-bottom: 0.5rem;
}

.summary-label {
    color: var(--text-secondary);
    font-size: 1rem;
}

/* Стили для таблицы результатов */
.results-table {
    width: 100%;
    border-collapse: collapse;
    margin-top: 1rem;
}

.results-table th, .results-table td {
    padding: 12px;
    text-align: left;
    border-bottom: 1px solid #ddd;
    color: var(--text-light);
}

.results-table th {
    background-color: rgba(255, 255, 255, 0.1);
    font-weight: bold;
    color: var(--text-light);
}

.results-table tr:hover {
    background-color: #f5f5f5;
}

/* Стили для строк с разными статусами */
.results-table tr.passed {
    background-color: rgba(40, 167, 69, 0.05);
}

.results-table tr.passed:hover {
    background-color: rgba(40, 167, 69, 0.1);
}
```

```
.results-table tr.failed {
    background-color: rgba(220, 53, 69, 0.05);
}

.results-table tr.failed:hover {
    background-color: rgba(220, 53, 69, 0.1);
}

.results-table tr.not-attempted {
    background-color: rgba(108, 117, 125, 0.05);
}

.results-table tr.not-attempted:hover {
    background-color: rgba(108, 117, 125, 0.1);
}

/* Стили для статусов */

.status {
    display: inline-block;
    padding: 5px 10px;
    border-radius: 4px;
    font-weight: bold;
    font-size: 0.9rem;
}

.status.passed {
    background-color: var(--success);
    color: var(--text-light);
}

.status.failed {
    background-color: var(--danger);
    color: var(--text-light);
}

.status.not-attempted {
    background-color: rgba(255, 255, 255, 0.2);
    color: var(--text-light);
}

/* Стили для кнопок действий */

.actions {
```

```
    display: flex;
    justify-content: center;
    gap: 1rem;
    margin-top: 2rem;
    flex-wrap: wrap;
}
```

```
.no-results {
    text-align: center;
    color: var(--text-secondary);
    font-style: italic;
    padding: 2rem;
}
```

```
/* Основные стили */
```

```
.student-info {
    display: flex;
    justify-content: space-between;
    align-items: center;
    flex-wrap: wrap;
    gap: 1rem;
    color: var(--text-light);
}
```

```
/* Список попыток */
```

```
.attempts-list {
    margin-top: 1rem;
    padding-top: 1rem;
    border-top: 1px solid rgba(255, 255, 255, 0.1);
}
```

```
.attempts-list h4 {
    color: var(--text-light);
    margin-bottom: 1rem;
}
```

```
.grade-cell {
    font-weight: bold;
    color: var(--text-light);
}
```

```
/* Заголовки */
```

```
.page-header h1,  
.page-header .description,  
.students-results h2 {  
    color: var(--text-light);  
}  
  
.no-attempts {  
    color: var(--text-secondary);  
    font-style: italic;  
}  
  
/* Обновляем все текстовые элементы на белый цвет */  
body,  
h1, h2, h3, h4, h5, h6,  
p, span, div, td, th,  
.student-info,  
.attempt-info,  
.attempts-count,  
.best-score,  
.grade,  
.no-attempts,  
.empty-state p,  
.stat-label,  
.stat-value,  
.summary-value,  
.summary-label,  
.description,  
.student-card h3,  
.attempts-list h4,  
.no-data,  
.status {  
    color: var(--text-light) !important;  
}  
  
/* Обновляем цвета для таблицы */  
.attempts-table th,  
.attempts-table td {  
    color: var(--text-light) !important;  
}  
  
/* Обновляем цвета для статистики */  
.stat-item {
```

```
    color: var(--text-light) !important;
}

.stat-item .stat-value,
.stat-item .stat-label {
    color: var(--text-light) !important;
}

/* Обновляем цвета для карточек студентов */
.student-card {
    color: var(--text-light) !important;
}

.student-card * {
    color: var(--text-light) !important;
}

/* Обновляем цвета для попыток */
.attempt-info span {
    color: var(--text-light) !important;
}

/* Обновляем цвета для статусов, сохраняя цветовое кодирование */
.status.passed {
    background-color: var(--success);
    color: var(--text-light) !important;
}

.status.failed {
    background-color: var(--danger);
    color: var(--text-light) !important;
}

.status.not-attempted {
    background-color: rgba(255, 255, 255, 0.2);
    color: var(--text-light) !important;
}

/* Обновляем цвета для оценок */
.grade-cell {
    color: var(--text-light) !important;
}
```

```

.grade-cell.passed {
    color: var(--success) !important;
}

.grade-cell.failed {
    color: var(--danger) !important;
}

/* Обновляем цвета для пустых состояний */
.no-data,
.no-attempts {
    color: rgba(255, 255, 255, 0.7) !important;
}

/* Обновляем цвета для заголовков таблицы */
.attempts-table thead th {
    color: var(--text-light) !important;
    background-color: rgba(255, 255, 255, 0.1);
}

/* Обновляем цвета для статистики */
.statistics-card h2,
.statistics-card .stat-label,
.statistics-card .stat-value {
    color: var(--text-light) !important;
}

/* Обновляем цвета для описания */
.page-header .description {
    color: var(--text-light) !important;
}

/* Обновляем цвета для иконок */
.fas {
    color: var(--text-light) !important;
}

```

## TestStyles.css

```

:root {
    --dark-purple: #1a1225;
    --medium-purple: #7743DB;
    --light-purple: #B39DDB;
}

```

```
--accent-purple: #9D4EDD;  
--text-light: #ffffff;  
--text-secondary: rgba(255, 255, 255, 0.85);  
--card-bg: #272134;  
--card-hover: #362b48;  
--gradient-bg: linear-gradient(135deg, #1a1225 0%, #3a1d69 100%);  
--shadow-sm: 0 4px 6px rgba(0, 0, 0, 0.3);  
--shadow-md: 0 6px 12px rgba(0, 0, 0, 0.4);  
--border-radius: 12px;  
--danger: #dc3545;  
--danger-hover: #c82333;  
}
```

```
html, body {  
background-color: var(--dark-purple) !important;  
color: var(--text-light) !important;  
font-family: 'Roboto', sans-serif;  
margin: 0;  
padding: 0;  
min-height: 100vh;  
}
```

```
.container {  
max-width: 1200px;  
margin: 2rem auto;  
padding: 0 2rem;  
background-color: transparent !important;  
box-shadow: none !important;  
}
```

```
/* Заголовок */  
h1 {  
font-size: 2.5rem;  
font-weight: 700;  
text-align: center;  
color: var(--text-light) !important;  
margin-bottom: 2rem;  
padding: 1.5rem;  
background: var(--gradient-bg);  
border-radius: var(--border-radius);  
box-shadow: var(--shadow-md);  
}
```

```
/* Форма создания теста */

.test-form-container {
    background-color: var(--card-bg);
    border-radius: var(--border-radius);
    padding: 2rem;
    margin-bottom: 2rem;
    box-shadow: var(--shadow-md);
    border: 1px solid rgba(255, 255, 255, 0.05);
}

.test-form {
    display: flex;
    flex-direction: column;
    gap: 1.5rem;
}

.form-group {
    display: flex;
    flex-direction: column;
    gap: 0.5rem;
}

.form-group label {
    color: var(--text-light);
    font-size: 1.1rem;
    font-weight: 500;
    display: flex;
    align-items: center;
    gap: 0.5rem;
}

.form-group label i {
    color: var(--accent-purple);
    width: 20px;
}

.form-group input,
.form-group textarea,
.form-group select {
    background-color: rgba(255, 255, 255, 0.05);
    border: 1px solid rgba(255, 255, 255, 0.1);
}
```

```
border-radius: var(--border-radius);
padding: 0.8rem 1rem;
color: var(--text-light);
font-size: 1rem;
transition: all 0.3s ease;
width: 100%;
box-sizing: border-box;
}

.form-group textarea {
min-height: 150px;
resize: vertical;
line-height: 1.5;
}

.form-group input:focus,
.form-group textarea:focus,
.form-group select:focus {
outline: none;
border-color: var(--accent-purple);
box-shadow: 0 0 0 2px rgba(157, 78, 221, 0.2);
}

/* Группа кнопок */
.button-group {
display: flex;
gap: 1rem;
margin: 1rem 0 2rem;
align-items: center;
flex-wrap: wrap;
}

/* Кнопка "Назад" */
.back-button {
display: inline-flex;
align-items: center;
gap: 0.5rem;
padding: 0.7rem 1.2rem;
background: var(--dark-purple);
color: white !important;
text-decoration: none;
border-radius: var(--border-radius);
```

```
    font-size: 1rem;
    font-weight: 500;
    transition: all 0.3s ease;
    border: 1px solid rgba(255, 255, 255, 0.1);
    box-shadow: var(--shadow-sm);
}
```

```
.back-button:hover {
    background: var(--card-hover);
    transform: translateY(-2px);
    box-shadow: var(--shadow-md);
}
```

```
/* Кнопка сохранения */
```

```
.save-button {
    display: inline-flex;
    align-items: center;
    gap: 0.5rem;
    padding: 0.7rem 1.2rem;
    background: var(--medium-purple);
    color: white !important;
    border: none;
    border-radius: var(--border-radius);
    font-size: 1rem;
    font-weight: 500;
    cursor: pointer;
    transition: all 0.3s ease;
    text-decoration: none;
    border: 1px solid rgba(255, 255, 255, 0.1);
    box-shadow: var(--shadow-sm);
}
```

```
.save-button:hover {
    background: var(--accent-purple);
    transform: translateY(-2px);
    box-shadow: var(--shadow-md);
}
```

```
/* Сообщение об ошибке */
```

```
.error-message {
    background-color: rgba(220, 53, 69, 0.1);
    color: #ff6b6b;
```

```
padding: 1rem 1.5rem;  
border-radius: var(--border-radius);  
margin-bottom: 1.5rem;  
display: flex;  
align-items: center;  
gap: 0.5rem;  
border-left: 4px solid #ff6b6b;  
}
```

```
.error-message i {  
color: #ff6b6b;  
}
```

```
/* Адаптивность */  
@media (max-width: 768px) {  
.container {  
padding: 0 1rem;  
}}
```

```
h1 {  
font-size: 2rem;  
padding: 1.2rem;  
}
```

```
.test-form-container {  
padding: 1.5rem;  
}
```

```
.form-group input,  
.form-group textarea,  
.form-group select {  
font-size: 16px;  
}
```

```
.button-group {  
flex-direction: column;  
width: 100%;  
}
```

```
.back-button,  
.save-button {  
width: 100%;
```

```
    justify-content: center;
}
}
```

## UserCRUD.css

```
:root {
    --dark-purple: #1a1225;
    --medium-purple: #7743DB;
    --light-purple: #B39DDB;
    --accent-purple: #9D4EDD;
    --text-light: #ffffff;
    --text-secondary: rgba(255, 255, 255, 0.85);
    --card-bg: #272134;
    --card-hover: #362b48;
    --gradient-bg: linear-gradient(135deg, #1a1225 0%, #3a1d69 100%);
    --shadow-sm: 0 4px 6px rgba(0, 0, 0, 0.3);
    --shadow-md: 0 6px 12px rgba(0, 0, 0, 0.4);
    --shadow-lg: 0 10px 25px rgba(0, 0, 0, 0.5);
    --border-radius: 12px;
}

/* Глобальные стили */
*, *::before, *::after {
    color: var(--text-light) !important;
}

body {
    font-family: 'Roboto', sans-serif;
    background-color: var(--dark-purple) !important;
    margin: 0;
    min-height: 100vh;
}

.container {
    max-width: 1200px;
    margin: 2rem auto;
    padding: 2rem;
    background: var(--gradient-bg) !important;
    border-radius: var(--border-radius);
    box-shadow: var(--shadow-md);
    position: relative;
    overflow: visible;
}
```

```
.container::before {  
    content: "";  
    position: absolute;  
    top: 0;  
    left: 0;  
    right: 0;  
    bottom: 0;  
    background: url('data:image/svg+xml;utf8,<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0  
100 100" preserveAspectRatio="none"><path fill="none" stroke="rgba(255,255,255,0.05)" stroke-width="1"  
d="M0,0 L100,100 M100,0 L0,100"/></svg>');  
    background-size: 20px 20px;  
    opacity: 0.3;  
    z-index: 0;  
}  
  
/* Заголовки */  
h1, h2, h3 {  
    color: var(--text-light) !important;  
    text-align: center;  
    margin-bottom: 2rem;  
    position: relative;  
    z-index: 1;  
}  
  
h1 {  
    font-size: 2.25rem;  
    font-weight: 700;  
}  
  
/* Поиск и фильтры */  
.search-options,  
.filter-options {  
    background-color: var(--card-bg);  
    padding: 1.25rem;  
    border-radius: var(--border-radius);  
    margin-bottom: 2rem;  
    position: relative;  
    z-index: 2;  
}  
  
.search-options form,
```

```
.filter-options form {
    display: flex;
    gap: 1rem;
    align-items: center;
}

.filter-options label {
    white-space: nowrap;
    margin-right: 0.5rem;
}

input[type="text"],
select {
    width: 100%;
    padding: 0.75rem;
    background-color: rgba(255, 255, 255, 0.05);
    border: 1px solid rgba(255, 255, 255, 0.1);
    border-radius: var(--border-radius);
    color: var(--text-light) !important;
    cursor: pointer;
}

input[type="text"]:focus,
select:focus {
    outline: none;
    border-color: var(--accent-purple);
    box-shadow: 0 0 0 2px rgba(157, 78, 221, 0.2);
}

/* Кнопки */
button {
    padding: 0.75rem 1rem;
    background-color: var(--medium-purple);
    color: var(--text-light) !important;
    border: none;
    border-radius: var(--border-radius);
    cursor: pointer;
    transition: all 0.2s ease;
    position: relative;
    overflow: hidden;
}
```

```
button:hover {  
    background-color: var(--accent-purple);  
    transform: translateY(-2px);  
}  
  
button:active {  
    transform: translateY(0);  
}  
  
/* Список пользователей */  
.user-list {  
    display: grid;  
    grid-template-columns: repeat(auto-fill, minmax(300px, 1fr));  
    gap: 1.5rem;  
    margin-top: 2rem;  
    max-height: none;  
    overflow-y: visible;  
}  
  
.user-card {  
    background-color: var(--card-bg);  
    border: 1px solid rgba(255, 255, 255, 0.1);  
    border-radius: var(--border-radius);  
    padding: 1.5rem;  
    display: flex;  
    flex-direction: column;  
    gap: 1rem;  
    transition: transform 0.3s ease, box-shadow 0.3s ease;  
    word-break: break-word;  
}  
  
.user-info {  
    padding-bottom: 1rem;  
    border-bottom: 1px solid rgba(255, 255, 255, 0.1);  
}  
  
.user-info h3 {  
    font-size: 1.2rem;  
    margin-bottom: 1rem;  
    color: var(--text-light) !important;  
    word-wrap: break-word;  
}
```

```
.user-info p {  
    margin: 0.5rem 0;  
    display: flex;  
    flex-direction: column;  
    gap: 0.25rem;  
}  
  
.user-info p strong {  
    min-width: 80px;  
}  
  
/* Формы редактирования */  
.crud-form {  
    display: flex;  
    flex-direction: column;  
    gap: 0.75rem;  
}  
  
.crud-form input,  
.crud-form select {  
    width: 100%;  
    padding: 0.75rem;  
    background-color: rgba(255, 255, 255, 0.05);  
    border: 1px solid rgba(255, 255, 255, 0.1);  
    border-radius: var(--border-radius);  
    color: var(--text-light) !important;  
    cursor: pointer;  
}  
  
.crud-form button,  
.search-options button {  
    background-color: var(--medium-purple);  
    color: var(--text-light) !important;  
    padding: 0.75rem 1.5rem;  
    border-radius: var(--border-radius);  
    border: none;  
    cursor: pointer;  
    transition: all 0.3s ease;  
    font-weight: 500;  
}
```

```
.crud-form button:hover,  
.search-options button:hover {  
    background-color: var(--accent-purple);  
    transform: translateY(-2px);  
    box-shadow: var(--shadow-md);  
}  
  
.delete-button {  
    background-color: #dc3545;  
    margin-top: 0.5rem;  
    width: 100%;  
}  
  
.delete-button:hover {  
    background-color: #c82333;  
}  
  
/* Кнопка "Назад" */  
.header-section {  
    position: relative;  
    margin-bottom: 2rem;  
    z-index: 2;  
}  
  
.back-button {  
    display: inline-flex;  
    align-items: center;  
    gap: 0.75rem;  
    padding: 0.75rem 1.5rem;  
    background-color: var(--medium-purple);  
    color: var(--text-light) !important;  
    border-radius: var(--border-radius);  
    text-decoration: none;  
    font-weight: 500;  
    transition: all 0.3s ease;  
    border: 1px solid rgba(255, 255, 255, 0.1);  
    margin-bottom: 1rem;  
}  
  
.back-button:hover {  
    background-color: var(--accent-purple);  
    transform: translateY(-2px);
```

```
    box-shadow: var(--shadow-md);
    text-decoration: none;
}

.back-button i {
    font-size: 1.1rem;
}

/* Удаляем стили для сортировки */
.sort-options {
    display: none;
}

/* Обновляем отступы после удаления сортировки */
.filter-options {
    margin-bottom: 2rem;
}

/* Анимации */
@keyframes fadeIn {
    from {
        opacity: 0;
        transform: translateY(20px);
    }
    to {
        opacity: 1;
        transform: translateY(0);
    }
}

/* Адаптивность */
@media (max-width: 768px) {
    .container {
        margin: 1rem;
        padding: 1rem;
    }
}

.search-options form,
.filter-options form {
    flex-direction: column;
}
```

```
.search-options input,  
.filter-options select {  
    width: 100%;  
}  
  
.user-card {  
    padding: 1rem;  
}  
  
.user-info h3 {  
    font-size: 1.1rem;  
}  
}  
  
/* Добавляем стили для выпадающих списков */  
select {  
    appearance: none;  
    background-image: url("data:image/svg+xml; charset=UTF-8,%3csvg  
xmlns='http://www.w3.org/2000/svg' viewBox='0 0 24 24' fill='white'%3e%3cpath d='M7 10l5 5-5z'%3e%3c/svg%3e");  
    background-repeat: no-repeat;  
    background-position: right 0.7rem center;  
    background-size: 1.5em;  
    padding-right: 2.5rem;  
}  
  
/* Добавляем стили для фокуса на интерактивных элементах */  
input:focus,  
select:focus,  
button:focus {  
    outline: none;  
    border-color: var(--accent-purple);  
    box-shadow: 0 0 0 2px rgba(157, 78, 221, 0.2);  
}  
  
/* Добавляем стили для наведения на карточки */  
.user-card:hover {  
    transform: translateY(-5px);  
    box-shadow: var(--shadow-lg);  
}
```

```

.messages {
    margin: 20px 0;
}

.message {
    padding: 10px;
    border-radius: 5px;
    margin-bottom: 10px;
    color: #fff;
}

.message.error {
    background-color: #f44336; /* Красный для ошибок */
}

.message.success {
    background-color: #4CAF50; /* Зеленый для успеха */
}

```

## WorkMaterial.css

```

/* Современный адаптивный стиль для страницы практических работ - темная тема с светлыми
блоками */

:root {
    --dark-purple: #1a1225;
    --medium-purple: #7743DB;
    --light-purple: #B39DDB;
    --accent-purple: #9D4EDD;
    --text-light: #ffffff;
    --text-dark: #333333;
    --text-secondary: rgba(255, 255, 255, 0.85);
    --card-bg-light: rgba(255, 255, 255, 0.9);
    --card-bg-accent: rgba(179, 157, 219, 0.2);
    --card-hover: #f8f5ff;
    --gradient-bg: linear-gradient(135deg, #1a1225 0%, #3a1d69 100%);
    --shadow-sm: 0 4px 6px rgba(0, 0, 0, 0.3);
    --shadow-md: 0 6px 12px rgba(0, 0, 0, 0.4);
    --shadow-lg: 0 10px 25px rgba(0, 0, 0, 0.5);
    --transition-slow: all 0.4s cubic-bezier(0.165, 0.84, 0.44, 1);
    --transition-fast: all 0.2s ease;
    --border-radius: 12px;
    --card-spacing: 1.75rem;
}

```

```
/* Принудительное переопределение базовых стилей */
html {
    background-color: var(--dark-purple) !important;
}

body {
    background-color: var(--dark-purple) !important;
    color: var(--text-light) !important;
    font-family: 'Roboto', sans-serif;
    margin: 0;
    padding: 0;
    min-height: 100vh;
    width: 100%;
    overflow-x: hidden;
}

/* Корректировка стилей header */
.main-header {
    background-color: var(--dark-purple) !important;
    width: 100%;
    box-shadow: var(--shadow-md);
    position: relative;
    z-index: 100;
    border-bottom: 1px solid rgba(255, 255, 255, 0.1);
}

.main-header .container {
    max-width: 100% !important;
    width: 100%;
    padding: 0.8rem 2rem;
    display: flex;
    justify-content: space-between;
    align-items: center;
    margin: 0 !important;
    background-color: transparent !important;
    box-shadow: none !important;
}

/* Основной контейнер страницы */
.container {
    max-width: 1200px;
```

```

margin: 0 auto;
padding: 2rem;
background-color: transparent !important;
box-shadow: none !important;
}

/* Заголовок практической работы */
.practical-work-title {
    font-size: 2.5rem;
    font-weight: 700;
    margin-bottom: 1.5rem;
    color: white !important;
    text-shadow: 0 2px 4px rgba(0, 0, 0, 0.3);
    text-align: center;
    background: var(--gradient-bg);
    padding: 2rem;
    border-radius: var(--border-radius);
    box-shadow: var(--shadow-md);
    margin-top: 2rem;
    position: relative;
    overflow: hidden;
}

.practical-work-title::before {
    content: "";
    position: absolute;
    top: 0;
    left: 0;
    right: 0;
    bottom: 0;
    background: url('data:image/svg+xml;utf8,<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 100 100" preserveAspectRatio="none"><path fill="none" stroke="rgba(255,255,255,0.05)" stroke-width="1" d="M0,0 L100,100 M100,0 L0,100"/></svg>');
    background-size: 20px 20px;
    opacity: 0.3;
    z-index: 0;
}

/* Описание практической работы - СВЕТЛЫЙ БЛОК */
.practical-work-description {
    background-color: var(--card-bg-light);
    padding: 1.5rem;
}

```

```
border-radius: var(--border-radius);
margin-bottom: 2rem;
box-shadow: var(--shadow-sm);
border: 1px solid rgba(255, 255, 255, 0.5);
color: var(--text-dark) !important;
line-height: 1.6;
font-size: 1.1rem;
white-space: pre-line;
position: relative;
overflow: hidden;
}

.practical-work-description::before {
content: "";
position: absolute;
top: 0;
left: 0;
width: 5px;
height: 100%;
background: linear-gradient(to bottom, var(--medium-purple), var(--accent-purple));
}

/* Информация о дате и авторе - СВЕТЛЫЙ БЛОК */
.meta-info {
display: flex;
flex-wrap: wrap;
justify-content: space-between;
gap: 1rem;
margin-bottom: 2rem;
padding: 1.2rem 1.2rem 1.2rem 1.5rem;
background-color: var(--card-bg-light);
border-radius: var(--border-radius);
border: 1px solid rgba(255, 255, 255, 0.5);
box-shadow: var(--shadow-sm);
}

.meta-info p {
margin: 0;
font-size: 0.95rem;
color: var(--text-dark);
}
```

```
.meta-info strong {  
    color: var(--accent-purple);  
    margin-right: 0.5rem;  
    font-weight: 600;  
}  
  
/* Материалы практической работы - СВЕТЛЫЙ БЛОК */  
.materials-container {  
    margin-bottom: 2.5rem;  
    background-color: var(--card-bg-light);  
    border-radius: var(--border-radius);  
    padding: 1.5rem;  
    box-shadow: var(--shadow-md);  
    border: 1px solid rgba(255, 255, 255, 0.5);  
    position: relative;  
    overflow: hidden;  
}  
  
.materials-container::before {  
    content: " ";  
    position: absolute;  
    top: 0;  
    left: 0;  
    width: 5px;  
    height: 100%;  
    background: linear-gradient(to bottom, var(--medium-purple), var(--accent-purple));  
}  
  
.materials-title {  
    color: var(--text-dark) !important;  
    font-size: 1.8rem;  
    margin-top: 0;  
    margin-bottom: 1.5rem;  
    padding-bottom: 0.5rem;  
    border-bottom: 2px solid var(--accent-purple);  
    position: relative;  
    font-weight: 600;  
}  
  
.material-link {  
    display: block;  
    padding: 1.2rem;
```

```
background: rgba(255, 255, 255, 0.7);
color: var(--text-dark) !important;
text-decoration: none;
border-radius: var(--border-radius);
transition: var(--transition-fast);
box-shadow: var(--shadow-sm);
border: 1px solid rgba(255, 255, 255, 0.7);
position: relative;
overflow: hidden;
margin-bottom: 1rem;
font-weight: 500;
}

.material-link::before {
content: "";
position: absolute;
top: 0;
left: 0;
width: 4px;
height: 100%;
background: linear-gradient(to bottom, var(--medium-purple), var(--accent-purple));
opacity: 0.7;
}

.material-link:hover {
background: var(--card-hover);
color: var(--accent-purple) !important;
transform: translateY(-3px);
box-shadow: var(--shadow-md);
text-decoration: none;
padding-left: 1.5rem;
}

.material-link i {
margin-right: 0.75rem;
color: var(--accent-purple);
transition: all 0.3s;
}

.material-link:hover i {
transform: scale(1.1);
color: var(--medium-purple);
```

```
}
```

```
/* Стили для формы отправки работы - СВЕТЛЫЙ БЛОК */
```

```
.submit-work-container {  
    margin-top: 2.5rem;  
    background-color: var(--card-bg-light);  
    border-radius: var(--border-radius);  
    padding: 1.5rem;  
    box-shadow: var(--shadow-md);  
    border: 1px solid rgba(255, 255, 255, 0.5);  
    position: relative;  
    overflow: hidden;  
}
```

```
.submit-work-container::before {  
    content: ":";  
    position: absolute;  
    top: 0;  
    left: 0;  
    width: 5px;  
    height: 100%;  
    background: linear-gradient(to bottom, var(--medium-purple), var(--accent-purple));  
}
```

```
.submit-work-title {  
    color: var(--text-dark) !important;  
    font-size: 1.8rem;  
    margin-top: 0;  
    margin-bottom: 1.5rem;  
    padding-bottom: 0.5rem;  
    border-bottom: 2px solid var(--accent-purple);  
    position: relative;  
    font-weight: 600;  
}
```

```
.submit-btn {  
    display: inline-block;  
    padding: 0.75rem 1.5rem;  
    background: linear-gradient(135deg, var(--accent-purple) 0%, var(--medium-purple) 100%);  
    color: white !important;  
    font-weight: 600;  
    text-decoration: none;
```

```
border-radius: var(--border-radius);  
border: none;  
cursor: pointer;  
transition: var(--transition-fast);  
text-align: center;  
font-size: 1rem;  
box-shadow: var(--shadow-sm);  
margin-top: 1rem;  
}
```

```
.submit-btn:hover {  
    transform: translateY(-3px);  
    box-shadow: var(--shadow-md);  
    text-decoration: none;  
}  
  
.submit-btn i {  
    margin-right: 0.5rem;  
}
```

```
/* Стили для полей формы */  
.submit-work-container label {  
    display: block;  
    margin-bottom: 0.5rem;  
    color: var(--text-dark) !important;  
    font-weight: 500;  
}
```

```
.submit-work-container input[type="text"],  
.submit-work-container input[type="file"],  
.submit-work-container textarea,  
.submit-work-container select {  
    width: 100%;  
    padding: 0.75rem;  
    margin-bottom: 1rem;  
    border: 1px solid rgba(0, 0, 0, 0.1);  
    border-radius: 8px;  
    background-color: white;  
    color: var(--text-dark);  
    font-size: 1rem;  
}
```

```
.submit-work-container input[type="text"]:focus,  
.submit-work-container input[type="file"]:focus,  
.submit-work-container textarea:focus,  
.submit-work-container select:focus {  
    border-color: var(--accent-purple);  
    outline: none;  
    box-shadow: 0 0 0 2px rgba(157, 78, 221, 0.2);  
}  
  
/* Стили для списка сданных работ - СВЕТЛЫЙ БЛОК */  
.completed-works-container {  
    margin-top: 2.5rem;  
    background-color: var(--card-bg-light);  
    border-radius: var(--border-radius);  
    padding: 1.5rem;  
    box-shadow: var(--shadow-md);  
    border: 1px solid rgba(255, 255, 255, 0.5);  
    position: relative;  
    overflow: hidden;  
}  
  
.completed-works-container::before {  
    content: " ";  
    position: absolute;  
    top: 0;  
    left: 0;  
    width: 5px;  
    height: 100%;  
    background: linear-gradient(to bottom, var(--medium-purple), var(--accent-purple));  
}  
  
.completed-works-title {  
    color: var(--text-dark) !important;  
    font-size: 1.8rem;  
    margin-top: 0;  
    margin-bottom: 1.5rem;  
    padding-bottom: 0.5rem;  
    border-bottom: 2px solid var(--accent-purple);  
    position: relative;  
    font-weight: 600;  
}
```

```
.completed-work-item {  
    background: rgba(255, 255, 255, 0.7);  
    border-radius: var(--border-radius);  
    padding: 1.2rem;  
    margin-bottom: 1rem;  
    box-shadow: var(--shadow-sm);  
    border: 1px solid rgba(255, 255, 255, 0.7);  
    transition: var(--transition-fast);  
    position: relative;  
    padding-left: 1.5rem;  
}  
  
.completed-work-item::before {  
    content: " ";  
    position: absolute;  
    top: 0;  
    left: 0;  
    width: 4px;  
    height: 100%;  
    background: linear-gradient(to bottom, var(--medium-purple), var(--accent-purple));  
    opacity: 0.7;  
    border-radius: var(--border-radius) 0 0 var(--border-radius);  
}  
  
.completed-work-item:hover {  
    transform: translateY(-3px);  
    box-shadow: var(--shadow-md);  
    background: var(--card-hover);  
}  
  
.completed-work-item p {  
    margin: 0.5rem 0;  
    color: var(--text-dark) !important;  
}  
  
.completed-work-item .work-meta {  
    display: flex;  
    justify-content: space-between;  
    flex-wrap: wrap;  
    gap: 0.5rem;  
    margin-top: 1rem;  
    font-size: 0.9rem;
```

```
color: var(--text-dark);  
}  
  
.completed-work-item .work-link {  
color: var(--accent-purple);  
text-decoration: none;  
transition: var(--transition-fast);  
font-weight: 500;  
}  
  
.completed-work-item .work-link:hover {  
color: var(--medium-purple);  
text-decoration: underline;  
}  
  
.work-status {  
display: inline-block;  
padding: 0.3rem 0.8rem;  
border-radius: 20px;  
font-size: 0.8rem;  
font-weight: 600;  
}  
  
.status-pending {  
background-color: rgba(255, 193, 7, 0.2);  
color: #cc9700;  
border: 1px solid rgba(255, 193, 7, 0.3);  
}  
  
.status-passed {  
background-color: rgba(40, 167, 69, 0.2);  
color: #1e7e34;  
border: 1px solid rgba(40, 167, 69, 0.3);  
}  
  
.status-failed {  
background-color: rgba(220, 53, 69, 0.2);  
color: #b21f2d;  
border: 1px solid rgba(220, 53, 69, 0.3);  
}  
  
/* Стили для статистики преподавателя - СВЕТЛЫЕ БЛОКИ */
```

```
.teacher-stats {  
    display: flex;  
    gap: 1rem;  
    margin-bottom: 1.5rem;  
}  
  
.stat-card {  
    flex: 1;  
    padding: 1.2rem;  
    background-color: var(--card-bg-light);  
    border-radius: var(--border-radius);  
    text-align: center;  
    border: 1px solid rgba(255, 255, 255, 0.5);  
    box-shadow: var(--shadow-sm);  
    transition: var(--transition-fast);  
}  
  
.stat-card:hover {  
    transform: translateY(-3px);  
    box-shadow: var(--shadow-md);  
}  
  
.stat-number {  
    font-size: 2.5rem;  
    font-weight: 700;  
    color: var(--accent-purple);  
    margin-bottom: 0.5rem;  
}  
  
.stat-label {  
    font-size: 0.9rem;  
    color: var(--text-dark);  
    font-weight: 500;  
}  
  
/* Стили для формы оценки работы - СВЕТЛЫЙ БЛОК */  
.grading-form {  
    background-color: var(--card-bg-light);  
    border-radius: var(--border-radius);  
    padding: 1.2rem;  
    margin-top: 1rem;  
    border: 1px solid rgba(255, 255, 255, 0.5);
```

```
    box-shadow: var(--shadow-sm);  
}  
  
.grading-form select,  
.grading-form button {  
    margin-top: 0.5rem;  
}  
  
.grading-form select {  
    padding: 0.5rem;  
    border-radius: 4px;  
    border: 1px solid rgba(0, 0, 0, 0.1);  
    background-color: white;  
    color: var(--text-dark);  
}  
  
.grading-form button {  
    padding: 0.5rem 1rem;  
    background: linear-gradient(135deg, var(--accent-purple) 0%, var(--medium-purple) 100%);  
    color: white;  
    border: none;  
    border-radius: 4px;  
    cursor: pointer;  
    font-weight: 500;  
}  
  
.grading-form button:hover {  
    transform: translateY(-2px);  
    box-shadow: var(--shadow-sm);  
}  
  
/* Стили для студентов и работ в режиме преподавателя - СВЕТЛЫЕ БЛОКИ */  
.student-list {  
    margin-top: 2rem;  
}  
  
.student-item {  
    background-color: var(--card-bg-light);  
    border-radius: var(--border-radius);  
    margin-bottom: 1.5rem;  
    overflow: hidden;  
    box-shadow: var(--shadow-sm);
```

```
border: 1px solid rgba(255, 255, 255, 0.5);
transition: var(--transition-fast);
}

.student-item:hover {
  transform: translateY(-3px);
  box-shadow: var(--shadow-md);
}

.student-header {
  background: linear-gradient(135deg, var(--medium-purple) 0%, var(--accent-purple) 100%);
  padding: 1rem 1.5rem;
  font-weight: 600;
  color: white;
  display: flex;
  justify-content: space-between;
  align-items: center;
}

.student-works {
  padding: 1.2rem;
}

.student-work-item {
  background-color: rgba(255, 255, 255, 0.7);
  border-radius: var(--border-radius);
  padding: 1rem;
  margin-bottom: 1rem;
  box-shadow: var(--shadow-sm);
  border: 1px solid rgba(255, 255, 255, 0.7);
  transition: var(--transition-fast);
  color: var(--text-dark);
  position: relative;
  padding-left: 1.5rem;
}

.student-work-item::before {
  content: "";
  position: absolute;
  top: 0;
  left: 0;
  width: 4px;
}
```

```
height: 100%;
```

```
background: linear-gradient(to bottom, var(--medium-purple), var(--accent-purple));
```

```
opacity: 0.7;
```

```
border-radius: var(--border-radius) 0 0 var(--border-radius);
```

```
}
```

```
.student-work-item:hover {
```

```
transform: translateY(-3px);
```

```
box-shadow: var(--shadow-md);
```

```
background: var(--card-hover);
```

```
}
```

```
.grade-selector {
```

```
display: inline-block;
```

```
padding: 0.5rem;
```

```
background-color: white;
```

```
border-radius: 4px;
```

```
margin-left: 1rem;
```

```
border: 1px solid rgba(0, 0, 0, 0.1);
```

```
color: var(--text-dark);
```

```
font-weight: 500;
```

```
}
```

```
.grade-button {
```

```
background-color: var(--accent-purple);
```

```
color: white;
```

```
border: none;
```

```
border-radius: 4px;
```

```
padding: 0.5rem 1rem;
```

```
margin-left: 0.5rem;
```

```
cursor: pointer;
```

```
transition: var(--transition-fast);
```

```
}
```

```
.grade-button:hover {
```

```
background-color: var(--medium-purple);
```

```
transform: translateY(-2px);
```

```
}
```

```
/* Сообщение об отсутствии работ */
```

```
.no-works-message {
```

```
text-align: center;
```

```
padding: 3rem;  
color: var(--text-dark);  
font-style: italic;  
background-color: var(--card-bg-light);  
border-radius: var(--border-radius);  
margin-top: 2rem;  
border: 1px solid rgba(255, 255, 255, 0.5);  
box-shadow: var(--shadow-sm);  
}
```

```
.no-works-message i {  
font-size: 3rem;  
color: var(--accent-purple);  
margin-bottom: 1rem;  
opacity: 0.5;  
}
```

```
/* Анимации и дополнительные эффекты */  
@keyframes fadeIn {  
from { opacity: 0; }  
to { opacity: 1; }  
}
```

```
.fadeIn {  
animation: fadeIn 0.5s ease-out;  
}
```

```
/* Адаптивный дизайн */  
@media (max-width: 768px) {  
.practical-work-title {  
font-size: 1.8rem;  
padding: 1.5rem;  
}}
```

```
.container {  
padding: 1rem;  
}
```

```
.teacher-stats {  
flex-direction: column;  
}
```

```

.meta-info {
    flex-direction: column;
    padding: 1rem;
}

/*
Обеспечение светлого текста для всех элементов на странице практических работ */
.practical-work-container {
    color: #ffffff !important;
}

.practical-work-container * {
    color: inherit !important;
}

/*
Для специфических элементов, которые могут иметь переопределенные стили */
.task-details, .work-header h1, .work-header h2, .work-instructions p,
.submission-block label, .submission-block p, .completion-info,
.grade-info, .deadline-info, .attachment-info {
    color: #ffffff !important;
}

```

## GroupCRUD.html

```

<!-- GroupCRUD.html -->
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <title>Управление Группами</title>
    {% load static %}
    <link rel="stylesheet" href="{{ static 'css/GroupCRUD.css' }}>
</head>
<body>
    {% include 'header.html' %}

    <div class="container">
        <div class="header-section">
            <a href="{% url 'adminPage' %}" class="back-button">
                <i class="fas fa-arrow-left"></i>
                Назад к панели администратора
            </a>
            <h1>Управление Группами</h1>
        </div>
    </div>

```

```

<div class="create-section">
    <h2>Создать новую группу</h2>
    <form action="{% url 'create_group' %}" method="post" class="crud-form">
        {% csrf_token %}
        <div class="input-group">
            <input type="text" name="nameGroup" placeholder="Введите название новой группы"
required>
            <button type="submit">Добавить Группу</button>
        </div>
    </form>
</div>
</div>

<div class="container">
    <div class="groups-section">
        <h2>Список групп</h2>
        <div class="group-list">
            {% for group in groups %}
                <div class="group-card" id="group-{{ group.idGroup }}">
                    <div class="group-info">
                        <h3>{{ group.nameGroup }}</h3>
                    </div>
                    <div class="crud-actions">
                        <form action="{% url 'edit_group' group.idGroup %}" method="post" class="crud-form">
                            {% csrf_token %}
                            <div class="input-group">
                                <input type="text" name="nameGroup" value="{{ group.nameGroup }}" required>
                                <button type="submit">Сохранить</button>
                            </div>
                        </form>
                        <form action="{% url 'delete_group' group.idGroup %}" method="post" class="delete-form">
                            {% csrf_token %}
                            <button type="submit" class="delete-button">Удалить</button>
                        </form>
                    </div>
                </div>
            {% endfor %}
        </div>
    </div>
</div>

```

```

</body>
</html>

SubjectCRUD.html

<!-- SubjectCRUD.html -->
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Управление Предметами</title>
    {% load static %}
    <link rel="stylesheet" href="{% static 'css/SubjectCRUD.css' %}">
</head>
<body>
    {% include 'header.html' %}

<div class="container">
    <div class="header-section">
        <a href="{% url 'adminPage' %}" class="back-button">
            <i class="fas fa-arrow-left"></i>
            Назад к панели администратора
        </a>
        <h1>Управление Предметами</h1>
    </div>

    <form action="{% url 'subject_crud' %}" method="post" class="crud-form">
        {% csrf_token %}
        <label for="nameSubject">Название предмета:</label>
        <input type="text" id="nameSubject" name="nameSubject" required>

        <label for="idGroup">Группа:</label>
        <select id="idGroup" name="idGroup" required>
            {% for group in groups %}
                <option value="{{ group.idGroup }}>{{ group.nameGroup }}</option>
            {% endfor %}
        </select>

        <label for="userId">Преподаватель:</label>
        <select id="userId" name="userId" required>
            {% for user in users %}
                {% if user.idRole == 2 %}

```

```

<option value="{{ user.idUser }}>
    {{ user.firstName }} {{ user.secondName }} {{ user.middleName }}
</option>
{ % endif %}
{ % endfor %}
</select>

<button type="submit">Добавить предмет</button>
</form>
</div>

<div class="container">

<h2>Список предметов</h2>
<div class="subject-list">

{ % for subject in subjects %}
<div class="subject-card">
    <h3>{{ subject.nameSubject }}</h3>
    <p><strong>Группа:</strong>
        { % for group in groups %}
        { % if group.idGroup == subject.idGroup %}
            {{ group.nameGroup }}
        { % endif %}
        { % endfor %}
    </p>
    <p><strong>Преподаватель:</strong>
        { % for user in users %}
        { % if user.idUser == subject.userId %}
            {{ user.firstName }} {{ user.secondName }} {{ user.middleName }}
        { % endif %}
        { % endfor %}
    </p>
    <div class="crud-actions">
        <a href="{% url 'edit_subject' subject.idSubject %}" class="button">Изменить</a>
        <form action="{% url 'delete_subject' subject.idSubject %}" method="post"
style="display:inline;">
            { % csrf_token %}
            <button type="submit" class="delete-button">Удалить</button>
        </form>
    </div>
</div>

```

```

    {% endfor %}

```

```

</div>

```

```

</div>

```

```

</body>

```

```

</html>

```

## SubjectEdit.html

```

<!-- SubjectEdit.html -->
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Редактирование Предмета</title>
    {% load static %}
    <link rel="stylesheet" href="{% static 'css/SubjectCRUD.css' %}">
</head>
<body>
    {% include 'header.html' %}

    <div class="container">
        <div class="header-section">
            <a href="{% url 'subject_crud' %}" class="back-button">
                <i class="fas fa-arrow-left"></i>
                Назад к списку предметов
            </a>
            <h1>Редактирование Предмета</h1>
        </div>

        <form action="{% url 'edit_subject' subject.idSubject %}" method="post" class="crud-form">
            {% csrf_token %}
            <label for="nameSubject">Название предмета:</label>
            <input type="text" id="nameSubject" name="nameSubject" value="{{ subject.nameSubject }}" required>

            <label for="idGroup">Группа:</label>
            <select id="idGroup" name="idGroup" required>
                {% for group in groups %}
                    <option value="{{ group.idGroup }}" {% if group.idGroup == subject.idGroup %}selected{% endif %}>
                        {{ group.nameGroup }}
                    </option>
                {% endfor %}

```

```

</select>

<label for="userId">Преподаватель:</label>
<select id="userId" name="userId" required>
    { % for user in users %}
        { % if user.idRole == 2 %}
            <option value="{{ user.idUser }}" { % if user.idUser == subject.userId % }selected{ % endif %}
        %}>
            {{ user.firstName }} {{ user.secondName }} {{ user.middleName }}
        </option>
    { % endif %}
    { % endfor %}
</select>

<button type="submit">Сохранить изменения</button>
</form>
</div>
</body>
</html>

```

## UserCRUD.html

```

<!-- AdminPanel/UserCRUD.html -->
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Управление Пользователями</title>
    { % load static %}
    <link rel="stylesheet" href="{ % static 'css/UserCRUD.css' % }">
</head>
<body>
    { % include 'header.html' % }

<div class="container">
    <div class="header-section">
        <a href="{ % url 'adminPage' % }" class="back-button">
            <i class="fas fa-arrow-left"></i>
            Назад к панели администратора
        </a>
        <h1>Управление Пользователями</h1>
    </div>
    <div class="search-options">

```

```

<form method="GET" action="{% url 'user_crud' %}">
    <input type="text" name="search" placeholder="Поиск по ФИО" value="{{ request.GET.search }}">
        <button type="submit">Поиск</button>
    </form>
</div>

<div class="filter-options">
    <form method="GET" action="{% url 'user_crud' %}">
        <label for="group_id">Фильтр по группе:</label>
        <select name="group_id" id="group_id" onchange="this.form.submit()">
            <option value="">Все группы</option>
            {% for group in groups %}
                <option value="{{ group.idGroup }}" {{ group.idGroup|stringformat:"s" == request.GET.group_id ? "selected" : "" }}>{{ group.nameGroup }}</option>
            {% endfor %}
        </select>
    </form>
</div>

<div class="filter-options">
    <form method="GET" action="{% url 'user_crud' %}">
        <label for="sort_by">Сортировка по пользователям:</label>
        <select name="sort_by" id="sort_by" onchange="this.form.submit()">
            <option value="secondName" {{ request.GET.sort_by == 'secondName' ? "selected" : "" }}>Имя</option>
            <option value="firstName" {{ request.GET.sort_by == 'firstName' ? "selected" : "" }}>Фамилия</option>
            <option value="group" {{ request.GET.sort_by == 'group' ? "selected" : "" }}>Группа</option>
        </select>
        <input type="hidden" name="search" value="{{ request.GET.search }}"/>
        <input type="hidden" name="group_id" value="{{ request.GET.group_id }}"/>
    </form>
</div>

```

```

<div class="user-list">
    {% for user in users %}
        <div class="user-card" id="user-{{ user.idUser }}">
            <div class="user-info">
                <h3> {{ user.firstName }} {{ user.secondName }} {{ user.middleName }}</h3>

            </div>

        <div class="crud-actions">

            <form action="{% url 'edit_user' user.idUser %}" method="post" class="crud-form">
                {% csrf_token %}
                <input type="text" name="firstName" value="{{ user.firstName }}" required>
                <input type="text" name="secondName" value="{{ user.secondName }}" required>
                <input type="text" name="middleName" value="{{ user.middleName }}>
                <input type="email" name="email" value="{{ user.email }}" required>
                <select name="idRole">
                    {% for role in roles %}
                        <option value="{{ role.idRole }}" {% if role.idRole == user.idRole %}selected{% endif %}>
                            {{ role.roleName }}
                        </option>
                    {% endfor %}
                </select>
                <select name="idGroup">
                    {% for group in groups %}
                        <option value="{{ group.idGroup }}" {% if group.idGroup == user.idGroup %}selected{% endif %}>
                            {{ group.nameGroup }}
                        </option>
                    {% endfor %}
                </select>
                <button type="submit">Сохранить</button>
            </form>
            <form action="{% url 'delete_user' user.idUser %}" method="post" style="display:inline;">
                {% csrf_token %}
                <button type="submit" class="delete-button">Удалить</button>
            </form>
        </div>
    {% endfor %}
</div>

```

```
</div>
</div>
{ % endfor %
</div>
</div>
</body>
</html>
```

## AddCompleteWork.html

```
<!-- AddCompleteWork.html -->
<!DOCTYPE html>
<html lang="ru">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Добавить выполненную работу</title>
{ % load static %
<link rel="stylesheet" href="{ % static 'css/base.css' % }">
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css">
<link href="https://fonts.googleapis.com/css2?family=Roboto:wght@300,400,500,700&display=swap"
rel="stylesheet">
<style>
/* Принудительное переопределение стилей для тёмной темы */
html, body {
background-color: #1a1225 !important;
color: #ffffff !important;
min-height: 100vh;
width: 100%;
font-family: 'Roboto', sans-serif !important;
}

/* Глобальное переопределение цвета текста */
*, *::before, *::after {
color: #ffffff !important;
}

/* Базовые переменные */
:root {
--dark-purple: #1a1225;
--medium-purple: #7743DB;
--light-purple: #B39DDB;
--accent-purple: #9D4EDD;
--text-light: #ffffff;
```

```

--text-secondary: rgba(255, 255, 255, 0.85);
--card-bg: #272134;
--card-hover: #362b48;
--gradient-bg: linear-gradient(135deg, #1a1225 0%, #3a1d69 100%);
--shadow-md: 0 6px 12px rgba(0, 0, 0, 0.4);
--border-radius: 12px;
--glow: 0 0 15px rgba(157, 78, 221, 0.5);

}

/* Обеспечение темного фона для всех потенциальных контейнеров */
.main-header {
    background-color: var(--dark-purple) !important;
    border-bottom: 1px solid rgba(255, 255, 255, 0.1);
}

.main-header .container {
    background-color: transparent !important;
    box-shadow: none !important;
}

.main-header .logo,
.main-header .user-info,
.main-header .user-name,
.main-header .user-role {
    color: var(--text-light) !important;
}

.container {
    max-width: 800px;
    margin: 30px auto;
    padding: 30px;
    background: linear-gradient(145deg, #272134, #231630) !important;
    box-shadow: 0 15px 25px rgba(0, 0, 0, 0.4);
    border-radius: var(--border-radius);
    border: 1px solid rgba(255, 255, 255, 0.05);
    animation: fadeIn 0.5s ease-out;
}

/* Анимация появления */
@keyframes fadeIn {
    from { opacity: 0; transform: translateY(20px); }
    to { opacity: 1; transform: translateY(0); }
}

```

```
}

/* Улучшенные стили заголовка */

h1 {
    color: var(--text-light) !important;
    text-align: center;
    padding-bottom: 15px;
    margin-bottom: 25px;
    font-size: 28px;
    font-weight: 700;
    text-shadow: 0 2px 4px rgba(0, 0, 0, 0.3);
    position: relative;
    display: inline-block;
    width: 100%;
}

h1:after {
    content: "";
    position: absolute;
    left: 50%;
    bottom: 0;
    transform: translateX(-50%);
    width: 60%;
    height: 3px;
    background: linear-gradient(90deg, transparent, var(--accent-purple), transparent);
}

form {
    margin-top: 20px;
}

label {
    display: block;
    margin-bottom: 10px;
    font-weight: 500;
    color: var(--text-light) !important;
    font-size: 16px;
    letter-spacing: 0.5px;
}

/* Улучшенные поля ввода */

input[type="text"],
```

```
input[type="url"] {  
    width: 100%;  
    padding: 14px;  
    margin-bottom: 25px;  
    background-color: rgba(0, 0, 0, 0.25);  
    border: 1px solid rgba(255, 255, 255, 0.1);  
    border-radius: 10px;  
    font-size: 16px;  
    color: var(--text-light) !important;  
    transition: all 0.3s ease;  
    box-sizing: border-box;  
    box-shadow: inset 0 2px 4px rgba(0, 0, 0, 0.2);  
}  
  
input[type="text"]:focus,  
input[type="url"]:focus {  
    border-color: var(--accent-purple);  
    outline: none;  
    box-shadow: inset 0 2px 4px rgba(0, 0, 0, 0.2), 0 0 10px rgba(157, 78, 221, 0.4);  
    background-color: rgba(0, 0, 0, 0.35);  
}  
  
/* Улучшенные стили плейсхолдеров */  
::placeholder {  
    color: rgba(255, 255, 255, 0.5) !important;  
    opacity: 1;  
}  
  
/* Улучшенная кнопка отправки */  
button[type="submit"] {  
    background: linear-gradient(135deg, var(--accent-purple) 0%, var(--medium-purple) 100%);  
    color: white !important;  
    padding: 15px 25px;  
    border: none;  
    border-radius: 12px;  
    cursor: pointer;  
    font-size: 16px;  
    font-weight: 600;  
    display: block;  
    width: 100%;  
    margin-top: 20px;  
    transition: all 0.4s cubic-bezier(0.175, 0.885, 0.32, 1.275);  
}
```

```
    box-shadow: 0 4px 15px rgba(0, 0, 0, 0.3);
    letter-spacing: 0.5px;
    position: relative;
    overflow: hidden;
    z-index: 1;
}

button[type="submit"]:before {
    content: "";
    position: absolute;
    top: 0;
    left: -100%;
    width: 100%;
    height: 100%;
    background: linear-gradient(90deg,
        transparent,
        rgba(255, 255, 255, 0.2),
        transparent);
    transition: all 0.6s ease;
    z-index: -1;
}

button[type="submit"]:hover {
    transform: translateY(-4px);
    box-shadow: 0 8px 20px rgba(0, 0, 0, 0.3), 0 0 15px rgba(157, 78, 221, 0.5);
}

button[type="submit"]:hover:before {
    left: 100%;
}

button[type="submit"]:active {
    transform: translateY(-2px);
    box-shadow: 0 4px 12px rgba(0, 0, 0, 0.2);
}

button[type="submit"]::after {
    content: '\f15b';
    font-family: 'Font Awesome 5 Free';
    font-weight: 900;
    margin-left: 10px;
    position: absolute;
```

```
    right: 25px;
    top: 50%;
    transform: translateY(-50%);
    opacity: 0;
    transition: all 0.3s ease;
}

button[type="submit"]:hover::after {
    opacity: 1;
    right: 20px;
}

/* Улучшенная кнопка "Назад" */
.back-button {
    display: inline-block;
    background: rgba(255, 255, 255, 0.08);
    color: var(--text-light) !important;
    padding: 12px 24px;
    border-radius: 10px;
    text-decoration: none;
    margin-top: 25px;
    transition: all 0.3s ease;
    font-weight: 500;
    border: 1px solid rgba(255, 255, 255, 0.1);
    position: relative;
    overflow: hidden;
    z-index: 1;
}

.back-button:before {
    content: "";
    position: absolute;
    top: 0;
    left: 0;
    width: 0;
    height: 100%;
    background: linear-gradient(90deg,
        rgba(157, 78, 221, 0.1),
        rgba(157, 78, 221, 0.3));
    transition: all 0.3s ease;
    z-index: -1;
}
```

```
.back-button:hover {  
    background-color: rgba(255, 255, 255, 0.12);  
    transform: translateY(-3px);  
    box-shadow: 0 5px 15px rgba(0, 0, 0, 0.2);  
    text-decoration: none;  
    color: white !important;  
    border-color: rgba(157, 78, 221, 0.3);  
}  
  
.back-button:hover:before {  
    width: 100%;  
}  
  
.back-button::after {  
    content: '\f060';  
    font-family: 'Font Awesome 5 Free';  
    font-weight: 900;  
    margin-right: 10px;  
}  
  
/* Улучшенный счетчик попыток */  
.attempt-counter {  
    display: flex;  
    margin: 15px 0 25px 0;  
    padding: 20px;  
    background: rgba(0, 0, 0, 0.2);  
    border-radius: 12px;  
    align-items: center;  
    border-left: 5px solid var(--accent-purple);  
    position: relative;  
    overflow: hidden;  
}  
  
.attempt-counter:before {  
    content: " ";  
    position: absolute;  
    top: 0;  
    left: 0;  
    width: 100%;  
    height: 100%;  
    background: linear-gradient(135deg,
```

```
    rgba(157, 78, 221, 0.05),  
    transparent);  
    z-index: 0;  
}  
  
.attempt-counter-label {  
    font-weight: 500;  
    margin-right: 15px;  
    color: var(--text-light) !important;  
    position: relative;  
    z-index: 1;  
}  
  
.attempt-circle {  
    width: 38px;  
    height: 38px;  
    border-radius: 50%;  
    margin: 0 8px;  
    display: inline-flex;  
    align-items: center;  
    justify-content: center;  
    font-weight: bold;  
    font-size: 16px;  
    color: white !important;  
    transition: all 0.3s ease;  
    box-shadow: 0 3px 8px rgba(0, 0, 0, 0.3);  
    position: relative;  
    z-index: 1;  
}  
  
.attempt-used {  
    background: linear-gradient(135deg, #f44336, #d32f2f);  
}  
  
.attempt-current {  
    background: linear-gradient(135deg, #ff9800, #f57c00);  
    box-shadow: 0 0 15px rgba(255, 152, 0, 0.5);  
    transform: scale(1.2);  
}  
  
.attempt-available {  
    background: linear-gradient(135deg, #4caf50, #2e7d32);
```

```
}

.attempt-inactive {
    background: linear-gradient(135deg, #9e9e9e, #616161);
    opacity: 0.6;
}

/* Улучшенные баннеры */
.info-banner {
    background: rgba(33, 150, 243, 0.1);
    border-left: 5px solid #2196f3;
    color: var(--text-light) !important;
    padding: 20px;
    border-radius: 10px;
    margin: 20px 0 30px 0;
    font-weight: 500;
    box-shadow: 0 3px 8px rgba(0, 0, 0, 0.2);
    position: relative;
    overflow: hidden;
}

.info-banner:before {
    content: "";
    position: absolute;
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
    background: linear-gradient(135deg,
        rgba(33, 150, 243, 0.05),
        transparent);
    z-index: 0;
}

.info-banner i {
    color: #2196f3 !important;
    margin-right: 10px;
    font-size: 18px;
    position: relative;
    z-index: 1;
}
```

```
.error-banner {  
    background: rgba(244, 67, 54, 0.1);  
    border-left: 5px solid #f44336;  
    color: var(--text-light) !important;  
    padding: 20px;  
    border-radius: 10px;  
    margin: 20px 0 30px 0;  
    font-weight: 500;  
    box-shadow: 0 3px 8px rgba(0, 0, 0, 0.2);  
    position: relative;  
    overflow: hidden;  
}  
}
```

```
.error-banner:before {  
    content: " ";  
    position: absolute;  
    top: 0;  
    left: 0;  
    width: 100%;  
    height: 100%;  
    background: linear-gradient(135deg,  
        rgba(244, 67, 54, 0.05),  
        transparent);  
    z-index: 0;  
}  
}
```

```
.error-banner i {  
    color: #f44336 !important;  
    margin-right: 10px;  
    font-size: 18px;  
    position: relative;  
    z-index: 1;  
}  
}
```

```
/* Улучшенные группы полей */  
.form-group {  
    margin-bottom: 25px;  
    position: relative;  
}  
}
```

```
.form-group-icon {  
    position: relative;  
}
```

```
        }

.form-group-icon i {
    position: absolute;
    left: 14px;
    top: 50%;
    transform: translateY(-50%);
    color: var(--accent-purple) !important;
    font-size: 18px;
    transition: all 0.3s ease;
    z-index: 1;
}

.form-group-icon:hover i {
    color: var(--text-light) !important;
}

.form-group-icon input {
    padding-left: 45px;
}

.form-helper {
    font-size: 13px;
    color: var(--text-secondary) !important;
    margin-top: -20px;
    margin-bottom: 20px;
    display: block;
    opacity: 0.8;
    margin-left: 5px;
    transition: all 0.3s ease;
}

.form-group:hover .form-helper {
    opacity: 1;
    color: var(--text-light) !important;
}

</style>
</head>
<body>
    { % include 'header.html' % }

<div class="container">
```

```
<h1>Добавить выполненную работу</h1>
```

```
{% if attempt_count %}  
    <div class="attempt-counter">  
        <span class="attempt-counter-label">  
            {% if attempt_count == 1 %}  
                Первая попытка:  
            {% elif attempt_count == 2 %}  
                Вторая попытка:  
            {% elif attempt_count == 3 %}  
                Последняя попытка:  
            {% endif %}  
        </span>  
        {% for i in "123" %}  
            {% if forloop.counter < attempt_count %}  
                <div class="attempt-circle attempt-used">{{ forloop.counter }}</div>  
            {% elif forloop.counter == attempt_count %}  
                <div class="attempt-circle attempt-current">{{ forloop.counter }}</div>  
            {% else %}  
                <div class="attempt-circle attempt-available">{{ forloop.counter }}</div>  
            {% endif %}  
        {% endfor %}  
    </div>  
  
    {% if max_attempts_reached %}  
        <div class="error-banner">  
            <i class="fas fa-exclamation-triangle"></i> Вы исчерпали все попытки для сдачи этой  
работы. Обратитесь к преподавателю.  
        </div>  
    {% else %}  
        <div class="info-banner">  
            <i class="fas fa-info-circle"></i> Введите название файла и URL, по которому можно  
найти вашу работу.  
            {% if attempt_count > 1 %}  
                <br><i class="fas fa-exclamation-circle"></i> Это попытка {{ attempt_count }} из 3.  
Осталось попыток: {{ 3|add:"-|add:attempt_count|add:"1" }}.  
            {% endif %}  
        </div>  
  
<form method="POST" action="">  
    {% csrf_token %}
```

```

<div class="form-group">
    <label for="nameFileCompletedWork">Название файла:</label>
    <div class="form-group-icon">
        <i class="fas fa-file-alt"></i>
        <input type="text" id="nameFileCompletedWork" name="nameFileCompletedWork"
required
placeholder="Например: Иванов_задание1.docx">
    </div>
    <span class="form-helper">Введите понятное имя файла с вашей работой</span>
</div>

<div class="form-group">
    <label for="urlFileCompletedWork">Ссылка на файл:</label>
    <div class="form-group-icon">
        <i class="fas fa-link"></i>
        <input type="url" id="urlFileCompletedWork" name="urlFileCompletedWork" required
placeholder="https://example.com/my-work.pdf">
    </div>
    <span class="form-helper">Укажите URL-адрес, где хранится ваша работа (Google
Drive, Dropbox и т.д.)</span>
</div>

<input type="hidden" name="statusId" value="2">
<input type="hidden" name="attempt_count" value="{{ attempt_count|default:'1' }}">

<button type="submit">
    {% if attempt_count > 1 %}
        Отправить исправленную работу (попытка {{ attempt_count }} из 3)
    {% else %}
        Добавить выполненную работу
    {% endif %}
</button>
</form>
{% endif %}
{% else %}
<div class="attempt-counter">
    <span class="attempt-counter-label">Первая попытка:</span>
    <div class="attempt-circle attempt-current">1</div>
    <div class="attempt-circle attempt-available">2</div>
    <div class="attempt-circle attempt-available">3</div>
</div>
{% endif %}

```

```

{ % if subject_id % }
    <a href="{% url 'SubjectPage' subject_id %}" class="back-button">Назад к предмету</a>
{ % else %}
    <a href="{% url 'PracticalWorkMaterial' practical_work.idPracticalWork %}" class="back-
button">Назад к практической работе</a>
{ % endif %}
</div>
</body>
</html>
```

## CompletedWork.html

```

<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Практическая работа: {{ practical_work.namePracticalWork }}</title>
    { % load static %}
    <link rel="stylesheet" href="{% static 'css/WorkMaterial.css' %}">
</head>
<body>
    { % include 'header.html' %}

    <div class="container">
        <h1>Практическая работа: {{ practical_work.namePracticalWork }}</h1>
        <div class="details">
            <p><strong>Описание:</strong> {{ practical_work.descriptionPracticalWork }}</p>
            <p><strong>Дата загрузки:</strong> {{ practical_work.dateUploadPracticalWork }}</p>
            <p><strong>Время загрузки:</strong> {{ practical_work.timeUploadPracticalWork }}</p>
            <div class="file-info">
                <strong>Ссылка на файл:</strong>
                <a href="{{ practical_work.urlFilePracticalWork }}" class="file-link">
                    <div class="file-icon">📄</div>
                    <span>{{ practical_work.nameFilePracticalWork }}</span>
                </a>
            </div>
        </div>
    </div>

    <a href="{% url 'AddCompleteWork' practical_work.idPracticalWork %}" class="add-completed-
work-button">Добавить выполненную работу</a>
    <a href="{% url 'SubjectPage' practical_work.subjectId %}" class="back-button">Назад к
предмету</a>
```

```

<h2>Сданные работы:</h2>
<ul>
    { % for work in completed_works %}
        <li class="completed-work">
            <strong>Имя файла:</strong> <a href="{{ work.urlFileCompletedWork }}" class="file-link">{{ work.nameFileCompletedWork }}</a> <br>
            <strong>Дата:</strong> {{ work.dateUploadCompletedWork }} <br>
            <strong>Время:</strong> {{ work.timeUploadCompletedWork }} <br>
            <strong>Оценка:</strong> {{ work.grade }} <br>
            <strong>Статус:</strong>
            { % if work.statusId == 1 %}
                Работа оценена.
            { % else %}
                Работа не оценена.
            { % endif %}
        </li>
    { % empty %
        <li>Нет сданных работ.</li>
    { % endfor %}
</ul>
</div>
</body>
</html>

```

## CreateTest.html

```

<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    { % load static %}
    { % load custom_filters %}
    <title>Создание теста</title>
    <link rel="stylesheet" href="{% static 'css/CreateTest.css' %}">
    <link rel="stylesheet" href="{% static 'css/TestEditor.css' %}">
    <script src="{% static 'js/scoreCounter.js' %}"></script>
</head>
<body>
    { % include 'header.html' %}

```

```

<div class="container">
    <h1>Создание теста</h1>

```

```

<form id="testForm" method="POST" action="{% url 'create_test' subject_id %}" class="test-form">
    {% csrf_token %}
    <input type="hidden" name="subject_id" value="{{ subject_id }}>
    <input type="hidden" id="gradingCriteriaFinal" name="gradingCriteria">

    <!-- Основная информация о тесте -->
    <div class="test-info-section">
        <div class="form-group">
            <label for="nameTest">Название теста:</label>
            <input type="text" id="nameTest" name="nameTest" required>
        </div>

        <div class="form-group">
            <label for="descriptionTest">Описание теста:</label>
            <textarea id="descriptionTest" name="descriptionTest" required></textarea>
        </div>

        <div class="form-group">
            <label for="timeLimit">Ограничение по времени (в минутах):</label>
            <input type="number" id="timeLimit" name="timeLimit" required min="1">
        </div>
    </div>

    <!-- Критерии оценивания -->
    <div class="grading-section">
        <h3 class="section-title">Критерии оценивания</h3>
        <div id="gradingCriteria" class="grading-criteria-container">
            <div class="grading-row">
                <div class="grade-label">Оценка 5:</div>
                <div class="points-range">
                    <div class="range-input">
                        <input type="number" id="grade5Min" name="grade5Min" placeholder="От" min="0"
required>
                        <span class="range-separator">-</span>
                        <input type="number" id="grade5Max" name="grade5Max" placeholder="До"
min="0" required>
                    </div>
                    <span class="points-label">баллов</span>
                </div>
            </div>
        </div>
    </div>

```

```

<div class="grading-row">
    <div class="grade-label">Оценка 4:</div>
    <div class="points-range">
        <div class="range-input">
            <input type="number" id="grade4Min" name="grade4Min" placeholder="От" min="0"
required>
            <span class="range-separator">-</span>
            <input type="number" id="grade4Max" name="grade4Max" placeholder="До"
min="0" required>
        </div>
        <span class="points-label">баллов</span>
    </div>
</div>

<div class="grading-row">
    <div class="grade-label">Оценка 3:</div>
    <div class="points-range">
        <div class="range-input">
            <input type="number" id="grade3Min" name="grade3Min" placeholder="От" min="0"
required>
            <span class="range-separator">-</span>
            <input type="number" id="grade3Max" name="grade3Max" placeholder="До"
min="0" required>
        </div>
        <span class="points-label">баллов</span>
    </div>
</div>
<div id="gradingError" class="error-message"></div>
</div>

<div id="questions-container">
    <!-- Здесь будут добавляться вопросы -->
</div>

<button type="button" class="add-question-btn" onclick="addQuestion()">Добавить
вопрос</button>
<button type="submit" class="submit-btn">Создать тест</button>
</form>
</div>

<template id="question-template">

```

```

<div class="question-block">
    <div class="question-header">
        <h3>Вопрос #<span class="question-number"></span></h3>
        <button type="button" onclick="removeQuestion(this)" class="remove-btn">Удалить
            вопрос</button>
    </div>
    <div class="form-group">
        <label>Текст вопроса:</label>
        <input type="text" name="questions[][text]" required>
    </div>

    <div class="form-group">
        <label>Тип вопроса:</label>
        <select name="questions[][type]" onchange="handleQuestionTypeChange(this)" required>
            <option value="1">Один правильный ответ</option>
            <option value="2">Несколько правильных ответов</option>
            <option value="3">Сопоставление</option>
            <option value="4">Ввод ответа</option>
            <option value="5">Установление порядка</option>
        </select>
    </div>

    <div class="form-group">
        <label>Баллы за вопрос:</label>
        <input type="number" name="questions[][points]" min="1" value="1" required>
    </div>

    <div class="answers-container">
        <!-- Здесь будут добавляться варианты ответов в зависимости от типа вопроса -->
    </div>
</div>
</template>

<script>
    document.addEventListener('DOMContentLoaded', function() {
        // Инициализация счетчика вопросов
        window.questionCounter = 0;

        // Функция добавления вопроса
        window.addQuestion = function() {
            const template = document.getElementById('question-template');
            const container = document.getElementById('questions-container');

```

```

const clone = template.content.cloneNode(true);

// Обновляем номер вопроса
const questionNumber = ++window.questionCounter;
const questionBlock = clone.querySelector('.question-block');
const numberElement = clone.querySelector('.question-number');
if (numberElement) {
    numberElement.textContent = questionNumber;
}

// Добавляем data-question-id
questionBlock.dataset.questionId = `new_${questionNumber}`;

// Обновляем имена полей
const inputs = clone.querySelectorAll('input, select, textarea');
inputs.forEach(input => {
    const name = input.getAttribute('name');
    if (name) {
        input.setAttribute('name', name.replace('[]', `[new_${questionNumber}]`));
    }
});

container.appendChild(clone);

// Инициализируем тип вопроса
const select = questionBlock.querySelector('select');
select.value = '1';
handleQuestionTypeChange(select);
};

// Функция удаления вопроса
window.removeQuestion = function(button) {
    const questionBlock = button.closest('.question-block');
    if (questionBlock) {
        questionBlock.remove();
        updateQuestionNumbers();
    }
};

// Функция обновления номеров вопросов
window.updateQuestionNumbers = function() {
    const questions = document.querySelectorAll('.question-block');

```

```

questions.forEach((question, index) => {
    const numberElement = question.querySelector('.question-number');
    if (numberElement) {
        numberElement.textContent = index + 1;
    }
});
window.questionCounter = questions.length;
};

// Добавляем первый вопрос при загрузке страницы
addQuestion();
});

function handleQuestionTypeChange(select) {
    const questionBlock = select.closest('.question-block');
    const questionId = questionBlock.dataset.questionId || 'new_question';
    const answersContainer = questionBlock.querySelector('.answers-container');
    const type = select.value;

    let html = "";

    switch(type) {
        case '1': // Один правильный ответ
        case '2': // Несколько правильных ответов
            html = `
                <div class="answers-group">
                    ${[0,1,2,3].map(i => `
                        <div class="form-group">
                            <label>Вариант ${i + 1}:</label>
                            <input type="text"
                                name="questions[${
                                    questionId
                                }][answers][]" required>
                            <input type="${type === '1' ? 'radio' : 'checkbox'}"
                                name="questions[${
                                    questionId
                                }][correct]${type === '2' ? '[]' : ''}"
                                value="${i}">
                        </div>
                    `).join("")}
                </div>
            `;
            break;
        case '3': // Сопоставление
    }
}


```

```

html = `

<div class="matching-group">
  ${[1,2,3,4].map(i => `
    <div class="form-group matching-pair">
      <label>Пара ${i}:</label>
      <input type="text"
        name="questions[${{questionId}}][left][]"
        placeholder="Левая часть"
        required>
      <input type="text"
        name="questions[${{questionId}}][right][]"
        placeholder="Правая часть"
        required>
    </div>
  `).join("")}
</div>
`;

break;

case '4': // Ввод ответа
  html = `

<div class="form-group">
  <label>Правильный ответ:</label>
  <input type="text"
    name="questions[${{questionId}}][correct_answer]"
    required>
</div>
`;
break;

case '5': // Установление порядка
  html = `

<div class="order-group">
  ${[1,2,3,4].map(i => `
    <div class="form-group">
      <label>Элемент ${i}:</label>
      <input type="text"
        name="questions[${{questionId}}][order][]"
        required>
      <span class="order-number">${i}</span>
    </div>
  `).join("")}

```

```

        </div>
    `;
    break;
}

answersContainer.innerHTML = html;
}

document.getElementById('testForm').addEventListener('submit', function(e) {
    e.preventDefault();

    // Получаем все значения диапазонов
    const grade5Min = parseInt(document.getElementById('grade5Min').value) || 0;
    const grade5Max = parseInt(document.getElementById('grade5Max').value) || 0;
    const grade4Min = parseInt(document.getElementById('grade4Min').value) || 0;
    const grade4Max = parseInt(document.getElementById('grade4Max').value) || 0;
    const grade3Min = parseInt(document.getElementById('grade3Min').value) || 0;
    const grade3Max = parseInt(document.getElementById('grade3Max').value) || 0;

    // Проверяем корректность диапазонов
    let errorMessage = "";

    if (grade5Min > grade5Max) {
        errorMessage = 'Для оценки 5: значение "От" не может быть больше значения "До"';
    } else if (grade4Min > grade4Max) {
        errorMessage = 'Для оценки 4: значение "От" не может быть больше значения "До"';
    } else if (grade3Min > grade3Max) {
        errorMessage = 'Для оценки 3: значение "От" не может быть больше значения "До"';
    } else if (grade5Min <= grade4Max) {
        errorMessage = 'Диапазон для оценки 5 должен начинаться после максимального значения для оценки 4';
    } else if (grade4Min <= grade3Max) {
        errorMessage = 'Диапазон для оценки 4 должен начинаться после максимального значения для оценки 3';
    }

    const errorDiv = document.getElementById('gradingError');
    if (errorMessage) {
        errorDiv.textContent = errorMessage;
        errorDiv.style.display = 'block';
        return;
    }
})

```

```

errorDiv.style.display = 'none';

// Формируем текст критериев оценивания
const criteriaText = JSON.stringify([
  { grade: 5, minPoints: grade5Min, maxPoints: grade5Max },
  { grade: 4, minPoints: grade4Min, maxPoints: grade4Max },
  { grade: 3, minPoints: grade3Min, maxPoints: grade3Max }
]);

document.getElementById('gradingCriteriaFinal').value = criteriaText;

// Собираем данные о вопросах
const questions = [];
document.querySelectorAll('.question-block').forEach((block, index) => {
  const questionData = {
    text: block.querySelector('input[name$="[text]"]').value,
    type: parseInt(block.querySelector('select[name$="[type]"]').value),
    points: parseInt(block.querySelector('input[name$="[points]"]').value),
    answers: [],
    correctAnswer: ""
  };

  // Обрабатываем ответы в зависимости от типа вопроса
  switch(questionData.type) {
    case 1: // Один правильный ответ
    case 2: // Несколько правильных ответов
      const answers = block.querySelectorAll('input[type="text"]')[name$="[answers][]"]');
      const selected = block.querySelectorAll(`input[type="${questionData.type === 1 ? 'radio' : 'checkbox'}"])[name$="[correct]${questionData.type === 2 ? '[]' : '}"]`);

      answers.forEach((answer, i) => {
        questionData.answers.push(answer.value);
        if (selected[i].checked) {
          if (questionData.type === 1) {
            questionData.correctAnswer = answer.value;
          } else {
            questionData.correctAnswer += (questionData.correctAnswer ? ' ; ' : '') +
              answer.value;
          }
        }
      });
    });
  break;
});

```

```

        case 3: // Сопоставление
            const leftParts = block.querySelectorAll('input[name$="[left][]"']);
            const rightParts = block.querySelectorAll('input[name$="[right][]"']");
            leftParts.forEach((left, i) => {
                const pair = `${left.value} - ${rightParts[i].value}`;
                questionData.answers.push(pair);
            });
            questionData.correctAnswer = questionData.answers.join(' ');
            break;

        case 4: // Ввод ответа
            questionData.correctAnswer =
block.querySelector('input[name$="[correct_answer]"]').value;
            break;

        case 5: // Установление порядка
            const orderItems = block.querySelectorAll('input[name$="[order][]""]');
            orderItems.forEach(item => {
                questionData.answers.push(item.value);
            });
            questionData.correctAnswer = questionData.answers.join(' ');
            break;
    }

    questions.push(questionData);
});

// Добавляем вопросы в форму
const questionsInput = document.createElement('input');
questionsInput.type = 'hidden';
questionsInput.name = 'questions';
questionsInput.value = JSON.stringify(questions);
this.appendChild(questionsInput);

// Отправляем форму
this.submit();
};

</script>
</body>
</html>

```

## CRUDPracticalWork.html

```

<!-- CRUDPracticalWork.html -->
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Добавление практической работы</title>
    {% load static %}
    <link rel="stylesheet" href="{{ static 'css/LectionCRUD.css' }}>
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css">
    <link href="https://fonts.googleapis.com/css2?family=Roboto:wght@300;400;500;700&display=swap" rel="stylesheet">
</head>
<body>
    <div class="container">
        <h1><i class="fas fa-laptop-code"></i> Добавить новую практическую работу</h1>
        <form method="POST">
            {% csrf_token %}
            <div class="form-group">
                <label for="namePracticalWork">Название практической работы:</label>
                <input type="text" id="namePracticalWork" name="namePracticalWork" required
                       placeholder="Введите название практической работы">
            </div>

            <div class="form-group">
                <label for="descriptionPracticalWork">Описание практической работы:</label>
                <textarea id="descriptionPracticalWork" name="descriptionPracticalWork" required
                          placeholder="Введите описание практической работы"></textarea>
            </div>

            <div class="form-group">
                <label for="nameFilePracticalWork">Название файла:</label>
                <input type="text" id="nameFilePracticalWork" name="nameFilePracticalWork" required
                       placeholder="Введите название файла">
            </div>

            <div class="form-group">
                <label for="urlFilePracticalWork">Ссылка на файл:</label>
                <input type="url" id="urlFilePracticalWork" name="urlFilePracticalWork" required
                       placeholder="Вставьте ссылку на файл">
            </div>
    </div>

```

```

<button type="submit" class="submit-button">
    <i class="fas fa-plus"></i> Добавить практическую работу
</button>
</form>
{%
    if error %
        <p class="error-message">
            <i class="fas fa-exclamation-circle"></i> {{ error }}
        </p>
    {% endif %}
</div>
</body>
</html>

```

## EditTest.html

```

<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    {%
        load static %
    }
    {%
        load custom_filters %
    }
    <title>Редактирование теста</title>
    <link rel="stylesheet" href="{% static 'css/CreateTest.css' %}">
    <link rel="stylesheet" href="{% static 'css/TestEditor.css' %}">
    <script src="{% static 'js/scoreCounter.js' %}"></script>
</head>
<body>
    {%
        include 'header.html' %
    }

    <div class="container">
        <h1>Редактирование теста</h1>
        <form method="POST" class="test-form">
            {%
                csrf_token %
            }
            <input type="hidden" name="subjectId" value="{{ test.subjectId }}>

            <div class="form-group">
                <label for="nameTest">Название теста:</label>
                <input type="text" id="nameTest" name="nameTest" value="{{ test.nameTest }}" required>
            </div>

            <div class="form-group">
                <label for="descriptionTest">Описание теста:</label>

```

```

<textarea      id="descriptionTest"      name="descriptionTest"      rows="4"      required>{{ test.descriptionTest }}</textarea>
</div>

<div class="form-group">
    <label for="timeLimit">Ограничение по времени (в минутах):</label>
    <input type="number" id="timeLimit" name="timeLimit" min="1" value="{{ test.timeLimit }}" required>
</div>

<div class="grading-criteria">
    <h2>Критерии оценивания</h2>
    <div id="gradingError"></div>
    <div class="grading-row">
        <div class="grade-label">Оценка 5:</div>
        <div class="points-range">
            <input type="number" id="grade5Min" name="grade5Min" placeholder="От" min="0" required>
            <span>-</span>
            <input type="number" id="grade5Max" name="grade5Max" placeholder="До" min="0" required>
            <span>баллов</span>
        </div>
    </div>
    <div class="grading-row">
        <div class="grade-label">Оценка 4:</div>
        <div class="points-range">
            <input type="number" id="grade4Min" name="grade4Min" placeholder="От" min="0" required>
            <span>-</span>
            <input type="number" id="grade4Max" name="grade4Max" placeholder="До" min="0" required>
            <span>баллов</span>
        </div>
    </div>
    <div class="grading-row">
        <div class="grade-label">Оценка 3:</div>
        <div class="points-range">
            <input type="number" id="grade3Min" name="grade3Min" placeholder="От" min="0" required>
            <span>-</span>
        </div>
    </div>
</div>

```

```

<input type="number" id="grade3Max" name="grade3Max" placeholder="До" min="0"
required>
    <span>баллов</span>
</div>
</div>
</div>

<div id="questions-container">
    { % for question in questions %}
        <div class="question-block" data-question-id="{{ question.idQuestion }}">
            <h3>Вопрос #{{ forloop.counter }}</h3>
            <div class="form-group">
                <label>Текст вопроса:</label>
                <input type="text"
                    name="questions[{{ question.idQuestion }}][text]"
                    value="{{ question.questionText }}"
                    required>
            </div>

            <div class="form-group">
                <label>Тип вопроса:</label>
                <select name="questions[{{ question.idQuestion }}][type]">
                    <option value="1" {{ if question.questionType == 1 }}selected{{ endif }}>Один
                    правильный ответ</option>
                    <option value="2" {{ if question.questionType == 2 }}selected{{ endif }}>Несколько
                    правильных ответов</option>
                    <option value="3" {{ if question.questionType == 3 }}selected{{ endif }}>Сопоставление</option>
                    <option value="4" {{ if question.questionType == 4 }}selected{{ endif }}>Ввод
                    ответа</option>
                    <option value="5" {{ if question.questionType == 5 }}selected{{ endif }}>Установление порядка</option>
                </select>
            </div>

            <div class="form-group">
                <label>Баллы за вопрос:</label>
                <input type="number"
                    name="questions[{{ question.idQuestion }}][points]"
                    min="1"
                    value="{{ question.points }}">
            </div>
        </div>
    { % endfor %}
</div>

```

```

    required>
</div>

<div class="answers-container">
{ % if question.questionType == 1 or question.questionType == 2 %}
{ % for answer in question.answerVariants|split:';' %}
<div class="form-group">
<label>Вариант {{ forloop.counter }}:</label>
<input type="text"
      name="questions[{{ question.idQuestion }}][answers][]"
      value="{{ answer }}"
      required>
<input type="{% if question.questionType == 1 %}radio{% else %}checkbox{% endif %}"
      name="questions[{{ question.idQuestion }}][correct]{% if question.questionType == 2 %}[]{% endif %}"
      value="{{ forloop.counter0 }}"
      { % if answer in question.correctAnswer|split:';' %}checked{% endif %}>
</div>
{ % endfor %

{ % elif question.questionType == 3 %
{ % for pair in question.answerVariants|split:';' %}
<div class="form-group matching-pair">
<label>Пара {{ forloop.counter }}:</label>
<input type="text" name="questions[{{ question.idQuestion }}][left][]"
      value="{{ pair|split:' - '|first }}" placeholder="Левая часть" required>
<input type="text" name="questions[{{ question.idQuestion }}][right][]"
      value="{{ pair|split:' - '|last }}" placeholder="Правая часть" required>
</div>
{ % endfor %

{ % elif question.questionType == 4 %
<div class="form-group">
<label>Правильный ответ:</label>
<input type="text" name="questions[{{ question.idQuestion }}][correct_answer]"
      value="{{ question.correctAnswer }}" required>
</div>
{ % elif question.questionType == 5 %
{ % for item in question.answerVariants|split:';' %}
<div class="form-group">
<label>Элемент {{ forloop.counter }}:</label>
<input type="text" name="questions[{{ question.idQuestion }}][order][]"
      value="{{ item }}" required>

```

```

        <span class="order-number">{ forloop.counter }</span>
    </div>
    { % endfor %

    { % endif %

    </div>

        <button type="button" onclick="removeQuestion(this)" class="remove-btn">Удалить
вопрос</button>
    </div>
    { % endfor %
    </div>

        <button type="button" class="add-question-btn" onclick="addQuestion()">
            Добавить вопрос
        </button>
        <button type="submit" class="submit-btn">
            Сохранить изменения
        </button>
    </form>
</div>

<template id="question-template">
    <div class="question-block">
        <div class="question-header">
            <h3>Вопрос <span class="question-number"></span></h3>
            <button type="button" onclick="removeQuestion(this)" class="remove-btn">Удалить
вопрос</button>
        </div>
        <div class="form-group">
            <label>Текст вопроса:</label>
            <input type="text" name="questions[][text]" required>
        </div>

        <div class="form-group">
            <label>Тип вопроса:</label>
            <select name="questions[][type]" onchange="handleQuestionTypeChange(this)" required>
                <option value="1">Один правильный ответ</option>
                <option value="2">Несколько правильных ответов</option>
                <option value="3">Сопоставление</option>
                <option value="4">Ввод ответа</option>
                <option value="5">Установление порядка</option>
            </select>
        </div>
    </div>
</template>

```

```

</div>

<div class="form-group">
    <label>Баллы за вопрос:</label>
    <input type="number" name="questions[]["points]" min="1" value="1" required>
</div>

<div class="answers-container">
    <!-- Здесь будут добавляться варианты ответов в зависимости от типа вопроса -->
</div>

<button type="button" onclick="removeQuestion(this)" class="remove-btn">Удалить
вопрос</button>
</div>
</template>

<script>
document.addEventListener('DOMContentLoaded', function() {
    // Инициализация счетчика вопросов
    window.questionCounter = { questions: length };

    // Функция добавления вопроса
    window.addQuestion = function() {
        const template = document.getElementById('question-template');
        const container = document.getElementById('questions-container');
        const clone = template.content.cloneNode(true);

        // Обновляем номер вопроса
        const questionNumber = ++window.questionCounter;
        const questionTitle = clone.querySelector('h3');
        questionTitle.textContent = `Вопрос #${questionNumber}`;

        // Создаем новый div для вопроса и добавляем ему data-question-id
        const questionBlock = clone.querySelector('.question-block');
        questionBlock.dataset.questionId = `new_${questionNumber}`;

        // Добавляем уникальные идентификаторы для полей
        const inputs = clone.querySelectorAll('input, select, textarea');
        inputs.forEach(input => {
            const name = input.getAttribute('name');
            if (name) {
                input.setAttribute('name', name.replace('[', `[new_${questionNumber}]`));
            }
        });
    };
});
</script>

```

```

        }

    });

    container.appendChild(clone);

    // Инициализируем поля для ответов по умолчанию
    const lastQuestion = container.lastElementChild;
    const select = lastQuestion.querySelector('select');

    // Устанавливаем тип вопроса "Один правильный ответ" по умолчанию
    select.value = '1';
    handleQuestionTypeChange(select);

    const testIdInput = document.createElement('input');
    testIdInput.type = 'hidden';
    testIdInput.name = `questions[new_${questionNumber}][idTest]`;
    testIdInput.value = '{ test.idTest }';
    questionBlock.appendChild(testIdInput);
}

// Функция удаления вопроса
window.removeQuestion = function(button) {
    const questionBlock = button.closest('.question-block');
    if (questionBlock) {
        // Проверяем, является ли это существующим вопросом
        const questionId = questionBlock.dataset.questionId;
        if (!questionId.startsWith('new_')) {
            // Если это существующий вопрос, добавляем скрытое поле для отметки удаления
            const deleteInput = document.createElement('input');
            deleteInput.type = 'hidden';
            deleteInput.name = `deleted_questions[]`;
            deleteInput.value = questionId;
            document.querySelector('.test-form').appendChild(deleteInput);
        }
    }
}

// Удаляем блок вопроса из DOM
questionBlock.remove();
updateQuestionNumbers();

// Убираем автоматическую отправку формы
// document.querySelector('.test-form').submit();
}

```

```

};

// Функция обновления номеров вопросов
window.updateQuestionNumbers = function() {
    const questions = document.querySelectorAll('.question-block h3');
    questions.forEach((header, index) => {
        header.textContent = `Вопрос ${index + 1}`;
    });
    window.questionCounter = questions.length;
};

// Функция обработки изменения типа вопроса
window.handleQuestionTypeChange = function(select) {
    const questionBlock = select.closest('.question-block');
    const questionId = questionBlock.dataset.questionId || 'new_question';
    const answersContainer = questionBlock.querySelector('.answers-container');
    const type = select.value;

    console.log('Changing question type:', {
        questionId,
        type,
        container: answersContainer
    });

    let html = '';

    switch(type) {
        case '1': // Один правильный ответ
        case '2': // Несколько правильных ответов
            html = `
<div class="answers-group">
${[0,1,2,3].map(i => `
<div class="form-group">
<label>Вариант ${i + 1}</label>
<input type="text"
      name="questions[$questionId][answers][]" required>
<input type="${type === '1' ? 'radio' : 'checkbox}'"
      name="questions[$questionId][correct]${type === '2' ? '[]' : ''}"
      value="${i}">
</div>
`)).join("")
    }
}

```

```

        </div>
    `;
break;

case '3': // Сопоставление
html = `
<div class="matching-group">
${[1,2,3,4].map(i => `
<div class="form-group matching-pair">
<label>Пара ${i}:</label>
<input type="text"
      name="questions[$questionId][left][]"
      placeholder="Левая часть"
      required>
<input type="text"
      name="questions[$questionId][right][]"
      placeholder="Правая часть"
      required>
</div>
`).join("")}
</div>
`;
break;

case '4': // Ввод ответа
html = `
<div class="form-group">
<label>Правильный ответ:</label>
<input type="text"
      name="questions[$questionId][correct_answer]"
      required>
</div>
`;
break;

case '5': // Установление порядка
html = `
<div class="order-group">
${[1,2,3,4].map(i => `
<div class="form-group">
<label>Элемент ${i}:</label>
<input type="text"

```

```

        name="questions[ ${questionId} ][order][]"
        required>
        <span class="order-number">${i}</span>
        </div>
        `).join(""))
        </div>
        `;
        break;
    }

    console.log('Generated HTML:', html);

    // Устанавливаем HTML содержимое
    answersContainer.innerHTML = html;

    // Добавляем скрытое поле для ID теста
    const testIdInput = document.createElement('input');
    testIdInput.type = 'hidden';
    testIdInput.name = `questions[ ${questionId} ][idTest]`;
    testIdInput.value = '{ test.idTest }';
    answersContainer.appendChild(testIdInput);
};

// Инициализация критериев оценивания
{ % for criterion in grading_criteria % }
    if ({ criterion.grade }) === 5 {
        document.getElementById('grade5Min').value = { criterion.minPoints };
        document.getElementById('grade5Max').value = { criterion.maxPoints };
    } else if ({ criterion.grade }) === 4 {
        document.getElementById('grade4Min').value = { criterion.minPoints };
        document.getElementById('grade4Max').value = { criterion.maxPoints };
    } else if ({ criterion.grade }) === 3 {
        document.getElementById('grade3Min').value = { criterion.minPoints };
        document.getElementById('grade3Max').value = { criterion.maxPoints };
    }
{ % endfor % }

// Добавляем обработчик отправки формы
const form = document.querySelector('.test-form');
if (form) {
    form.addEventListener('submit', function(e) {
        e.preventDefault();

```

```

// Собираем данные о правильных ответах перед отправкой
document.querySelectorAll('.question-block').forEach(questionBlock => {
    const questionId = questionBlock.dataset.questionId;
    const questionType = questionBlock.querySelector('select[name*="type"]').value;

    // Получаем контейнер для ответов
    const answersContainer = questionBlock.querySelector('.answers-container');

    switch(questionType) {
        case '1': // Один правильный ответ
            const selectedRadio = answersContainer.querySelector('input[type="radio"]:checked');
            if (selectedRadio) {
                const answerIndex = selectedRadio.value;
                const answerInput = answersContainer.querySelector('input[type="text"]')[answerIndex];
                if (answerInput) {
                    // Создаем скрытое поле для правильного ответа
                    const correctInput = document.createElement('input');
                    correctInput.type = 'hidden';
                    correctInput.name = `questions[ ${questionId} ][correctAnswer]`;
                    correctInput.value = answerInput.value;
                    answersContainer.appendChild(correctInput);
                }
            }
            break;

        case '2': // Несколько правильных ответов
            const selectedCheckboxes = answersContainer.querySelectorAll('input[type="checkbox"]:checked');
            const correctAnswers = Array.from(selectedCheckboxes).map(checkbox => {
                const answerInput = answersContainer.querySelector('input[type="text"]')[checkbox.value];
                return answerInput ? answerInput.value : '';
            }).filter(Boolean);

            // Создаем скрытое поле для правильных ответов
            const correctInput = document.createElement('input');
            correctInput.type = 'hidden';
            correctInput.name = `questions[ ${questionId} ][correctAnswer]`;
            correctInput.value = correctAnswers.join('; ');
            answersContainer.appendChild(correctInput);
    }
}

```

```

        break;
    }

    // Добавляем ID теста к каждому вопросу
    const testIdInput = document.createElement('input');
    testIdInput.type = 'hidden';
    testIdInput.name = `questions[$\{questionId\}][idTest]`;
    testIdInput.value = '{\` test.idTest }';
    answersContainer.appendChild(testIdInput);
});

// Отправляем форму
form.submit();
};

}
);

</script>
</body>
</html>

```

## LectionCRUD.html

```

<!-- LectionCRUD.html -->
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Добавление лекции</title>
    {% load static %}
    <link rel="stylesheet" href="{% static 'css/LectionCRUD.css' %}">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css">
    <link href="https://fonts.googleapis.com/css2?family=Roboto:wght@300;400;500;700&display=swap" rel="stylesheet">
</head>
<body>
    <div class="container">
        <h1><i class="fas fa-book-open"></i> Добавить новую лекцию</h1>
        <form method="POST">
            {% csrf_token %}
            <div class="form-group">
                <label for="nameLectionMaterial">Название лекции:</label>
                <input type="text" id="nameLectionMaterial" name="nameLectionMaterial" required
placeholder="Введите название лекции">

```

```

</div>

<div class="form-group">
    <label for="descriptionLectionMaterial">Описание лекции:</label>
    <textarea id="descriptionLectionMaterial" name="descriptionLectionMaterial" required
              placeholder="Введите описание лекции"></textarea>
</div>

<div class="form-group">
    <label for="nameFileLection">Название файла лекции:</label>
    <input type="text" id="nameFileLection" name="nameFileLection" required
           placeholder="Введите название файла">
</div>

<div class="form-group">
    <label for="urlFileLection">Ссылка на файл лекции:</label>
    <input type="url" id="urlFileLection" name="urlFileLection" required
           placeholder="Вставьте ссылку на файл">
</div>

<button type="submit" class="submit-button">
    <i class="fas fa-plus"></i> Добавить лекцию
</button>
</form>
{ % if error %
<p class="error-message">
    <i class="fas fa-exclamation-circle"></i> {{ error }}
</p>
{ % endif %
</div>
</body>
</html>

```

## LectionEdit.html

```

<!-- LectionEdit.html -->
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Редактировать лекцию</title>
    { % load static %}
    <link rel="stylesheet" href="{% static 'css/LectionMaterial.css' %}">

```

```

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css">
<link href="https://fonts.googleapis.com/css2?family=Roboto:wght@300;400;500;700&display=swap"
rel="stylesheet">
</head>
<body>
{ % include 'header.html' % }

<div class="container">
<h1>Редактирование лекции</h1>

<div class="button-group">
<a href="{ % url 'SubjectPage' lection.subjectId % }" class="back-button">
<i class="fas fa-arrow-left"></i> Назад к предмету
</a>
</div>

{ % if error % }
<div class="error-message">
<i class="fas fa-exclamation-circle"></i> {{ error }}
</div>
{ % endif % }

<div class="edit-form-container">
<form method="POST" class="edit-form">
{ % csrf_token % }
<div class="form-group">
<label for="nameLectionMaterial">
<i class="fas fa-heading"></i> Название лекции:
</label>
<input type="text" id="nameLectionMaterial" name="nameLectionMaterial"
value="{{ lection.nameLectionMaterial }}" required>
</div>

<div class="form-group">
<label for="descriptionLectionMaterial">
<i class="fas fa-align-left"></i> Описание лекции:
</label>
<textarea id="descriptionLectionMaterial" name="descriptionLectionMaterial"
required>{{ lection.descriptionLectionMaterial }}</textarea>
</div>

<div class="form-group">

```

```

<label for="nameFileLection">
    <i class="fas fa-file-alt"></i> Имя файла:
</label>
<input type="text" id="nameFileLection" name="nameFileLection"
       value="{{ lection.nameFileLection }}" required>
</div>

<div class="form-group">
    <label for="urlFileLection">
        <i class="fas fa-link"></i> URL файла:
    </label>
    <input type="url" id="urlFileLection" name="urlFileLection"
           value="{{ lection.urlFileLection }}" required>
</div>

<div class="button-group">
    <button type="submit" class="save-button">
        <i class="fas fa-save"></i> Сохранить изменения
    </button>
</div>
</form>
</div>
</body>
</html>

```

## LectionMaterial.html

```

<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    {% load static %}
    <title>{{ lection.nameLectionMaterial }}</title>
    <link rel="stylesheet" href="{{ static 'css/LectionMaterial.css' }">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css">
    <link href="https://fonts.googleapis.com/css2?family=Roboto:wght@300;400;500;700&display=swap"
          rel="stylesheet">
</head>
<body>
    {% include 'header.html' %}

    <div class="container">

```

```

<h1>{{ lection.nameLectionMaterial }}</h1>

<div class="button-group">
    <a href="{% url 'SubjectPage' lection.subjectId %}" class="back-button">
        <i class="fas fa-arrow-left"></i> Назад к предмету
    </a>

    {% if role_id == 2 or role_id == 1 or lection.userId == user_id %}
        <a href="{% url 'edit_lection' lection.idLectionMaterial %}" class="edit-button">
            <i class="fas fa-edit"></i> Редактировать лекцию
        </a>
    {% endif %}

    <form method="POST" action="{% url 'delete_lection' lection.idLectionMaterial %}"
          class="delete-form" onsubmit="return confirm('Вы уверены, что хотите удалить эту
лекцию?');">
        {% csrf_token %}
        <input type="hidden" name="subjectId" value="{{ lection.subjectId }}>
        <button type="submit" class="delete-button">
            <i class="fas fa-trash"></i> Удалить лекцию
        </button>
    </form>
</div>

<div class="details">
    <p><strong>Описание:</strong> {{ lection.descriptionLectionMaterial }}</p>
    <p><strong>Дата загрузки:</strong> {{ lection.dateUploadLectionMaterial }}</p>
    <p><strong>Время загрузки:</strong> {{ lection.timeUploadLectionMaterial }}</p>
    <div class="file-info">
        <strong>Файл лекции:</strong>
        <a href="{{ lection.urlFileLection }}" class="file-link" target="_blank">
            <i class="fas fa-file-alt file-icon"></i>
            <span>{{ lection.nameFileLection }}</span>
        </a>
    </div>
    <p><strong>Преподаватель:</strong> {{ teacher.firstName }} {{ teacher.secondName }} {{
teacher.middleName }}</p>
    </div>
</div>
</body>
</html>

```

## PracticalWorkEdit.html

```

<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Редактировать практическую работу</title>
    {% load static %}
    <link rel="stylesheet" href="{% static 'css/SubjectPage.css' %}">
</head>
<body>
    <h1>Редактировать практическую работу</h1>

    {% if error %}
        <p style="color:red;">{{ error }}</p>
    {% endif %}

    <form method="POST">
        {% csrf_token %}
        <div>
            <label for="namePracticalWork">Название практической работы:</label>
            <input type="text" id="namePracticalWork" name="namePracticalWork" value="{{ practical_work.namePracticalWork }}" required>
        </div>
        <div>
            <label for="descriptionPracticalWork">Описание:</label>
            <textarea id="descriptionPracticalWork" name="descriptionPracticalWork" required>{{ practical_work.descriptionPracticalWork }}</textarea>
        </div>
        <div>
            <label for="nameFilePracticalWork">Имя файла:</label>
            <input type="text" id="nameFilePracticalWork" name="nameFilePracticalWork" value="{{ practical_work.nameFilePracticalWork }}">
        </div>
        <div>
            <label for="urlFilePracticalWork">URL файла:</label>
            <input type="url" id="urlFilePracticalWork" name="urlFilePracticalWork" value="{{ practical_work.urlFilePracticalWork }}">
        </div>
        <button type="submit">Сохранить изменения</button>
    </form>
    <a href="{% url 'SubjectPage' practical_work.subjectId %}">Назад к предмету</a>
</body>

```

```
</html>
```

## PracticalWorkMaterial.html

```
<!-- PracticalWorkMaterial.html -->
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    {% load static %}
    <title>Практическая работа: {{ practical_work.namePracticalWork }}</title>
    <link rel="stylesheet" href="{{ static 'css/base.css' }}>
    <link rel="stylesheet" href="{{ static 'css/PracticalWork.css' }}>
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css">
    <link
        href="https://fonts.googleapis.com/css2?family=Roboto:wght@300;400;500;700;800&display=swap"
        rel="stylesheet">
        <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
</head>
<body>
    {% include 'header.html' %}

    <div class="container">
        <h1>Практическая работа: {{ practical_work.namePracticalWork }}</h1>

        <!-- Добавляем кнопку удаления рядом с кнопкой "Назад к предмету" -->
        <div class="button-group">
            <a href="{% url 'SubjectPage' practical_work.subjectId %}" class="back-button">
                <i class="fas fa-arrow-left"></i> Назад к предмету
            </a>
        </div>

        <div class="details">
            <p><strong>Описание:</strong> {{ practical_work.descriptionPracticalWork }}</p>
            <p><strong>Дата загрузки:</strong> {{ practical_work.dateUploadPracticalWork }}</p>
            <p><strong>Время загрузки:</strong> {{ practical_work.timeUploadPracticalWork }}</p>
            <div class="file-info">
                <strong>Ссылка на файл:</strong>
                <a href="{{ practical_work.urlFilePracticalWork }}" class="file-link" target="_blank">
                    <span class="file-icon">  </span>
                    {{ practical_work.nameFilePracticalWork }}
                </a>
            </div>
        </div>
    </div>
</body>
```

```

        </a>
    </div>
</div>

<div>
{%
    if user_role_id == 2 or user_role_id == 1 %
}
<form method="POST" action="{% url 'delete_practical_work' practical_work.idPracticalWork
%}">
    class="delete-form" onsubmit="return confirm('Вы уверены, что хотите удалить эту
практическую работу?');">
        {% csrf_token %}
        <input type="hidden" name="subjectId" value="{{ practical_work.subjectId }}">
        <button type="submit" class="delete-button">
            <i class="fas fa-trash"></i> Удалить работу
        </button>
    </form>
{%
    endif %
}
</div>

<!-- Секция для студента -->
{%
    if user_role_id == 3 %
}
{%
    if completed_works %
}
{%
    for work in completed_works %
}
<div class="completed-work">
    <h3>Ваша работа</h3>
    <p><strong>Имя файла:</strong>
        <a href="{{ work.urlFileCompletedWork }}" target="_blank">
            <span class="file-icon"> </span>
            {{ work.nameFileCompletedWork }}
        </a>
    </p>
    <p><strong>Дата сдачи:</strong> {{ work.dateUploadCompletedWork }} в {{ work.timeUploadCompletedWork }}</p>
    <p><strong>Статус работы:</strong>
        {% if work.grade == "На проверке" %}
            <span class="grade-pending">{{ work.grade }}</span>
        {% elif '2' in work.grade %}
            <span class="grade-2">{{ work.grade }}</span>
        {% elif work.grade == '3' or work.grade == '4' or work.grade == '5' %}
            <span class="grade-3-5">{{ work.grade }}</span>
        {% else %}
            <span class="grade-1-1-5">{{ work.grade }}</span>
        {% endif %}
    </p>
</div>
{%
    endfor %
}
{%
    endif %
}
</div>

```

```

{ % else % }
    {{ work.grade }}
{ % endif % }
</p>
<div class="work-info">
    <p>Текущее количество попыток: {{ work.attempt_count }} из 3</p>

    <div class="attempts-visualization">
        <p>Использовано попыток:</p>
        <div class="attempt-circles">
            { % for i in "123" %}
                { % if forloop.counter <= work.attempt_count %}
                    <div class="attempt-circle attempt-used">{{ forloop.counter }}</div>
                { % else %}
                    <div class="attempt-circle attempt-available">{{ forloop.counter }}</div>
                { % endif %}
            { % endfor %}
        </div>
    </div>
</div>

<div class="student-grade">
    <strong>Оценка:</strong>
    { % if work.grade == "На проверке" %}
        <span class="grade-pending">{{ work.grade }}</span>
        <div class="info-banner">
             Ваша работа находится на проверке. Пожалуйста, ожидайте оценки преподавателя.
        </div>
    { % if forloop.counter >= 3 %}
        <div class="error-banner">
             Вы использовали все доступные попытки (3 из 3) для сдачи этой работы.
        </div>
    { % endif %}
    { % elif '2' in work.grade %}
        <span class="grade-2">{{ work.grade }}</span>
        { % if forloop.counter >= 3 %}
            <div class="error-banner">
                 Вы исчерпали все попытки (3 из 3) для пересдачи этой работы.
            </div>
        <div class="disabled-button">

```

```

     Пересдача недоступна
    </div>
    {% else %}
        <a href="{% url 'AddCompleteWork' practical_work.idPracticalWork %}"
        class="active-button">
             Отправить исправленную работу (попытка {{ forloop.counter|add:"1" }} из
        3)
        </a>
    {% endif %}
    {% elif work.grade == '3' or work.grade == '4' or work.grade == '5' %}
        <span class="grade-3-5">{{ work.grade }}</span>
        <div class="success-banner">
             Поздравляем! Ваша работа зачтена с оценкой {{ work.grade }}. Пересдача
не требуется.
        </div>
        <div class="disabled-button">
             Повторная отправка недоступна для зачтенных работ
        </div>
    {% else %}
        {{ work.grade }}
    {% endif %}
    </div>
    </div>
    {% endfor %}
    {% else %}
        <div class="warning-banner">
             Вы еще не сдали работу по данной теме. Используйте кнопку ниже для добавления
своей первой работы.
        </div>
        <a href="{% url 'AddCompleteWork' practical_work.idPracticalWork %}" class="active-
button">
             Добавить работу
        </a>
        <!-- Добавляем блок с информацией о попытках, используя user_role_id == 3 вместо
is_student -->
        <div class="attempts-info">
            <h3>Статистика попыток</h3>
            <div class="attempts-container">
                <div class="attempts-summary">
                    <h4>Использовано попыток: {{ used_attempts|length }} из 3</h4>

```

```

<!-- Визуализация попыток кружками -->
<div class="attempt-circles">
    { % for i in "123" % }
    { % if forloop.counter|stringformat:"i" in used_attempts|stringformat:"s" % }
        <div class="attempt-circle attempt-used">{{ forloop.counter }}</div>
    { % else %}
        <div class="attempt-circle attempt-available">{{ forloop.counter }}</div>
    { % endif %}
    { % endfor %}
</div>

<!-- Список использованных попыток -->
<div class="attempts-list">
    <h5>Номера использованных попыток:</h5>
    { % for attempt in used_attempts % }
        <span class="attempt-badge">{{ attempt }}</span>
    { % endfor %}
</div>
</div>

<div class="remaining-attempts">
    { % if remaining_attempts > 0 % }
        <p>У вас осталось <strong>{{ remaining_attempts }}</strong> попыток для сдачи
этой работы.</p>
    { % else %}
        <p>Вы использовали все доступные попытки.</p>
    { % endif %}
</div>
</div>

{ % endif %}
{ % endif %

<!-- Секция для преподавателя -->
{ % if user_role_id == 2 % }
<h2>Сданные работы студентов</h2>

{ % if not students_completed_works % }
<div class="warning-banner">
```

 Пока нет сданных работ по этой практической работе.

```
</div>
{%
  else %
<div class="student-table-container">
  <table class="student-table">
    <thead>
      <tr>
        <th>Студент</th>
        <th>Последняя попытка</th>
        <th>Статус</th>
        <th>Оценка</th>
        <th>Действия</th>
      </tr>
    </thead>
    <tbody>
      {%
        for user_id, student in students_completed_works.items %
        {
          % with latest_work=student.works|first %
          <tr class="student-row">
            <td>{{ student.full_name }}</td>
            <td>{{ latest_work.dateUploadCompletedWork }} в {{ latest_work.timeUploadCompletedWork }}</td>
            <td>
              {%
                if latest_work.statusId == 1 %
                  <span class="status-badge status-passed">Задание сдано</span>
                {%
                  else %
                    <span class="status-badge status-pending">Задание не сдано</span>
                {%
                  endif %
                }
              </td>
              <td>
                {%
                  if latest_work.grade == "На проверке" %
                    <span class="grade-pending">{{ latest_work.grade }}</span>
                  {%
                    elif '2' in latest_work.grade %
                      <span class="grade-2">{{ latest_work.grade }}</span>
                    {%
                      elif latest_work.grade == '3' or latest_work.grade == '4' or latest_work.grade
                        == '5' %
                          <span class="grade-3-5">{{ latest_work.grade }}</span>
                        {%
                          else %
                            {{ latest_work.grade }}
                          {%
                            endif %
                          }
                        </td>
                        <td>
                          <button type="button" class="toggle-details-btn">

```

```

        id="button-{{ user_id }}"
        data-student-id="{{ user_id }}"
        onclick="handleDetails('{{ user_id }}')">>
    Подробнее
    </button>
</td>
</tr>
<tr class="student-details" id="details-{{ user_id }}" style="display: none;">
    <td colspan="5">
        <div class="student-work-details">
            <!-- Форма для просмотра/оценки последней работы -->
            <div class="quick-grade-section">
                <h4>Последняя работа студента</h4>
                <div class="completed-work">
                    <div class="work-header">
                        <span class="attempt-number">Попытка {{ latest_work.attemptCount }} из 3</span>
                        <span class="work-date">{{ latest_work.dateUploadCompletedWork }} в {{ latest_work.timeUploadCompletedWork }}</span>
                    </div>
                    <p>
                        <strong>Файл:</strong>
                        <a href="{{ latest_work.urlFileCompletedWork }}" target="_blank">
                            <span class="file-icon">📄</span>
                            {{ latest_work.nameFileCompletedWork }}
                        </a>
                    </p>
                    <p>
                        <strong>Статус:</strong>
                        {% if latest_work.grade == "На проверке" %}
                            <span class="grade-pending">Требуется проверка</span>
                        {% elif '2' in latest_work.grade %}
                            <span class="grade-2">Не засчитано ({{ latest_work.grade }})</span>
                        {% elif latest_work.grade == '3' or latest_work.grade == '4' or latest_work.grade == '5' %}
                            <span class="grade-3-5">Засчитано ({{ latest_work.grade }})</span>
                        {% else %}
                            {{ latest_work.grade }}
                        {% endif %}
                    </p>

```

```

<!-- Форма оценки только для последней работы, если она на
проверке -->
{%
    if latest_work.grade == "На проверке" %}
    <form      action="{%
        url      'grade_completed_work'
practical_work.idPracticalWork latest_work.userId %}" method="post" class="grade-form">
        {%
            csrf_token %}
        <input type="hidden" name="completed_work_id" value="{{ latest_work.idCompletedWork }}"/>
        <div class="grade-form-group">
            <label for="grade-latest-{{ latest_work.idCompletedWork }}">Оценка:</label>
            <select      name="grade"      id="grade-latest-{{ latest_work.idCompletedWork }}"
latest_work.idCompletedWork %}" required>
                <option value="" disabled selected>Выберите оценку</option>
                <option value="2">2 (Не засчитано)</option>
                <option value="3">3 (Удовлетворительно)</option>
                <option value="4">4 (Хорошо)</option>
                <option value="5">5 (Отлично)</option>
            </select>
            <button      type="submit"      class="grade-submit-btn">Оценить
работу</button>
        </div>
    </form>
{%
    elif '2' in latest_work.grade or latest_work.grade == '3' or
latest_work.grade == '4' or latest_work.grade == '5' %}
    <div class="grade-result">
        <h5>Результат проверки:</h5>
        <p>
            <strong>Итоговая оценка:</strong>
            <span class="{{% if '2' in latest_work.grade %}grade-2{{% else
%}}grade-3-5{{% endif %}}}>
                {{ latest_work.grade }}
            </span>
        </p>
        <div class="{{% if '2' in latest_work.grade %}error-banner{{% else
%}}success-banner{{% endif %}}">
            {{% if '2' in latest_work.grade %}}

```

 Работа не засчитана. Студент может отправить исправленную работу.

```

            {{% else %}}
                 Работа засчитана с оценкой {{ latest_work.grade }}
            {{% endif %}}
    
```

```

        </div>
        </div>
        { % endif %

        </div>
        </div>
        </div>
        </td>
        </tr>
        { % endwith %

        { % endfor %

        </tbody>
        </table>
        </div>
        { % endif %

<div class="export-actions">

    <a href="{% url 'export_students_performance' practical_work.idPracticalWork %}"
class="export-button">
         Выгрузить данные об успеваемости в Word
    </a>
</div>
{ % endif %

<!-- Статистика только для преподавателей -->
{ % if user_role_id == 2 %

<h2>Статистика успеваемости</h2>
<canvas id="gradesChart" style="display: block; box-sizing: border-box; height: 2000px; width: 2000px;" width="320" height="160"></canvas>
{ % endif %

</div>

<script>
"use strict";

// Глобальная функция для переключения отображения деталей - доступна напрямую из HTML
function handleDetails(userId) {
    console.log("Функция handleDetails вызвана с ID:", userId);
    var detailsRow = document.getElementById('details-' + userId);
    var button = document.getElementById('button-' + userId);

```

```

if (detailsRow) {
    // Используем classList.toggle вместо прямой манипуляции style.display
    var isHidden = detailsRow.style.display === 'none' || detailsRow.style.display === '';

    if (isHidden) {
        detailsRow.style.display = 'table-row';
        button.textContent = 'Скрыть';
        console.log("Показываем строку details-" + userId);
    } else {
        detailsRow.style.display = 'none';
        button.textContent = 'Подробнее';
        console.log("Скрываем строку details-" + userId);
    }
} else {
    console.error("Строка details-" + userId + " не найдена!");
    alert("Ошибка: детали не найдены. Пожалуйста, обновите страницу.");
}

// Предотвращаем выполнение других обработчиков
return false;
}

document.addEventListener('DOMContentLoaded', function() {
    // Получаем данные из серверных переменных Django и преобразуем их в JavaScript
    переменные
    var passedCount = parseInt("{% passed_count %}") || 0;
    var notPassedCount = parseInt("{% not_passed_count %}") || 0;

    var chartElement = document.getElementById('gradesChart');
    if (chartElement) {
        var ctx = chartElement.getContext('2d');
        var chart = new Chart(ctx, {
            type: 'bar',
            data: {
                labels: ['Сдано', 'Не сдано'],
                datasets: [{
                    label: 'Количество работ',
                    data: [passedCount, notPassedCount],
                    backgroundColor: [
                        'rgba(76, 175, 80, 0.2)',
                        'rgba(244, 67, 54, 0.2)'
                    ]
                }]
            }
        });
    }
});

```

```

        ],
        borderColor: [
            'rgb(76, 175, 80)',
            'rgb(244, 67, 54)'
        ],
        borderWidth: 1
    }]
},
options: {
    scales: {
        y: {
            beginAtZero: true,
            ticks: {
                precision: 0
            }
        }
    },
    plugins: {
        title: {
            display: true,
            text: 'Распределение оценок по работе'
        }
    }
}
});

}

// ПРОВЕРКА НАЛИЧИЯ ЭЛЕМЕНТОВ В DOM
console.log("Проверка наличия студенческих строк в DOM:");
var studentRows = document.querySelectorAll('.student-row');
console.log("Найдено строк студентов:", studentRows.length);

var detailsRows = document.querySelectorAll('.student-details');
console.log("Найдено строк с деталями:", detailsRows.length);

// Вывод всех ID строк с деталями для отладки
detailsRows.forEach(function(row) {
    console.log("ID строки с деталями:", row.id);
});

// Дублирующий обработчик событий для кнопок (на случай, если прямой onclick не
сработает)

```

```

var detailButtons = document.querySelectorAll('.toggle-details-btn');
console.log("Найдено кнопок 'Подробнее':", detailButtons.length);

detailButtons.forEach(function(button) {
    button.addEventListener('click', function() {
        var studentId = this.getAttribute('data-student-id');
        console.log("Событие click на кнопке для студента с ID:", studentId);
        handleDetails(studentId);
    });
});

// Для совместимости с существующим кодом
window.toggleStudentDetails = handleDetails;
});

// Этот скрипт выполняется в самом конце, когда весь DOM уже загружен
(function() {
    console.log("== ИТОГОВАЯ ПРОВЕРКА КНОПОК ПОДРОБНЕЕ ==");

    // Проверяем, существуют ли кнопки и строки с деталями
    var allButtons = document.querySelectorAll('.toggle-details-btn');
    var allDetailsRows = document.querySelectorAll('.student-details');

    console.log("Найдено кнопок:", allButtons.length);
    console.log("Найдено строк с деталями:", allDetailsRows.length);

    if (allButtons.length === 0 || allDetailsRows.length === 0) {
        console.error("ОШИБКА: Не найдены кнопки или строки с деталями!");
        return;
    }

    // Принудительное добавление обработчиков для всех кнопок
    allButtons.forEach(function(btn) {
        console.log("Повторное добавление обработчика для кнопки:", btn.id);

        // Удаляем существующие обработчики (чтобы избежать дублирования)
        var btnClone = btn.cloneNode(true);
        btn.parentNode.replaceChild(btnClone, btn);

        // Добавляем новый обработчик
        btnClone.addEventListener('click', function(e) {
            e.preventDefault();
        });
    });
});

```

```

e.stopPropagation();

var studentId = this.getAttribute('data-student-id');
console.log("Нажатие на кнопку:", this.id, "для студента:", studentId);

var detailsRow = document.getElementById('details-' + studentId);
if (!detailsRow) {
    console.error("Не найдена строка details-" + studentId);
    return;
}

// Простое переключение видимости
if (detailsRow.style.display === 'table-row') {
    detailsRow.style.display = 'none';
    this.textContent = 'Подробнее';
} else {
    detailsRow.style.display = 'table-row';
    this.textContent = 'Скрыть';
}
});

// Также установим атрибут onclick напрямую
btnClone.setAttribute('onclick', 'event.preventDefault(); var row = document.getElementById("details-' +
    btnClone.getAttribute('data-student-id') + ""); if(row) { row.style.display = row.style.display === "table-row" ? "none" : "table-row"; this.textContent = row.style.display === "table-row" ? "Скрыть" : "Подробнее"; } return false;');
});

console.log("Обработчики успешно переустановлены для всех кнопок.");
})());

```

## SuccessMessage.html

```

<!-- SuccessMessage.html -->
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Работа прикреплена</title>

```

```

{ % load static % }

<link rel="stylesheet" href="{% static 'css/SubjectPage.css' %}">
</head>
<body>
<div class="container">
    <h1>Вы прикрепили работу!</h1>
    <p><strong>Имя файла:</strong> {{ name_file_completed_work }}</p>
    <p><strong>Дата:</strong> {{ date_upload }}</p>
    <p><strong>Время:</strong> {{ time_upload }}</p>

    <a href="{% url 'SubjectPage' subject_id %}" class="back-button">Назад к предмету</a>
</div>
</body>
</html>

```

## TestAttempts.html

```

<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    { % load static % }
    { % load custom_filters % }
    <title>Попытки прохождения теста - {{ test.nameTest }}</title>
    <link rel="stylesheet" href="{% static 'css/TestAttempts.css' %}">
</head>
<body>
    { % include 'header.html' % }

    <div class="container">
        <h1>Попытки прохождения теста "{{ test.nameTest }}"</h1>

        <div class="test-info">
            <p><strong>Описание:</strong> {{ test.descriptionTest }}</p>
            <p><strong>Ограничение по времени:</strong> {{ test.timeLimit }} минут</p>
        </div>

        { % if grading_criteria % }
        <div class="grading-criteria">
            <h2>Критерии оценивания</h2>
            <table class="criteria-table">
                <thead>
                    <tr>

```

```

<th>Оценка</th>
<th>Минимальные баллы</th>
<th>Максимальные баллы</th>
</tr>
</thead>
<tbody>
{ % for criterion in grading_criteria % }

<tr>
<td><strong data-grade="{ { criterion.grade } }">{ { criterion.grade } }</strong></td>
<td>{ { criterion.minPoints } }</td>
<td>{ { criterion.maxPoints } }</td>
</tr>

{ % endfor %}

<tr class="grade-2-row">
<td><strong data-grade="2">2</strong></td>
<td colspan="2">
{ % with grade_3=grading_criteria|filter_by_grade:3 %}

{ % if grade_3 %}
    Менее {{ grade_3.minPoints }} баллов
{ % else %}
    Недостаточно баллов
{ % endif %}

{ % endwith %}

</td>
</tr>
</tbody>
</table>
</div>

{ % endif %}

<div class="students-attempts">
{ % if students %}

{ % for user_id, student in students.items %}

<div class="student-card">
<h2>{{ student.full_name }}</h2>

{ % if student.attempts %}

<table class="attempts-table">
<thead>
<tr>
<th>Дата</th>
<th>Время</th>

```

```

<th>Статус</th>
<th>Баллы</th>
<th>Оценка</th>
</tr>
</thead>
<tbody>
{ % for attempt in student.attempts % }

<tr>
<td>{ { attempt.endTime|date:"d.m.Y"|default:"-" } }</td>
<td>{ { attempt.endTime|time:"H:i"|default:"-" } }</td>
<td>
{ % if attempt.status == 'completed' % }
<span class="status completed">Завершено</span>
{ % elif attempt.status == 'in_progress' % }
<span class="status in-progress">В процессе</span>
{ % else % }
<span class="status">{ { attempt.status } }</span>
{ % endif %}
</td>
<td>{ { attempt.score|default:"-" } }</td>
<td>
{ % if attempt.grade % }
<strong data-grade="{ { attempt.grade } }">{ { attempt.grade } }</strong>
{ % else % }
-
{ % endif %}
</td>
</tr>
{ % endfor % }

</tbody>
</table>

{ % else % }
<p class="no-attempts">Нет попыток прохождения теста</p>
{ % endif %}
</div>
{ % endfor % }

{ % else % }
<p class="no-students">Нет данных о попытках прохождения теста</p>
{ % endif %}
</div>

<div class="actions">
```

```

<a href="{% url 'test_material' test.idTest %}" class="button">Вернуться к материалам теста</a>
<a href="{% url 'SubjectPage' test.subjectId %}" class="button">Вернуться к предмету</a>
</div>
</div>
</body>
</html>

```

## TestMaterial.html

```

<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    {% load static %}
    {% load custom_filters %}
    <title>{{ test.nameTest }} - Материалы</title>
    <link rel="stylesheet" href="{% static 'css/TestMaterial.css' %}">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css">
</head>
<body>
    {% include 'header.html' %}

    <div class="container">
        <div class="page-header">
            <div class="header-content">
                <h1>{{ test.nameTest }}</h1>
                <p class="description">{{ test.descriptionTest }}</p>
                <div class="meta-info">
                    <div class="meta-item">
                        <i class="fas fa-clock"></i>
                        <span>Время на выполнение: {{ test.timeLimit }} минут</span>
                    </div>
                    {% if subject_name %}
                    <div class="meta-item">
                        <i class="fas fa-book"></i>
                        <span>Предмет: {{ subject_name }}</span>
                    </div>
                    {% endif %}
                </div>
            </div>
        </div>
    </div>

    <!-- Табы навигации -->

```

```

<div class="tabs-container">
  <div class="tabs-nav">
    <button class="tab-link active" onclick="openTab(event, 'test-info-tab')">
      <i class="fas fa-info-circle"></i> Информация
    </button>
    {% if teacher_view %}
      <button class="tab-link" onclick="openTab(event, 'questions-tab')">
        <i class="fas fa-question-circle"></i> Вопросы
      </button>
    {% endif %}
    <button class="tab-link" onclick="openTab(event, 'criteria-tab')">
      <i class="fas fa-star"></i> Критерии оценивания
    </button>
    {% if teacher_view %}
      <button class="tab-link" onclick="openTab(event, 'results-tab')">
        <i class="fas fa-chart-bar"></i> Результаты
      </button>
    {% endif %}
  </div>

  <!-- Вкладка с информацией о тесте -->
  <div id="test-info-tab" class="tab-content active">
    <div class="section-header">
      <h2>Информация о тесте</h2>
    </div>
    <div class="test-info card">
      <div class="card-content">
        <div class="info-row">
          <span class="info-label">Название:</span>
          <span class="info-value">{{ test.nameTest }}</span>
        </div>
        <div class="info-row">
          <span class="info-label">Описание:</span>
          <span class="info-value">{{ test.descriptionTest }}</span>
        </div>
        <div class="info-row">
          <span class="info-label">Ограничение по времени:</span>
          <span class="info-value">{{ test.timeLimit }} минут</span>
        </div>
      </div>
    {% if student_view %}
      <div class="test-status"

```

```

{ % if latest_attempt.grade and latest_attempt.grade >= 3 % }passed
{ % elif attempts_count > 0 % }failed
{ % else % }attempts{ % endif % }">

{ % if latest_attempt.grade and latest_attempt.grade >= 3 % }
<p><i class="fas fa-check-circle"></i> Тест успешно сдан с оценкой {{ latest_attempt.grade }}</p>
{ % elif attempts_count > 0 % }
<p><i class="fas fa-exclamation-circle"></i> Тест не сдан. У вас осталось {{ remaining_attempts }}</p>
{ % if remaining_attempts == 1 %}попытка{ % elif remaining_attempts > 1 and remaining_attempts < 5 %}попытки{ % else %}попыток{ % endif %}</p>
{ % else % }
<p><i class="fas fa-info-circle"></i> У вас есть {{ max_attempts }}</p>
{ % if max_attempts == 1 %}попытка{ % elif max_attempts > 1 and max_attempts < 5 %}попытки{ % else %}попыток{ % endif %} для сдачи теста</p>
{ % endif %}
</div>
{ % endif %}
</div>
</div>

<div class="actions">
{ % if teacher_view %}
<a href="{ % url 'edit_test' test.idTest % }" class="button primary">
<i class="fas fa-edit"></i> Редактировать тест
</a>
<a href="{ % url 'test_students_results' test.idTest % }" class="button secondary">
<i class="fas fa-users"></i> Результаты студентов
</a>
<form method="POST" action="{ % url 'delete_test' test.idTest % }" style="display: inline;">
{ % csrf_token %
<button type="submit" class="btn btn-danger" onclick="return confirm('Вы уверены, что хотите удалить этот тест? Это действие нельзя отменить.');" ">
Удалить тест
</button>
</form>
{ % elif student_view %}
<div class="actions">
{ % if latest_attempt and latest_attempt.grade and latest_attempt.grade >= 3 % }
<a href="{ % url 'test_results' test.idTest % }" class="button primary">
<i class="fas fa-poll"></i> Просмотр результатов

```

```

        </a>
        {% elif attempts_count == 0 or remaining_attempts > 0 %}
            <a href="{% url 'start_test' test.idTest %}" class="button success">
                <i class="fas fa-play"></i> Начать тест
            </a>
        {% if attempts_count > 0 %}
            <a href="{% url 'test_results' test.idTest %}" class="button secondary">
                <i class="fas fa-poll"></i> Просмотр результатов
            </a>
        {% endif %}
        {% else %}
            <a href="{% url 'test_results' test.idTest %}" class="button primary">
                <i class="fas fa-poll"></i> Просмотр результатов
            </a>
            <span class="button disabled">
                <i class="fas fa-ban"></i> Тест не сдан, попыток не осталось
            </span>
        {% endif %}
    </div>
    {% endif %}

    {% if test.subjectId %}
        <a href="{% url 'SubjectPage' test.subjectId %}" class="button back">
            <i class="fas fa-arrow-left"></i> Вернуться к предмету
        </a>
    {% elif subject_id %}
        <a href="{% url 'SubjectPage' subject_id %}" class="button back">
            <i class="fas fa-arrow-left"></i> Вернуться к предмету
        </a>
    {% else %}
        <a href="{% url 'AllSubjectPage' %}" class="button back">
            <i class="fas fa-arrow-left"></i> Вернуться к предметам
        </a>
    {% endif %}
    </div>
</div>

<!-- Вкладка с вопросами (только для преподавателя) --&gt;
{% if teacher_view %}
&lt;div id="questions-tab" class="tab-content"&gt;
    &lt;div class="section-header"&gt;
        &lt;h2&gt;Вопросы теста&lt;/h2&gt;
</pre>

```

```

</div>
{ % if questions % }
<div class="questions-preview">
{ % for question in questions % }
<div class="question-card card">
<div class="card-header">
<h3>Вопрос {{ forloop.counter }}</h3>
<span class="question-type">
{ % if question.questionType == 1 % }
<i class="fas fa-dot-circle"></i> Один правильный ответ
{ % elif question.questionType == 2 % }
<i class="fas fa-check-square"></i> Несколько правильных ответов
{ % elif question.questionType == 3 % }
<i class="fas fa-exchange-alt"></i> Сопоставление
{ % elif question.questionType == 4 % }
<i class="fas fa-keyboard"></i> Ввод ответа
{ % elif question.questionType == 5 % }
<i class="fas fa-sort-numeric-down"></i> Установление порядка
{ % else % }
<i class="fas fa-question"></i> Неизвестный тип
{ % endif %}
</span>
</div>
<div class="card-content">
<p class="question-text">{{ question.questionText }}</p>

{ % if question.answerVariants % }
<div class="answers">
<p class="answers-header"><strong>Варианты ответов:</strong></p>
<ul class="answer-list">
{ % for answer in question.answerVariants|split:';' % }
<li class="answer-item">{{ answer }}</li>
{ % endfor %}
</ul>
</div>
{ % endif %}
</div>
{ % endfor %}
</div>
{ % else % }
<div class="empty-state">

```

```

<i class="fas fa-question-circle empty-icon"></i>
<p>Нет доступных вопросов для этого теста.</p>
</div>
{%- endif %}
</div>
{%- endif %}

<!-- Вкладка с критериями оценивания --&gt;
<div id="criteria-tab" class="tab-content">


<h2>Критерии оценивания</h2>
</div>
{%- if grading_criteria %}
<div class="grading-criteria card">
<div class="card-content">
<table class="criteria-table">
<thead>
<tr>
<th>Оценка</th>
<th>Минимальные баллы</th>
<th>Максимальные баллы</th>
</tr>
</thead>
<tbody>
{%- for criterion in grading_criteria %}
<tr>
<td><strong data-grade="{{ criterion.grade }}">{{ criterion.grade }}</strong></td>
<td>{{ criterion.minPoints }}</td>
<td>{{ criterion.maxPoints }}</td>
</tr>
{%- endfor %}
<tr class="grade-2-row">
<td><strong data-grade="2">2</strong></td>
<td colspan="2">
{%- with grade_3=grading_criteria|filter_by_grade:3 %}
{%- if grade_3 %}
    Менее {{ grade_3.minPoints }} баллов
{%- else %}
    Недостаточно баллов
{%- endif %}
{%- endwith %}



338


```

```

        </td>
    </tr>
</tbody>
</table>
</div>
</div>

{% else %}

<div class="empty-state">
    <i class="fas fa-star empty-icon"></i>
    <p>Критерии оценивания не указаны.</p>
</div>

{% endif %}

</div>

<!-- Вкладка с результатами (только для преподавателя) --&gt;
{% if teacher_view %}

&lt;div id="results-tab" class="tab-content"&gt;
    &lt;div class="section-header"&gt;
        &lt;h2&gt;Результаты студентов&lt;/h2&gt;
    &lt;/div&gt;
    &lt;div class="results-info card"&gt;
        &lt;div class="card-content"&gt;
            &lt;p&gt;Для просмотра полной информации о результатах студентов перейдите на страницу результатов:&lt;/p&gt;
            &lt;div class="actions" style="justify-content: flex-start; margin-top: 20px;"&gt;
                &lt;a href="{% url 'test_students_results' test.idTest %}" class="button primary"&gt;
                    &lt;i class="fas fa-users"&gt;&lt;/i&gt; Просмотр результатов студентов
                &lt;/a&gt;
            &lt;/div&gt;
        &lt;/div&gt;
    &lt;/div&gt;
    {% endif %}
&lt;/div&gt;

&lt;style&gt;
.tabs-container {
    margin-bottom: 2rem;
}
.tabs-nav {
</pre>

```

```
display: flex;
flex-wrap: wrap;
gap: 0.5rem;
margin-bottom: 1.5rem;
}

.tab-link {
  padding: 0.8rem 1.2rem;
  background-color: rgba(255, 255, 255, 0.1);
  color: var(--text-secondary);
  border: none;
  border-radius: 8px;
  cursor: pointer;
  font-weight: 600;
  font-size: 0.95rem;
  transition: all 0.2s;
  display: flex;
  align-items: center;
  box-shadow: var(--shadow-sm);
}

.tab-link i {
  margin-right: 0.5rem;
  color: var(--light-purple);
}

.tab-link:hover {
  background-color: rgba(255, 255, 255, 0.2);
  color: var(--text-light);
  transform: translateY(-2px);
}

.tab-link.active {
  background-color: var(--accent-purple);
  color: white;
}

.tab-content {
  display: none;
  animation: fadeIn 0.3s ease-in-out;
}
```

```
.tab-content.active {
    display: block;
}

@keyframes fadeIn {
    from { opacity: 0; transform: translateY(10px); }
    to { opacity: 1; transform: translateY(0); }
}

/* Секции */
.section-header {
    margin-bottom: 1.5rem;
    position: relative;
}

.section-header h2 {
    font-size: 1.5rem;
    color: var(--text-light);
    margin: 0;
    padding-bottom: 0.5rem;
    display: inline-block;
    position: relative;
}

.section-header h2::after {
    content: "";
    position: absolute;
    bottom: 0;
    left: 0;
    width: 50px;
    height: 3px;
    background-color: var(--accent-purple);
}

/* Карточки */
.card {
    background-color: var(--card-bg);
    border-radius: var(--border-radius);
    box-shadow: var(--shadow-sm);
    margin-bottom: 1.5rem;
    overflow: hidden;
    transition: transform 0.2s, box-shadow 0.2s;
```

```
border: 1px solid rgba(255, 255, 255, 0.05);
}

.card:hover {
    transform: translateY(-3px);
    box-shadow: var(--shadow-md);
    background-color: var(--card-hover);
}

.card-header {
    padding: 1.2rem 1.5rem;
    border-bottom: 1px solid rgba(255, 255, 255, 0.1);
    display: flex;
    justify-content: space-between;
    align-items: center;
}

.card-content {
    padding: 1.5rem;
}

/* Информационные поля */
.info-row {
    display: flex;
    margin-bottom: 1rem;
    align-items: flex-start;
}

.info-label {
    font-weight: 600;
    color: var(--accent-purple);
    width: 200px;
    flex-shrink: 0;
}

.info-value {
    flex: 1;
    color: var(--text-secondary);
}

/* Кнопки */
.actions {
```

```
display: flex;
flex-wrap: wrap;
gap: 1rem;
margin-top: 2rem;
justify-content: center;
}

.button {
    display: inline-flex;
    align-items: center;
    padding: 0.8rem 1.5rem;
    border-radius: 8px;
    font-weight: 600;
    text-decoration: none;
    cursor: pointer;
    transition: all 0.2s;
    box-shadow: var(--shadow-sm);
}

.button i {
    margin-right: 0.5rem;
}

.button.primary {
    background-color: var(--accent-purple);
    color: white;
}

.button.primary:hover {
    background-color: #8637c6;
    transform: translateY(-2px);
    box-shadow: var(--shadow-md);
}

.button.secondary {
    background-color: rgba(255, 255, 255, 0.1);
    color: var(--text-light);
}

.button.secondary:hover {
    background-color: rgba(255, 255, 255, 0.2);
    transform: translateY(-2px);
}
```

```
    box-shadow: var(--shadow-md);  
}  
  
.button.success {  
    background-color: var(--success);  
    color: white;  
}  
  
.button.success:hover {  
    background-color: var(--success-dark);  
    transform: translateY(-2px);  
    box-shadow: var(--shadow-md);  
}  
  
.button.danger {  
    background-color: var(--danger);  
    color: white;  
}  
  
.button.danger:hover {  
    background-color: var(--danger-dark);  
    transform: translateY(-2px);  
    box-shadow: var(--shadow-md);  
}  
  
.button.back {  
    background-color: rgba(255, 255, 255, 0.05);  
    color: var(--text-secondary);  
}  
  
.button.back:hover {  
    background-color: rgba(255, 255, 255, 0.1);  
    color: var(--text-light);  
    transform: translateY(-2px);  
    box-shadow: var(--shadow-md);  
}  
  
.button.disabled {  
    background-color: rgba(255, 255, 255, 0.05);  
    color: rgba(255, 255, 255, 0.3);  
    cursor: not-allowed;  
    pointer-events: none;
```

```
}

/* Вопросы */
.question-text {
    font-size: 1.1rem;
    margin-bottom: 1rem;
    line-height: 1.5;
    color: var(--text-light);
}

.question-type {
    color: var(--light-purple);
    font-size: 0.9rem;
    display: flex;
    align-items: center;
}

.question-type i {
    margin-right: 0.5rem;
}

.answers-header {
    margin-bottom: 0.8rem;
    color: var(--light-purple);
}

.answer-list {
    list-style-type: none;
    padding: 0;
    margin: 0;
}

.answer-item {
    padding: 0.6rem 1rem;
    margin-bottom: 0.5rem;
    background-color: rgba(255, 255, 255, 0.05);
    border-radius: 4px;
    border-left: 3px solid var(--accent-purple);
    transition: background-color 0.2s;
    color: var(--text-secondary);
}
```

```
.answer-item:hover {
    background-color: rgba(255, 255, 255, 0.1);
}

/* Статус теста */

.test-status {
    margin-top: 1.5rem;
    padding: 1rem;
    border-radius: 8px;
    font-weight: 500;
    display: flex;
    align-items: center;
}

.test-status i {
    margin-right: 0.8rem;
    font-size: 1.2rem;
}

.test-status.passed {
    background-color: rgba(76, 175, 80, 0.15);
    color: #A5D6A7;
    border-left: 4px solid var(--success);
}

.test-status.failed {
    background-color: rgba(244, 67, 54, 0.15);
    color: #EF9A9A;
    border-left: 4px solid var(--danger);
}

.test-status.attempts {
    background-color: rgba(255, 193, 7, 0.15);
    color: #FFE082;
    border-left: 4px solid var(--warning);
}

/* Таблица критериев - оставляем текущие стили из CSS файла */

/* Пустое состояние */

.empty-state {
    text-align: center;
```

```
padding: 3rem 1rem;  
background-color: rgba(0, 0, 0, 0.2);  
border-radius: 10px;  
border: 1px dashed rgba(255, 255, 255, 0.1);  
margin-bottom: 2rem;  
}
```

```
.empty-icon {  
    font-size: 3rem;  
    color: var(--accent-purple);  
    margin-bottom: 1rem;  
    display: block;  
    opacity: 0.5;  
}
```

```
.empty-state p {  
    color: var(--text-secondary);  
    font-style: italic;  
}
```

```
/* Медиа-запросы для адаптивности */  
@media (max-width: 768px) {  
    .info-row {  
        flex-direction: column;  
    }
```

```
.info-label {  
    width: 100%;  
    margin-bottom: 0.3rem;  
}
```

```
.tab-link {  
    flex: 1;  
    justify-content: center;  
    padding: 0.6rem 0.8rem;  
}
```

```
.actions {  
    flex-direction: column;  
}
```

```
.button {
```

```

        width: 100%;

        justify-content: center;
    }
}

</style>

<script>

function openTab(evt, tabName) {
    // Скрываем все вкладки
    var tabContents = document.getElementsByClassName("tab-content");
    for (var i = 0; i < tabContents.length; i++) {
        tabContents[i].classList.remove("active");
    }

    // Убираем активный класс с кнопок
    var tabLinks = document.getElementsByClassName("tab-link");
    for (var i = 0; i < tabLinks.length; i++) {
        tabLinks[i].classList.remove("active");
    }

    // Показываем выбранную вкладку и активируем кнопку
    document.getElementById(tabName).classList.add("active");
    evt.currentTarget.classList.add("active");
}

</script>
</body>
</html>
```

## TestProcess.html

```

<!DOCTYPE html>
<html lang="ru">
<head>

    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    {% load static %}

    {% load custom_filters %}

    <title>{{ test.nameTest }} - Тестирование</title>
    <link rel="stylesheet" href="{{ static 'css/TestProcess.css' }}>
    <script src="{{ static 'js/shuffle_test.js' }}></script>
    <script src="{{ static 'js/timer.js' }}></script>
    <script src="{{ static 'js/matching.js' }}></script>
    <script src="{{ static 'js/ordering.js' }}?v=2"></script>

</head>
```

```

<body>
    {% include 'header.html' %}

    <div class="container">
        <div class="test-header">
            <h1>{{ test.nameTest }}</h1>
            <div class="timer" id="timer">
                Оставшееся время: <span id="time-remaining">{{ time_limit }}:00</span>
            </div>
        </div>

        <form method="POST" action="{% url 'submit_test' test.idTest %}" id="test-form" onsubmit="return validateTestForm()">
            {% csrf_token %}
            <input type="hidden" name="attempt_id" value="{{ attempt_id }}">
            <input type="hidden" name="start_time" value="{{ start_time }}">

            <div class="questions">
                {% for question in questions %}
                    <div class="question-card">
                        data-question-id="{{ question.idQuestion }}"
                        data-question-type="{{ question.questionType }}"
                        <h3>Вопрос {{ forloop.counter }}</h3>
                        <p class="question-text">{{ question.questionText }}</p>

                        {% if question.questionType == 1 %}
                            
                            {% for answer in question.answerVariants|split:';' %}
                                <div class="answer-option">
                                    <input type="radio"
                                           name="question_{{ question.idQuestion }}"
                                           id="q{{ question.idQuestion }}_a{{ forloop.counter }}"
                                           value="{{ answer }}"/>
                                    <label for="q{{ question.idQuestion }}_a{{ forloop.counter }}">{{ answer }}</label>
                                </div>
                            {% endfor %}

                            {% elif question.questionType == 2 %}
                                
                                {% for answer in question.answerVariants|split:';' %}
                                    <div class="answer-option">
                                        <input type="checkbox"
                                               name="question_{{ question.idQuestion }}"
                                               id="q{{ question.idQuestion }}_a{{ forloop.counter }}"
                                               value="{{ answer }}"/>
                                    </div>
                                {% endfor %}
                            {% endif %}
                        {% endfor %}
                    </div>
                {% endfor %}
            </div>
        </form>
    </div>

```

```

        name="question_{{ question.idQuestion }}_option_{{ forloop.counter }}"
        id="q{{ question.idQuestion }}_a{{ forloop.counter }}"
        value="{{ answer }}"
    <label for="q{{ question.idQuestion }}_a{{ forloop.counter }}">{{ answer }}</label>
</div>
{%- endfor %}

{%- elif question.questionType == 3 %}
<!-- Сопоставление --&gt;
&lt;div class="matching-container" id="matching_{{ question.idQuestion }}"&gt;
    &lt;div class="matching-instructions"&gt;
        &lt;p&gt;Сопоставьте элементы из левой колонки с элементами из правой колонки,<br/>перетаскивая точки соединения.</p>
    </div>

    {%- with items=question.answerVariants|split:';' %}
        <div class="matching-area">
            <div class="matching-left-column">
                {%- for item in items %}
                    <div class="matching-item left-item" data-id="{{ forloop.counter0 }}" data-
value="{{ item|split:'-'|first }}">
                        <span class="item-text">{{ item|split:'-'|first }}</span>
                        <div class="connection-point left-point" data-id="{{ forloop.counter0
}}"></div>
                    </div>
                {%- empty %}
                <div class="matching-item left-item" data-id="0" data-value="Нет данных">
                    <span class="item-text">Нет данных для сопоставления</span>
                    <div class="connection-point left-point" data-id="0"></div>
                </div>
            {%- endfor %}
        </div>

        <div class="matching-canvas-container">
            <canvas id="matching-canvas-{{ question.idQuestion }}" class="matching-
canvas"></canvas>
        </div>

        <div class="matching-right-column">
            {%- for item in items|randomize %}
                <div class="matching-item right-item" data-id="{{ forloop.counter0 }}" data-
value="{{ item|split:'-'|last }}">

```

```

<div class="connection-point right-point" data-id="{{ forloop.counter0 }}"></div>
    <span class="item-text">{{ item|split:'-'|last }}</span>
</div>
    {% empty %}
<div class="matching-item right-item" data-id="0" data-value="Нет данных">
    <div class="connection-point right-point" data-id="0"></div>
        <span class="item-text">Нет данных для сопоставления</span>
    </div>
    {% endfor %}
</div>
</div>

<!-- Скрытые поля для сопоставлений --&gt;
&lt;div id="matching_inputs_{{ question.idQuestion }}" class="matching-hidden-inputs"&gt;
    &lt;input type="hidden" name="question_{{ question.idQuestion }}"
value="matched"&gt;
    {% with items=question.answerVariants|split:',' %}
        {% for item in items %}
            {% with pair=item|split:'-' %}
                &lt;div class="match-pair" data-left="{{ pair.0|safe }}" data-right="{{ pair.1|safe }}"&gt;
                    &lt;input type="hidden"
name="question_{{ question.idQuestion }}_left_{{ forloop.counter0 }}"
value="{{ pair.0|safe }}"
id="left_{{ question.idQuestion }}_{{ forloop.counter0 }}"&gt;
                    &lt;input type="hidden"
name="question_{{ question.idQuestion }}_match_{{ forloop.counter0 }}"
value=""
id="match_{{ question.idQuestion }}_{{ forloop.counter0 }}"&gt;
                &lt;/div&gt;
            {% endwith %}
        {% endfor %}
    {% endwith %}
&lt;/div&gt;

<!-- Добавляем кнопку для сброса соединений --&gt;
&lt;div class="matching-controls"&gt;
</pre>

```

```

<button type="button" class="reset-connections-btn" data-question-id="{{ question.idQuestion }}">
    Сбросить соединения
</button>
<div class="connection-counter">
    <span id="connection-count-{{ question.idQuestion }}>0</span> из <span>{{ items|length }}</span> соединений
    </div>
    </div>
    { % endwith % }
</div>

{ % elif question.questionType == 4 % }
<!-- Ввод ответа --&gt;
&lt;div class="answer-input"&gt;
    &lt;input type="text"
        name="question_{{ question.idQuestion }}"
        placeholder="Введите ответ"&gt;
&lt;/div&gt;

{ % elif question.questionType == 5 % }
<!-- Установление порядка --&gt;
&lt;div class="order-container" id="order_{{ question.idQuestion }}"&gt;
    &lt;div class="order-instructions"&gt;
        &lt;p&gt;Расположите элементы в правильном порядке, перетаскивая их.&lt;/p&gt;
    &lt;/div&gt;

    &lt;div class="order-items-container"&gt;
        { % for answer in question.answerVariants|split:'|randomize % }
        &lt;div class="order-item" draggable="true" data-value="{{ answer }}&gt;
            &lt;span class="order-number"&gt;{{ forloop.counter }}&lt;/span&gt;
            &lt;span class="order-text"&gt;{{ answer }}&lt;/span&gt;
            &lt;span class="order-handle"&gt;::&lt;/span&gt;
        &lt;/div&gt;
        { % endfor % }
    &lt;/div&gt;

    &lt;!-- Скрытые поля для отправки порядка --&gt;
    &lt;div class="order-hidden-inputs" id="order_inputs_{{ question.idQuestion }}"&gt;
        &lt;!-- Скрытые поля будут добавляться динамически --&gt;
    &lt;/div&gt;
&lt;/div&gt;
</pre>

```

```

        { % endif %

    </div>
    { % endfor %

</div>

<div class="form-actions">
    <button type="submit" class="submit-btn" id="submit-test">Завершить тест</button>
</div>
</form>
</div>

<script>
// Таймер

function startTimer(duration) {
    let timer = duration * 60;
    const display = document.getElementById('time-remaining');

    const countdown = setInterval(function () {
        const minutes = parseInt(timer / 60, 10);
        const seconds = parseInt(timer % 60, 10);

        display.textContent = minutes.toString().padStart(2, '0') + ':' +
            seconds.toString().padStart(2, '0');

        if (--timer < 0) {
            clearInterval(countdown);
            // Автоматически отправляем форму при истечении времени
            alert('Время истекло! Тест будет автоматически отправлен.');
            document.getElementById('test-form').submit();
        }
    }, 1000);
}

// Функция для проверки заполнения всех вопросов (теперь только для предупреждения)
function checkUnansweredQuestions() {
    let unansweredCount = 0;

    // Проверяем вопросы с одним правильным ответом
    const radioGroups = document.querySelectorAll('input[type="radio"]');
    const radioGroupNames = new Set();
    radioGroups.forEach(radio => radioGroupNames.add(radio.name));
}

```

```

radioGroupNames.forEach(name => {
    const checkedRadio = document.querySelector(`input[name="${name}"]:checked`);
    if (!checkedRadio) {
        unansweredCount++;
        // Находим карточку вопроса и добавляем класс предупреждения
        const questionCard = document.querySelector(`input[name="${name}"]`).closest('.question-card');
        questionCard.classList.add('warning');
    }
});

// Проверяем вопросы с вводом ответа
const textInputs = document.querySelectorAll('input[type="text"]');
textInputs.forEach(input => {
    if (!input.value.trim()) {
        unansweredCount++;
        // Находим карточку вопроса и добавляем класс предупреждения
        const questionCard = input.closest('.question-card');
        questionCard.classList.add('warning');
    }
});

// Проверяем вопросы на сопоставление
const matchingContainers = document.querySelectorAll('.matching-container');
matchingContainers.forEach(container => {
    const questionId = container.id.split('_')[1];
    const leftPoints = container.querySelectorAll('.left-point');
    const hiddenInputs = container.querySelectorAll('input[type="hidden"]');

    // Проверяем, что количество соединений равно количеству левых точек
    if (hiddenInputs.length !== leftPoints.length) {
        unansweredCount++;
        container.classList.add('warning');
    }

    // Находим карточку вопроса и добавляем класс предупреждения
    const questionCard = container.closest('.question-card');
    questionCard.classList.add('warning');
}

// Проверяем вопросы на установление порядка
const orderContainers = document.querySelectorAll('.order-container');

```

```

orderContainers.forEach(container => {
  const questionId = container.id.split('_')[1];
  const hiddenInputs = container.querySelectorAll('input[type="hidden"]');

  // Проверяем, что есть хотя бы один скрытый input
  if (hiddenInputs.length === 0) {
    unansweredCount++;
    container.classList.add('warning');

    // Находим карточку вопроса и добавляем класс предупреждения
    const questionCard = container.closest('.question-card');
    questionCard.classList.add('warning');
  }
});

return unansweredCount;
}

// Функция для обработки вопросов на установление порядка
function setupOrderQuestions() {
  const orderContainers = document.querySelectorAll('.order-container');

  orderContainers.forEach(container => {
    const questionId = container.id.split('_')[1];
    const items = container.querySelectorAll('.order-item');
    const hiddenInputsContainer = document.getElementById(`order_inputs_${questionId}`);

    // Создаем скрытые поля для отправки порядка
    function updateHiddenInputs() {
      // Очищаем контейнер
      hiddenInputsContainer.innerHTML = "";

      // Получаем текущий порядок элементов
      const currentItems = container.querySelectorAll('.order-item');

      // Создаем скрытые поля с текущим порядком
      currentItems.forEach((item, index) => {
        const input = document.createElement('input');
        input.type = 'hidden';
        input.name = `
```

## TestResults.html

<!DOCTYPE html>

```

<html lang="ru">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    {% load static %}
    {% load custom_filters %}
    <title>Результаты теста - {{ test.nameTest }}</title>
    <link rel="stylesheet" href="{% static 'css/TestResults.css' %}">
  </head>
  <body>
    {% include 'header.html' %}

    <div class="container">
      <h1>Результаты теста "{{ test.nameTest }}"</h1>

      <div class="results-card">
        <h2>Ваш последний результат</h2>
        {% if latest_attempt %}
          <div class="score-info">
            <p class="score">Набрано баллов: <strong>{{ latest_attempt.score }}</strong></p>
            <p class="grade">Оценка: <strong data-grade="{{ latest_attempt.grade }}"/>{{ latest_attempt.grade }}</strong></p>
            <p class="completion-time">Время завершения:
              {% if latest_attempt.endTime %}
                {{ latest_attempt.endTime|slice:"10" }} {{ latest_attempt.endTime|slice:"11:19" }}
              {% else %}
                Нет данных
              {% endif %}
            </p>
            {% if latest_attempt.grade >= 3 %}
              <div class="retake-info success">
                <p>Вы успешно сдали тест! Повторное прохождение невозможно.</p>
              </div>
            {% else %}
              <div class="retake-info {% if can_retake %}warning{% else %}danger{% endif %}">
                {% if can_retake %}
                  <p>У вас осталось {{ remaining_attempts }}. {% if remaining_attempts == 1 %}<br>%</p>
                  попытка<% elif remaining_attempts > 1 and remaining_attempts < 5 %> попытки<% else %> попыток<% endif %> для пересдачи теста.</p>
                  <a href="{% url 'test_material' test.idTest %}" class="retake-btn">Пересдать тест</a>
                {% else %}

```

```

<p>Вы исчерпали все попытки для пересдачи теста.</p>
{ % endif %
</div>
{ % endif %
</div>
{ % else %

<p>У вас пока нет попыток прохождения теста.</p>
{ % endif %
</div>

<div class="grading-criteria">
<h2>Критерии оценивания</h2>
{ % if grading_criteria %
<div class="criteria-table">
<table>
<thead>
<tr>
<th>Оценка</th>
<th>Минимальные баллы</th>
<th>Максимальные баллы</th>
</tr>
</thead>
<tbody>
{ % for criterion in grading_criteria %
<tr>
<td><strong data-grade="{{ criterion.grade }}">{{ criterion.grade }}</strong></td>
<td>{{ criterion.minPoints }}</td>
<td>{{ criterion.maxPoints }}</td>
</tr>
{ % endfor %
<tr class="grade-2-row">
<td><strong data-grade="2">2</strong></td>
<td colspan="2">
{ % with grade_3=grading_criteria|filter_by_grade:3 %
{ % if grade_3 %
    Менее {{ grade_3.minPoints }} баллов
{ % else %
    Недостаточно баллов
{ % endif %
{ % endwith %
</td>
</tr>

```

```

        </tbody>
    </table>
</div>
{ % else %
    <p>Критерии оценивания не указаны.</p>
{ % endif %
</div>

<div class="previous-attempts">
    <h2>История попыток</h2>
    <table>
        <thead>
            <tr>
                <th>Дата</th>
                <th>Время</th>
                <th>Баллы</th>
                <th>Оценка</th>
            </tr>
        </thead>
        <tbody>
            { % for attempt in attempts %}
                <tr>
                    <td>{%
                        if attempt.endTime %
                            {{ attempt.endTime|slice:"10" }}%
                        else %
                            Нет
                    данных{%
                        endif %
                    }</td>
                    <td>{%
                        if attempt.endTime %
                            {{ attempt.endTime|slice:"11:19" }}%
                        else %
                            Нет
                    данных{%
                        endif %
                    }</td>
                    <td>{{ attempt.score }}</td>
                    <td><strong data-grade="{{ attempt.grade }}">{{ attempt.grade }}</strong></td>
                </tr>
            { % empty %
                <tr>
                    <td colspan="4">Нет предыдущих попыток</td>
                </tr>
            { % endfor %
        </tbody>
    </table>
</div>

<div class="actions">
{ % if test.idTest %
    <a href="{% url 'test_material' test.idTest %}">Вернуться к материалам теста</a>
{ % elif test.id %

```

```

<a href="<% url 'test_material' test.id %>">Вернуться к материалам теста</a>
{ % else %
    <a href="<% url 'AllSubjectPage' %>">К списку предметов</a>
{ % endif %

{ % if test.subjectId %
    <a href="<% url 'SubjectPage' test.subjectId %>" class="button">Вернуться к предмету</a>
{ % else %
    <a href="<% url 'AllSubjectPage' %>" class="button">К списку предметов</a>
{ % endif %
</div>
</div>
</body>
</html>
```

## TestStart.html

```

<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    { % load static %
    { % load custom_filters %
    <title>{{ test.nameTest }} - Начало тестирования</title>
    <link rel="stylesheet" href="<% static 'css/TestProcess.css' %>">
</head>
<body>
    { % include 'header.html' %

<div class="container">
    <div class="test-header">
        <h1>{{ test.nameTest }}</h1>
        <p class="test-description">{{ test.descriptionTest }}</p>
    </div>

    <div class="test-start-info card">
        <h2>Информация о teste</h2>
        <div class="info-item">
            <i class="fas fa-clock"></i>
            <span>Время на выполнение: <strong>{{ time_limit }} минут</strong></span>
        </div>
        <div class="info-item">
            <i class="fas fa-question-circle"></i>
```

```

<span>Количество вопросов: <strong>{{ questions.length }}</strong></span>
</div>
<div class="info-item">
    <i class="fas fa-trophy"></i>
    <span>Проходной балл: <strong>{{ test.passingScore }}</strong></span>
</div>

<div class="instructions">
    <h3>Инструкции:</h3>
    <ul>
        <li>После начала теста запустится таймер.</li>
        <li>При истечении времени тест будет автоматически завершен.</li>
        <li>Для вопросов с одним ответом выберите один вариант.</li>
        <li>Для вопросов с несколькими ответами отметьте все правильные варианты.</li>
        <li>В вопросах на сопоставление соедините соответствующие элементы.</li>
        <li>В вопросах на установление порядка перетащите элементы в правильном
порядке.</li>
    </ul>
</div>

<div class="warning">
    <i class="fas fa-exclamation-triangle"></i>
    <p>Внимание! Не закрывайте страницу до завершения теста. После начала тест должен
быть завершен в рамках отведенного времени.</p>
</div>

<form method="GET" action="{% url 'test_process' test.idTest %}">
    <input type="hidden" name="attempt_id" value="{{ attempt_id }}"/>
    <div class="actions">
        <a href="{% url 'test_material' test.idTest %}" class="button secondary">
            <i class="fas fa-arrow-left"></i> Вернуться назад
        </a>
        <button type="submit" class="button primary">
            <i class="fas fa-play"></i> Начать тестирование
        </button>
    </div>
</form>
</div>
</div>
</body>
</html>

```

## TestStudentsResults.html

```

<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    { % load static %}
    <title>Результаты студентов - {{ test.nameTest }}</title>
    <link rel="stylesheet" href="{% static 'css/TestMaterial.css' %}">
    <link rel="stylesheet" href="{% static 'css/TestStudentsResults.css' %}">
</head>
<body>
    { % include 'header.html' % }

    <div class="container">
        <div class="page-header">
            <h1>{{ test.nameTest }} - Результаты студентов</h1>
            <p class="description">{{ test.descriptionTest }}</p>
        </div>

        <div class="statistics-card">
            <h2>Общая статистика</h2>
            <div class="stats-grid">
                <div class="stat-item">
                    <i class="fas fa-users"></i>
                    <span class="stat-value">{{ total_students }}</span>
                    <span class="stat-label">Всего студентов</span>
                </div>
                <div class="stat-item passed">
                    <i class="fas fa-check-circle"></i>
                    <span class="stat-value">{{ passed_students }}</span>
                    <span class="stat-label">Сдали тест</span>
                </div>
                <div class="stat-item failed">
                    <i class="fas fa-times-circle"></i>
                    <span class="stat-value">{{ not_passed_students }}</span>
                    <span class="stat-label">Не сдали тест</span>
                </div>
            </div>
        </div>
    </div>

    <div class="students-results">
        <h2>Результаты по студентам</h2>

```

```

{%- if students %}

{%- for student in students %}

<div class="student-card {%- if student.has_passed %}passed{%- else %}not-passed{%- endif %}">

<div class="student-info">
    <h3>{{ student.secondName }} {{ student.firstName }} {{ student.middleName }}</h3>
    <div class="attempt-info">
        <span class="attempts-count">
            <i class="fas fa-redo"></i> Попыток: {{ student.attempts_count }}
        </span>
        {%- if student.best_attempt %}
            <span class="best-score">
                <i class="fas fa-star"></i> Лучший результат: {{ student.best_attempt.score }} баллов
            </span>
            <span class="grade {%- if student.best_attempt.grade >= 3 %}passed{%- else %}failed{%- endif %}">
                <i class="fas fa-graduation-cap"></i> Оценка: {{ student.best_attempt.grade }}
            </span>
        {%- else %}
            <span class="no-attempts">Нет попыток</span>
        {%- endif %}
    </div>
</div>

{%- if student.all_attempts %}

<div class="attempts-list">
    <h4>История попыток:</h4>
    <table class="attempts-table">
        <thead>
            <tr>
                <th>Попытка</th>
                <th>Дата</th>
                <th>Время</th>
                <th>Оценка</th>
                <th>Баллы</th>
            </tr>
        </thead>
        <tbody>
            {%- for attempt in student.all_attempts %}
                <tr>
                    <td>{{ forloop.counter }}</td>

```

```

<td>{{ attempt.dateTestAttempt|default:"Нет данных" }}</td>
<td>{{ attempt.timeTestAttempt|default:"Нет данных" }}</td>
<td class="grade-cell" % if attempt.grade >= 3 %}passed{ % else
% }failed{ % endif %}>
    {{ attempt.grade }}
</td>
<td>{{ attempt.score }}</td>
</tr>
{ % empty %
<tr>
    <td colspan="5" class="no-data">Нет попыток</td>
</tr>
{ % endfor %

</tbody>
</table>
</div>
{ % endif %

{ % endfor %

{ % else %

<div class="empty-state">
    <i class="fas fa-user-graduate"></i>
    <p>Нет данных о результатах студентов</p>
</div>
{ % endif %

</div>

<div class="actions">
    <a href="{ % url 'test_material' test.idTest % }" class="button back">
        <i class="fas fa-arrow-left"></i> Вернуться к тесту
    </a>
</div>
</div>
</body>
</html>

```

## AdminPanel.html

```

<!-- AdminPanel.html -->
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">

```

```

<title>Административная Панель</title>
{ % load static % }
<link rel="stylesheet" href="{% static 'css/AdminPanel.css' % }">
<link rel="stylesheet" href="{% static 'css/base.css' % }">
<link rel="stylesheet" href="{% static 'css/PracticalWork.css' % }">
<link rel="stylesheet" href="{% static 'css/CreateTest.css' % }">
<link rel="stylesheet" href="{% static 'css/TestEditor.css' % }">
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css">
<link
href="https://fonts.googleapis.com/css2?family=Roboto:wght@300;400;500;700;800&display=swap"
rel="stylesheet">
</head>
<body>

{ % include 'header.html' % }

<div class="container">
<h1>Административная Панель</h1>

<nav class="menu">
<ul>
<li><a href="{% url 'group_crud' %}">Управление Группами</a></li>
<li><a href="{% url 'user_crud' %}">Управление Пользователями</a></li>
<li><a href="{% url 'subject_crud' %}">Управление Предметами</a></li>
</ul>
</nav>
<h2>Выгрузка данных</h2>

<div class="export-actions">
<a href="{% url 'export_groups_to_json' %}" class="export-button">
 Выгрузить группы в JSON
</a>

<form action="{% url 'import_groups_from_json' %}" method="post" enctype="multipart/form-data">
{ % csrf_token % }

<div class="form-group">
<label for="json_file" class="custom-file-upload">
    Выберите файл из которого надо загрузить группы
</label>

```

```

<label for="json_file" class="export-button" >
    <span id="file-name">Ваш файл</span>
    <input type="file" name="json_file" id="json_file" accept=".json" required style="display:
none;" onchange="updateFileName()">
    </label>
</div>

</form>

<button type="submit" class="export-button">
     Загрузить группы из JSON
</button>

</div>

<script>
function updateFileName() {
    const input = document.getElementById('json_file');
    const fileName = input.files.length > 0 ? input.files[0].name : 'Выберите файл';
    document.getElementById('file-name').textContent = fileName;
}
</script>

</div>

```

```

</body>
</html>

AllSubjectPage.html

<!-- AllSubjectPage.html -->
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">

```

```

<title>Все Предметы | YumlSchool</title>
{ % load static %

<link rel="stylesheet" href="{% static 'css/base.css' %}">
<link rel="stylesheet" href="{% static 'css/AllSubjectPage.css' %}">
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css">
<style>

/* Принудительное переопределение стилей */
html, body {
    background-color: #1a1225 !important;
    color: #e9e8ee !important;
}

.container {
    background-color: transparent !important;
    box-shadow: none !important;
}

.main-header {
    background-color: #1a1225 !important;
    border-bottom: 1px solid rgba(255, 255, 255, 0.1);
}

.main-header .container {
    margin: 0 !important;
    max-width: 100% !important;
}

</style>
</head>
<body>
{ % include 'header.html' % }

<section class="hero-section">
    <div class="container">
        <h1>Мои предметы</h1>
        <p>Изучайте материалы, выполняйте задания и проходите тесты для успешного освоения программы обучения</p>
    </div>
</section>

<div class="subjects-container">
{ % if subjects % }
    <div class="subject-list">

```

```

{ % for subject in subjects % }
    <a href="{% url 'SubjectPage' subject.idSubject %}" class="subject-card-link">
        <div class="subject-card">
            <div class="card-content">
                { % if request.session.role_id == 2 % }
                    <div class="subject-status">Преподаватель</div>
                { % endif %}
                <div class="card-header">
                    <div class="subject-icon">
                        <i class="fas fa-book"></i>
                    </div>
                    <h3 class="subject-title">{{ subject.nameSubject }}</h3>
                </div>
                <div class="card-body">
                    <div class="subject-info">
                        <div class="info-item">
                            <i class="fas fa-users"></i>
                            <span>
                                { % for group in groups % }
                                    { % if group.idGroup == subject.idGroup % }
                                        {{ group.nameGroup }}
                                    { % endif %}
                                { % endfor %}
                            </span>
                        </div>
                        <div class="info-item">
                            <i class="fas fa-chalkboard-teacher"></i>
                            <span>
                                { % for user in users % }
                                    { % if user.idUser == subject.userId % }
                                        {{ user.firstName }} {{ user.secondName }} {{ user.middleName }}
                                    { % endif %}
                                { % endfor %}
                            </span>
                        </div>
                    </div>
                    <div class="subject-actions">
                        <span class="view-subject">Перейти к предмету</span>
                    </div>
                </div>
            </div>
        </div>
    </a>

```

```

        </a>
        {% endfor %}
    </div>
    {% else %}
        <div class="no-subjects">
            <i class="fas fa-search"></i>
            <h2>Нет доступных предметов</h2>
            <p>В данный момент у вас нет доступных предметов для изучения.</p>
        </div>
        {% endif %}
    </div>
</body>
</html>

```

## header.html

```

<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    {% load static %}
    <link rel="stylesheet" href="{% static 'css/base.css' %}">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css">
    <link href="https://fonts.googleapis.com/css2?family=Roboto:wght@300;400;500;700&display=swap"
rel="stylesheet">
    <title>YumSchool</title>
    <style>
        .header {
            background-color: var(--primary);
            color: white;
            padding: 1rem 2rem;
            display: flex;
            align-items: center;
            justify-content: space-between;
            box-shadow: 0 2px 8px rgba(0, 0, 0, 0.1);
            position: sticky;
            top: 0;
            z-index: 1000;
        }
        .header-left {
            display: flex;
            align-items: center;

```

```
}
```

```
.logo {  
    font-size: 1.5rem;  
    font-weight: 700;  
    letter-spacing: 0.5px;  
    color: white;  
    text-decoration: none;  
    display: flex;  
    align-items: center;  
}
```

```
.logo i {  
    margin-right: 8px;  
    font-size: 1.8rem;  
}
```

```
.nav-links {  
    display: flex;  
    margin-left: 2rem;  
}
```

```
.nav-links a {  
    color: rgba(255, 255, 255, 0.85);  
    text-decoration: none;  
    padding: 0.5rem 1rem;  
    border-radius: 4px;  
    transition: all 0.2s;  
    font-weight: 500;  
}
```

```
.nav-links a:hover, .nav-links a.active {  
    color: white;  
    background-color: rgba(255, 255, 255, 0.1);  
}
```

```
.user-info {  
    display: flex;  
    align-items: center;  
    gap: 1rem;  
    position: relative;  
    z-index: 1001;
```

```
}

.user-avatar {
    width: 36px;
    height: 36px;
    border-radius: 50%;
    background-color: var(--primary-light);
    display: flex;
    align-items: center;
    justify-content: center;
    font-weight: 600;
    font-size: 1rem;
    color: var(--primary-dark);
}

.user-name {
    font-weight: 500;
    display: flex;
    align-items: center;
}

.user-role {
    font-size: 0.8rem;
    opacity: 0.8;
    display: block;
}

.logout-button, .auth-button {
    background-color: rgba(255, 255, 255, 0.15);
    color: white;
    border: none;
    border-radius: 4px;
    padding: 0.5rem 1rem;
    font-size: 0.9rem;
    cursor: pointer;
    transition: all 0.2s;
}

.logout-button:hover, .auth-button:hover {
    background-color: rgba(255, 255, 255, 0.25);
    transform: translateY(-2px);
}
```

```
.auth-buttons {  
    display: flex;  
    gap: 0.5rem;  
}  
  
.menu-toggle {  
    display: none;  
    background: none;  
    border: none;  
    color: white;  
    font-size: 1.5rem;  
    cursor: pointer;  
}  
  
.logout-form {  
    position: relative;  
    z-index: 1002;  
    margin: 0;  
}  
  
.logout-btn {  
    display: inline-flex;  
    align-items: center;  
    gap: 0.5rem;  
    padding: 0.5rem 1rem;  
    background-color: rgba(255, 255, 255, 0.1);  
    border: 1px solid rgba(255, 255, 255, 0.2);  
    border-radius: 4px;  
    color: white !important;  
    font-size: 0.9rem;  
    cursor: pointer;  
    transition: all 0.2s;  
    position: relative;  
    z-index: 1003;  
}  
  
.logout-btn:hover {  
    background-color: rgba(255, 255, 255, 0.2);  
    transform: translateY(-2px);  
}
```

```
.main-header .container {  
    position: relative;  
    z-index: 1000;  
}  
  
@media (max-width: 768px) {  
    .header {  
        padding: 1rem;  
    }  
  
    .nav-links {  
        display: none;  
        position: absolute;  
        top: 100%;  
        left: 0;  
        right: 0;  
        background-color: var(--primary);  
        flex-direction: column;  
        padding: 1rem;  
        box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);  
    }  
  
    .nav-links.active {  
        display: flex;  
    }  
  
    .nav-links a {  
        padding: 0.75rem 1rem;  
    }  
  
    .menu-toggle {  
        display: block;  
    }  
  
.user-name span {  
    display: none;  
}  
}  
}</style>  
</head>  
<body>  
    <header class="main-header">
```

```

<div class="container">
    <div class="logo">
        <i class="fas fa-graduation-cap"></i>
        <span>YumlSchool</span>
    </div>

    <div class="user-info">
        <div class="user-details">
            <span class="user-name">{{ request.session.full_name }}</span>
            {% if request.session.role_id == 1 %}
                <span class="user-role">Администратор</span>
            {% elif request.session.role_id == 2 %}
                <span class="user-role">Преподаватель</span>
            {% elif request.session.role_id == 3 %}
                <span class="user-role">Студент</span>
            {% elif request.session.role_id == 5 %}
                <span class="user-role">Новый пользователь</span>
            {% endif %}
        </div>

        <form method="POST" action="{% url 'logout' %}" class="logout-form">
            {% csrf_token %}
            <button type="submit" class="logout-btn">
                <i class="fas fa-sign-out-alt"></i>
                <span>Выход</span>
            </button>
        </form>
    </div>
</div>
</header>

<script>
    document.getElementById('menuToggle').addEventListener('click', function() {
        document.getElementById('navLinks').classList.toggle('active');
    });
</script>
</body>
</html>

```

**login.html**

```

<!-- templates/login.html -->
<!DOCTYPE html>
<html lang="ru">

```

```

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Вход в систему | YumlSchool</title>
    {% load static %}

    <link rel="stylesheet" href="{% static 'css/logreg.css' %}">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css">
    <link href="https://fonts.googleapis.com/css2?family=Roboto:wght@300;400;500;700&display=swap"
rel="stylesheet">

</head>
<body>

    <div class="auth-container">
        <div class="header">
            <div class="logo">
                <i class="fas fa-graduation-cap"></i>
                YumlSchool
            </div>
            <p class="subtitle">Войдите в свой аккаунт</p>
        </div>

        {% if error %}
            <div class="error-message">
                {{ error }}
            </div>
        {% endif %}

        <form class="login-form" method="POST" action="{% url 'login' %}">
            {% csrf_token %}

            <div class="form-group">
                <label for="login">Логин</label>
                <div class="input-group">
                    <i class="fas fa-user"></i>
                    <input type="text" id="login" name="login" placeholder="Введите ваш логин" value="{{ login }}" required>
                </div>
            </div>

            <div class="form-group">
                <label for="password">Пароль</label>
                <div class="input-group">
                    <i class="fas fa-lock"></i>

```

```

<input type="password" id="password" name="password" placeholder="Ведите ваш
пароль" required>
    </div>
</div>

<div class="button-container">
    <button type="submit" class="login-button">
        <i class="fas fa-sign-in-alt"></i> Войти
    </button>
</div>
</form>

<div class="divider">или</div>

<div class="auth-links">
    <span>Еще нет аккаунта?</span>
    <a href="{% url 'register' %}" class="register-link">Зарегистрируйтесь</a>
</div>
</div>
</body>
</html>

```

## NullSubjects.html

```

<!-- NullSubjects.html -->
<!DOCTYPE html>
<html lang="ru">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>Нет предметов | YumlSchool</title>
        {% load static %}
        <link rel="stylesheet" href="{% static 'css/base.css' %}">
        <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css">
        <link href="https://fonts.googleapis.com/css2?family=Roboto:wght@300;400;500;700&display=swap"
            rel="stylesheet">
        <style>
            :root {
                --dark-purple: #1a1225;
                --medium-purple: #7743DB;
                --light-purple: #B39DDB;
                --accent-purple: #9D4EDD;
            }

```

```
body {  
    background-color: var(--dark-purple);  
    min-height: 100vh;  
    margin: 0;  
    padding: 0;  
    display: flex;  
    flex-direction: column;  
}  
  
.content-container {  
    flex: 1;  
    display: flex;  
    flex-direction: column;  
    align-items: center;  
    justify-content: center;  
    padding: 2rem;  
    text-align: center;  
    margin-top: 2rem;  
}  
  
.message-box {  
    background: linear-gradient(135deg, #272134 0%, #3a1d69 100%);  
    padding: 2.5rem;  
    border-radius: 12px;  
    box-shadow: 0 10px 25px rgba(0, 0, 0, 0.5);  
    max-width: 600px;  
    width: 90%;  
    margin: 2rem auto;  
    border: 1px solid rgba(255, 255, 255, 0.1);  
}  
  
.icon-container {  
    margin-bottom: 1.5rem;  
}  
  
.icon-container i {  
    font-size: 4rem;  
    color: var(--accent-purple);  
}  
  
h1 {  
    color: var(--text-light);
```

```
        font-size: 1.8rem;
        margin-bottom: 1.5rem;
        line-height: 1.4;
    }

.user-info {
    color: var(--text-secondary);
    background-color: rgba(255, 255, 255, 0.1);
    padding: 0.75rem 1.5rem;
    border-radius: 8px;
    margin-bottom: 1.5rem;
    font-size: 0.95rem;
}

.back-button {
    display: inline-flex;
    align-items: center;
    gap: 0.5rem;
    padding: 0.75rem 1.5rem;
    background-color: var(--medium-purple);
    color: white;
    text-decoration: none;
    border-radius: 8px;
    font-weight: 500;
    transition: all 0.3s ease;
    border: none;
    cursor: pointer;
    margin-top: 1.5rem;
}

.back-button:hover {
    background-color: var(--accent-purple);
    transform: translateY(-2px);
    box-shadow: 0 4px 12px rgba(0, 0, 0, 0.3);
}

@media (max-width: 768px) {
    .message-box {
        padding: 1.5rem;
    }
}

h1 {
```

```

        font-size: 1.4rem;
    }

.icon-container i {
    font-size: 3rem;
}
}

</style>
</head>
<body>
{ % include 'header.html' % }

<div class="content-container">
<div class="message-box">
<div class="icon-container">
<i class="fas fa-book-reader"></i>
</div>

<div class="user-info">
<i class="fas fa-user"></i> {{ request.session.full_name }}
{ % if request.session.role_id == 1 %}
<span>(Администратор)</span>
{ % elif request.session.role_id == 2 %}
<span>(Преподаватель)</span>
{ % elif request.session.role_id == 3 %}
<span>(Студент)</span>
{ % elif request.session.role_id == 4 %}
<span>(Новый пользователь)</span>
{ % elif request.session.role_id == 6 %}
<span>(Заблокированный пользователь)</span>
{ % endif %}
</div>

{ % if request.session.role_id == 4 %

<h1>У вас пока нет доступных предметов</h1>
<p style="color: var(--text-secondary); margin-bottom: 2rem;">
    Для получения доступа к предметам необходимо быть зачисленным в группу.
    Пожалуйста, обратитесь к администратору.
</p>

```

```

{%- elif request.session.role_id == 6 %}

<h1>У вас нет доступных предметов</h1>
<p style="color: var(--text-secondary); margin-bottom: 2rem;">
    Доступ заблокирован.
    Пожалуйста, обратитесь к администратору.
</p>

{%- endif %}

</div>
</div>
</body>
</html>

registration.html

<!-- registration.html -->
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Регистрация | YumlSchool</title>
    {%- load static %}
    <link rel="stylesheet" href="{% static 'css/logreg.css' %}">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css">
    <link href="https://fonts.googleapis.com/css2?family=Roboto:wght@300;400;500;700&display=swap" rel="stylesheet">
</head>
<body>
    <div class="auth-container">
        <div class="header">
            <div class="logo">
                <i class="fas fa-graduation-cap"></i>
                YumlSchool
            </div>
            <p class="subtitle">Создайте свой аккаунт</p>
        </div>
        {%- if error %}
        <div class="error-message">
            {{ error }}
        </div>

```

```

{ % endif % }

<form method="POST" action="{% url 'register' %}>
    {% csrf_token %}
    <div class="form-group">
        <label for="firstName">Фамилия</label>
        <div class="input-group">
            <i class="fas fa-user"></i>
            <input type="text" id="firstName" name="firstName" placeholder="Введите фамилию"
required value="{{ first_name|default:'' }}>
            </div>
        </div>

        <div class="form-group">
            <label for="secondName">Имя</label>
            <div class="input-group">
                <i class="fas fa-user"></i>
                <input type="text" id="secondName" name="secondName" placeholder="Введите имя"
required value="{{ second_name|default:'' }}>
                </div>
            </div>

            <div class="form-group">
                <label for="middleName">Отчество</label>
                <div class="input-group">
                    <i class="fas fa-user"></i>
                    <input type="text" id="middleName" name="middleName" placeholder="Введите отчество"
value="{{ middle_name|default:'' }}>
                    </div>
                </div>

                <div class="form-group">
                    <label for="email">Email</label>
                    <div class="input-group">
                        <i class="fas fa-envelope"></i>
                        <input type="email" id="email" name="email" placeholder="Введите email" required
value="{{ email|default:'' }}>
                        </div>
                    </div>

                    <div class="form-group">
                        <label for="login">Логин</label>

```

```

<div class="input-group">
    <i class="fas fa-user-circle"></i>
    <input type="text" id="login" name="login" placeholder="Придумайте логин" required
value="{{ login|default:'{}' }}">
    </div>
</div>

<div class="form-group">
    <label for="password">Пароль</label>
    <div class="input-group">
        <i class="fas fa-lock"></i>
        <input type="password" id="password" name="password" placeholder="Придумайте
пароль" required>
        </div>
    </div>

    <button type="submit">
        <i class="fas fa-user-plus"></i>
        Зарегистрироваться
    </button>
</form>

<div class="divider">или</div>

<div class="auth-links">
    Уже есть аккаунт? <a href="{% url 'login' %}">Войдите</a>
</div>
</div>
</body>
</html>

```

## SubjectPage.html

```

<!DOCTYPE html>
<html>

    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>{{ subject.nameSubject }} | YumSchool</title>
        {% load static %}
        <link rel="stylesheet" href="{% static 'css/base.css' %}">
        <link rel="stylesheet" href="{% static 'css/SubjectPage.css' %}">
        <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css">
    </head>

```

```

<link href="https://fonts.googleapis.com/css2?family=Roboto:wght@300;400;500;700&display=swap"
rel="stylesheet">

<style>
/* Принудительное переопределение стилей */

html,
body {
    background-color: #1a1225 !important;
    color: #e9e8ee !important;
}

.container {
    background-color: transparent !important;
    box-shadow: none !important;
}

.main-header {
    background-color: #1a1225 !important;
    border-bottom: 1px solid rgba(255, 255, 255, 0.1);
}

.main-header .container {
    margin: 0 !important;
    max-width: 100% !important;
}

</style>
</head>

<body>
{%
    include 'header.html'
%}
<div class="container">
    <div class="button-group">
        <a href="{% url 'AllSubjectPage' %}" class="back-button">
            <i class="fas fa-arrow-left"></i> Назад к предметам
        </a>
    </div>
</div>

<div class="container">
    <h1>{{ subject.nameSubject }}</h1>

    <div class="subject-info">
        <p><strong>Группа:</strong> {{ group.nameGroup }}</p>

```

```

<p><strong>Преподаватель:</strong> {{ teacher.firstName }} {{ teacher.secondName }} {{ teacher.middleName }}</p>
</div>

<div class="content-section">
    <div class="section-header">
        <h2>Студенты группы</h2>
    </div>
    {% if students %}
        <ul class="content-list">
            {% for student in students %}
                <li>
                    {{ student.firstName }} {{ student.secondName }} {{ student.middleName }}
                </li>
            {% endfor %}
        </ul>
    {% else %}
        <p>Студенты отсутствуют</p>
    {% endif %}
</div>

<div class="content-section">
    <div class="section-header">
        <h2>Лекции</h2>
    {% if role_id == 2 or role_id == 1 %}
        <a href="{% url 'create_lection' subject.idSubject %}" class="add-content-btn">
            <i class="fas fa-plus"></i> Добавить лекцию
        </a>
    {% endif %}
    </div>
    {% if lections %}
        <ul class="content-list">
            {% for lection in lections %}
                <li>
                    <a href="{% url 'LectionMaterial' lection.idLectionMaterial %}">
                        <i class="fas fa-book-open"></i> {{ lection.nameLectionMaterial }}
                    </a>
                </li>
            {% endfor %}
        </ul>
    {% else %}
        <p>Лекции отсутствуют</p>
    
```

```

    { % endif % }

</div>

<div class="content-section">
    <div class="section-header">
        <h2>Практические работы</h2>
        { % if role_id == 2 or role_id == 1 %}
        <a href="{% url 'create_practical_work' subject.idSubject %}" class="add-content-btn">
            <i class="fas fa-plus"></i> Добавить практическую работу
        </a>
        { % endif % }
    </div>
    { % if practical_works %}
    <ul class="content-list">
        { % for work in practical_works %}
        <li>
            <a href="{% url 'PracticalWorkMaterial' work.idPracticalWork %}">
                <i class="fas fa-laptop-code"></i> {{ work.namePracticalWork }}
            </a>
        </li>
        { % endfor %}
    </ul>
    { % else %}
        <p>Практические работы отсутствуют</p>
    { % endif %}
    </div>

<div class="content-section">
    <div class="section-header">
        <h2>Тесты</h2>
        { % if role_id == 2 or role_id == 1 %}
        <a href="{% url 'create_test' subject.idSubject %}" class="add-content-btn">
            <i class="fas fa-plus"></i> Добавить тест
        </a>
        { % endif % }
    </div>
    { % if tests %}
    <ul class="content-list">
        { % for test in tests %}
        <li>
            <a href="{% url 'test_material' test.idTest %}">
                <i class="fas fa-tasks"></i> {{ test.nameTest }}
            </a>
        </li>
    { % endfor %}
    </ul>
    { % endif %}
</div>

```

```
        </a>
    </li>
    { % endfor %
</ul>
{ % else %
<p>Тесты отсутствуют</p>
{ % endif %
</div>
</div>
</div>
</body>

</html>
```

## Views.py

```
import datetime
from venv import logger
from django.shortcuts import redirect, render
from django.http import HttpResponse, JsonResponse, HttpResponseRedirect
import requests
from datetime import datetime
import datetime
import logging
import json
from django.contrib import messages
from django.template.loader import get_template
import base64
import bcrypt
import re

import urllib3
urllib3.disable_warnings(urllib3.exceptions.InsecureRequestWarning)

from django.shortcuts import render, redirect
from datetime import datetime, date
from django.utils import timezone
import csv
from django.http import HttpResponse
import re
import json
from django.contrib import messages
```

```
from django.http import HttpResponseRedirect  
from reportlab.lib.pagesizes import letter  
from reportlab.lib import colors  
from reportlab.lib.units import inch  
from reportlab.pdfgen import canvas
```

```
from reportlab.lib.pagesizes import letter  
from reportlab.pdfgen import canvas  
from django.http import HttpResponseRedirect
```

```
API_BASE_URL = 'https://172.20.10.3:7154/api'
```

```
from reportlab.lib.pagesizes import letter  
from reportlab.lib import colors  
from reportlab.pdfgen import canvas  
from django.http import HttpResponseRedirect
```

```
from django.http import HttpResponseRedirect  
from docx import Document  
from docx.shared import Pt
```

```
from django.http import HttpResponseRedirect  
from docx import Document  
from docx.shared import Pt  
from docx.enum.text import WD_ALIGN_PARAGRAPH  
from datetime import datetime  
import os  
from django.http import HttpResponseRedirect  
from django.shortcuts import render  
from django.core.files.storage import FileSystemStorage
```

```
import json  
import os  
from django.http import HttpResponseRedirect  
from django.core.files.storage import FileSystemStorage  
from django.contrib import messages
```

```

# Вспомогательная функция для выполнения API-запросов
def make_api_request(endpoint, method='get', data=None, params=None):
    # Убираем ведущий слеш, если он присутствует
    if endpoint.startswith('/'):
        endpoint = endpoint[1:]

    # Здесь должно быть API_BASE_URL без добавления "api/" в начале URL,
    # так как эта часть уже содержится в endpoint
    url = f'{API_BASE_URL}/{endpoint}'

    print(f"API Request: {method.upper()} {url}")

    try:
        response = requests.request(
            method,
            url,
            json=data if method.lower() in ['post', 'put'] else None,
            data=data if method.lower() not in ['post', 'put'] else None,
            params=params,
            verify=False
        )
        print(f"API Response: {response.status_code}")

        # Возвращаем словарь с данными вместо объекта ответа
        result = {
            'status_code': response.status_code,
            'data': response.json() if response.status_code == 200 else None
        }
        return result
    except requests.exceptions.RequestException as e:
        print(f"Ошибка при запросе к API: {e}")
        return {
            'status_code': 500,
            'data': None,
            'error': str(e)
        }

def export_groups_to_json(request):
    # Получаем данные из таблицы Group
    groups_response = make_api_request('Groups')

```

```

if groups_response['status_code'] != 200:
    return HttpResponse("Ошибка при получении данных групп.", status=500)

groups = groups_response['data']

# Формируем данные в нужном формате
formatted_groups = [{"nameGroup": group['nameGroup']} for group in groups]

# Создаем JSON файл
json_file_path = 'exported_groups.json'
with open(json_file_path, 'w', encoding='utf-8') as file:
    json.dump(formatted_groups, file, ensure_ascii=False, indent=4)

# Отправляем файл пользователю
response = HttpResponse(open(json_file_path, 'rb'), content_type='application/json')
response['Content-Disposition'] = f'attachment; filename="{os.path.basename(json_file_path)}"'

return response

```

```

def import_groups_from_json(request):
    if request.method == 'POST':
        json_file = request.FILES.get('json_file') # Используем get вместо прямого доступа
        if json_file:
            fs = FileSystemStorage()
            filename = fs.save(json_file.name, json_file)

            # Читаем JSON файл и загружаем данные
            file_path = fs.path(filename) # Получаем путь к файлу
            with open(file_path, 'r', encoding='utf-8') as file: # Используем file_path
                groups_data = json.load(file)

            # Отправляем данные на API
            api_url = f'{API_BASE_URL}/Groups'

            for group in groups_data:
                # Убедитесь, что данные имеют правильный формат
                if 'nameGroup' in group:
                    response = requests.post(api_url, json={'nameGroup': group['nameGroup']}, verify=False) #
                    Игнорируем проверку сертификата

```

```

        if response.status_code not in [201, 200]: # Проверяем, что запрос успешен
            messages.error(request, f'Ошибка при загрузке группы: {group["nameGroup"]}')
            return redirect('group_crud')

        else:
            messages.error(request, 'Некорректный формат данных в JSON файле.')
            return redirect('group_crud')

        messages.success(request, 'Данные успешно загружены из JSON файла.')
        return redirect('group_crud')

    else:
        messages.error(request, 'Файл не загружен. Пожалуйста, выберите файл.')
        return redirect('group_crud')

return render(request, 'AdminPanel.html')

```

```

def export_students_performance(request, practical_work_id):
    practical_work_url = f'{API_BASE_URL}/PracticalWorks/{practical_work_id}'
    response_from_api = requests.get(practical_work_url, verify=False)

    if response_from_api.status_code != 200:
        return HttpResponseRedirect("Практическая работа не найдена.", status=404)

    practical_work = response_from_api.json()

    # Получаем информацию о предмете
    subject_url = f'{API_BASE_URL}/Subjects/{practical_work["subjectId"]}'
    response_subject = requests.get(subject_url, verify=False)
    subject = response_subject.json() if response_subject.status_code == 200 else {}
    subject_name = subject.get('nameSubject', 'Неизвестно')
    group_id = subject.get('idGroup', None)

    # Получаем информацию о группе
    group_name = 'Не указано'
    if group_id:
        group_url = f'{API_BASE_URL}/Groups/{group_id}'
        response_group = requests.get(group_url, verify=False)
        group = response_group.json() if response_group.status_code == 200 else {}
        group_name = group.get('nameGroup', 'Не указано')

```

```

completed_works_url = f'{API_BASE_URL}/PracticalWorks/{practical_work_id}/completed'
response_completed_works = requests.get(completed_works_url, verify=False)

if response_completed_works.status_code != 200:
    return HttpResponse("Нет сданных работ.", status=404)

completed_works = response_completed_works.json()

# Create the Word document
document = Document()

# Title page
document.add_heading('YumlSchool', 0)
document.add_paragraph(f'{practical_work["namePracticalWork"]}')
document.add_paragraph(f'Предмет: {subject_name}')
document.add_paragraph(f'Группа: {group_name}')
document.paragraphs[-1].alignment = WD_ALIGN_PARAGRAPH.CENTER

# List of students and their grades
document.add_heading('Список студентов и их оценки', level=1)
table = document.add_table(rows=1, cols=2)
table.style = 'Table Grid'
hdr_cells = table.rows[0].cells
hdr_cells[0].text = 'ФИО'
hdr_cells[1].text = 'Оценка'

student_works = {}
for work in completed_works:
    user_id = work['userId']
    user_url = f'{API_BASE_URL}/Users/{user_id}'
    response_user = requests.get(user_url, verify=False)
    user = response_user.json() if response_user.status_code == 200 else {}

    full_name = f'{user.get("firstName", "Не указано")} {user.get("secondName", "Не указано")}' \
                f'{user.get("middleName", "Не указано")}'
    if user_id not in student_works:
        student_works[user_id] = {
            'full_name': full_name,
            'latest_grade': None
        }

# Находим последнюю попытку студента

```

```

if 'attemptCount' in work:
    attempt_count = int(work['attemptCount'])
    if attempt_count > (student_works[user_id]['latest_grade'].get('attemptCount', 0) if student_works[user_id]['latest_grade'] else 0):
        student_works[user_id]['latest_grade'] = work

for student_id, student_info in student_works.items():
    row_cells = table.add_row().cells
    row_cells[0].text = student_info['full_name']
    if student_info['latest_grade']:
        row_cells[1].text = str(student_info['latest_grade']['grade'])
    else:
        row_cells[1].text = 'Не сдано'

    # Signature of the teacher
    document.add_paragraph()
    row = document.add_paragraph()
    row.alignment = WD_ALIGN_PARAGRAPH.LEFT
    p = row.add_run('Подпись преподавателя _____')
    p = row.add_run('\n')
    p = row.add_run(f'{datetime.now().day} {datetime.now().strftime("%B")} {datetime.now().year} года')

    row = document.add_paragraph()
    row.alignment = WD_ALIGN_PARAGRAPH.RIGHT
    teacher_url = f'{API_BASE_URL}/Users/{practical_work["userId"]}'
    response_teacher = requests.get(teacher_url, verify=False)
    teacher = response_teacher.json() if response_teacher.status_code == 200 else {}
    full_teacher_name = f'{teacher.get("firstName", "Не указано")} {teacher.get("secondName", "Не указано")}' \
        {teacher.get("middleName", "Не указано")}'
    p = row.add_run(f'Проверил преподаватель {full_teacher_name}')

    # Save the document
    response = HttpResponse(
        content_type='application/vnd.openxmlformats-officedocument.wordprocessingml.document'
    )
    response['Content-Disposition'] = 'attachment; filename="Успеваемость студентов.docx"' # Изменено
название файла
    document.save(response)
    return response

```

```

def index(request):
    return redirect('login')

def adminPage(request):
    return render(request, 'AdminPanel.html')

def LectionCRUD(request):
    return render(request, 'SubjectInfo/LectionCRUD.html')

def LectionMaterial(request, id):
    # Проверяем авторизацию
    if 'user_id' not in request.session:
        request.session['next_url'] = request.path
        return redirect('login')

    lection_url = f'{API_BASE_URL}/LectionMaterials/{id}'
    response_from_api_lection = requests.get(lection_url, verify=False)
    lection = response_from_api_lection.json() if response_from_api_lection.status_code == 200 else {}

    # Получаем информацию о преподавателе
    teacher_url = f'{API_BASE_URL}/Users/{lection["userId"]}'
    response_from_api_teacher = requests.get(teacher_url, verify=False)
    teacher = response_from_api_teacher.json() if response_from_api_teacher.status_code == 200 else {}

    # Получаем роль пользователя из сессии
    role_id = request.session.get('role_id')
    user_id = request.session.get('user_id')

    return render(request, 'SubjectInfo/LectionMaterial.html', {
        'lection': lection,

```

```

'teacher': teacher,
'role_id': role_id, # Добавляем role_id в контекст
'user_id': user_id # Добавляем user_id в контекст
})

def SubjectPage(request, id):
    # Проверяем, авторизован ли пользователь
    if 'user_id' not in request.session:
        request.session['next_url'] = request.path
        return redirect('login')

    try:
        # Получаем информацию о предмете
        subject_url = f'{API_BASE_URL}/Subjects/{id}'
        subject_response = requests.get(subject_url, verify=False)

        if subject_response.status_code != 200:
            return redirect('AllSubjectPage')

        subject = subject_response.json()

        # Получаем информацию о группе
        group_id = subject.get('idGroup')
        group_url = f'{API_BASE_URL}/Groups/{group_id}'
        group_response = requests.get(group_url, verify=False)
        group = group_response.json() if group_response.status_code == 200 else None

        # Получаем студентов из группы
        students_url = f'{API_BASE_URL}/Users'
        students_response = requests.get(students_url, verify=False)
        all_users = students_response.json() if students_response.status_code == 200 else []
        students = [user for user in all_users if user.get('idGroup') == group_id and user.get('idRole') == 3] #

Студенты (role_id = 3)

        # Получаем лекции, практические работы и тесты
        lections_url = f'{API_BASE_URL}/LectureMaterials'
        lections_response = requests.get(lections_url, verify=False)
        lections = [lection for lection in lections_response.json() if lection.get('subjectId') == id] if
lections_response.status_code == 200 else []

```

```

practical_works_url = f'{API_BASE_URL}/PracticalWorks'
practical_works_response = requests.get(practical_works_url, verify=False)
practical_works = [pw for pw in practical_works_response.json() if pw.get('subjectId') == id] if
practical_works_response.status_code == 200 else []

tests_url = f'{API_BASE_URL}/Tests'
tests_response = requests.get(tests_url, verify=False)
tests = [test for test in tests_response.json() if test.get('subjectId') == id] if tests_response.status_code
== 200 else []

# Получаем информацию о преподавателе
teacher_id = subject.get('userId')
teacher_url = f'{API_BASE_URL}/Users/{teacher_id}'
teacher_response = requests.get(teacher_url, verify=False)
teacher = teacher_response.json() if teacher_response.status_code == 200 else None

context = {
    'subject': subject,
    'group': group,
    'students': students, # Добавляем студентов в контекст
    'lections': lections,
    'practical_works': practical_works,
    'tests': tests,
    'teacher': teacher, # Добавляем учителя в контекст
    'role_id': request.session.get('role_id')
}

return render(request, 'SubjectPage.html', context)

except Exception as e:
    print(f"Error in SubjectPage: {e}")
    return redirect('AllSubjectPage')

def create_lection(request, subject_id):
    if request.method == 'POST':
        name_lection_material = request.POST.get('nameLectionMaterial')

```

```

description_lection_material = request.POST.get('descriptionLectionMaterial')
name_file_lection = request.POST.get('nameFileLection')
url_file_lection = request.POST.get('urlFileLection')
user_id = request.session.get('user_id')

# Получаем текущую дату и время
date_upload = date.today()
time_upload = datetime.now().strftime('%H:%M:%S')

# Создаем лекцию
api_url = f'{API_BASE_URL}/LectionMaterials'
data = {
    'nameLectionMaterial': name_lection_material,
    'descriptionLectionMaterial': description_lection_material,
    'dateUploadLectionMaterial': str(date_upload),
    'timeUploadLectionMaterial': str(time_upload),
    'nameFileLection': name_file_lection,
    'urlFileLection': url_file_lection,
    'userId': user_id,
    'subjectId': subject_id # Связываем лекцию с предметом
}

try:
    response = requests.post(api_url, json=data, verify=False)
    response.raise_for_status()
    return redirect('SubjectPage', id=subject_id)
except requests.exceptions.RequestException as e:
    return render(request, 'SubjectInfo/LectionCRUD.html', {
        'error': 'Ошибка при добавлении лекции: ' + str(e),
    })

return render(request, 'SubjectInfo/LectionCRUD.html')

```

```

def edit_lection(request, id):
    lection_url = f'{API_BASE_URL}/LectionMaterials/{id}'

    # Получаем текущую лекцию
    try:
        response_from_api_lection = requests.get(lection_url, verify=False)
        response_from_api_lection.raise_for_status()
        lection = response_from_api_lection.json()

```

```

except requests.exceptions.RequestException as e:
    return render(request, 'SubjectInfo/LectureEdit.html', {
        'error': 'Ошибка при получении лекции: ' + str(e),
    })

if request.method == 'POST':
    updated_data = {
        'IdLectureMaterial': id,
        'nameLectureMaterial': request.POST.get('nameLectureMaterial', lection['nameLectureMaterial']),
        'descriptionLectureMaterial': request.POST.get('descriptionLectureMaterial',
            lection['descriptionLectureMaterial']),
        'dateUploadLectureMaterial': lection['dateUploadLectureMaterial'],
        'timeUploadLectureMaterial': lection['timeUploadLectureMaterial'],
        'nameFileLecture': request.POST.get('nameFileLecture', lection['nameFileLecture']),
        'urlFileLecture': request.POST.get('urlFileLecture', lection['urlFileLecture']),
        'userId': lection['userId'],
        'subjectId': lection['subjectId'],
    }
    print("Отправляемые данные для обновления:", updated_data)

try:
    response = requests.put(lection_url, json=updated_data, verify=False)
    response.raise_for_status()
    print("Ответ от API:", response.status_code, response.text)
    # Изменяем редирект на страницу лекции
    return redirect('LectureMaterial', id=id) # Перенаправляем на страницу лекции
except requests.exceptions.RequestException as e:
    return render(request, 'SubjectInfo/LectureEdit.html', {
        'error': 'Ошибка при обновлении лекции: ' + str(e),
        'lection': lection,
    })

return render(request, 'SubjectInfo/LectureEdit.html', {'lection': lection})

def delete_lection(request, id):
    # Получаем информацию о лекции для получения subjectId
    lection_url = f'{API_BASE_URL}/LectureMaterials/{id}'
    response_from_api_lection = requests.get(lection_url, verify=False)
    lection = response_from_api_lection.json() if response_from_api_lection.status_code == 200 else {}

```

```

if request.method == 'POST':
    response = requests.delete(lection_url, verify=False)
    if response.status_code == 204:
        return redirect('SubjectPage', id=lection['subjectId'])
    return HttpResponse(status=405)

def AllSubjectPage(request):
    # Проверяем, авторизован ли пользователь
    if 'user_id' not in request.session:
        # Сохраняем URL, на который пользователь пытался перейти
        request.session['next_url'] = request.path
        return redirect('login')

    try:
        # Получаем информацию о пользователе
        user_id = request.session.get('user_id')
        role_id = request.session.get('role_id')

        # Получаем список всех предметов
        subjects_response = make_api_request('Subjects')

        # Проверяем статус код в словаре, а не как атрибут объекта
        if subjects_response['status_code'] != 200:
            return redirect('nullSubjects')

        # Получаем данные из словаря, а не вызывая метод json()
        all_subjects = subjects_response['data']

        # Фильтруем предметы в зависимости от роли пользователя
        if role_id == 1: # Администратор
            subjects = all_subjects # Администратору доступны все предметы
        elif role_id == 2: # Преподаватель
            # Фильтруем предметы, где userId совпадает с ID пользователя
            subjects = [subject for subject in all_subjects if subject.get('userId') == user_id]
        elif role_id == 3: # Студент
            # Получаем информацию о группе пользователя
            user_response = make_api_request(f'Users/{user_id}')
            if user_response['status_code'] != 200:
                return redirect('nullSubjects')

```

```

        user = user_response['data']
        group_id = user.get('idGroup')

        # Фильтруем предметы, где idGroup совпадает с группой пользователя
        subjects = [subject for subject in all_subjects if subject.get('idGroup') == group_id]
    else:
        return redirect('nullSubjects')

    if not subjects:
        return redirect('nullSubjects')

    # Получаем информацию о группах
    groups_response = make_api_request('Groups')
    groups = groups_response['data'] if groups_response['status_code'] == 200 else []

    # Получаем информацию о пользователях (преподавателях)
    users_response = make_api_request('Users')
    users = users_response['data'] if users_response['status_code'] == 200 else []

    context = {
        'subjects': subjects,
        'groups': groups,
        'users': users,
    }

    return render(request, 'AllSubjectPage.html', context)

except Exception as e:
    print(f"Error in AllSubjectPage: {e}")
    return redirect('nullSubjects')

def nullSubjects(request):
    return render(request, 'NullSubjects.html')

def login(request):
    if request.method == 'POST':
        login = request.POST.get('login')
        password = request.POST.get('password')

    if not login or not password:

```

```
return render(request, 'login.html', {'error': 'Логин и пароль обязательны', 'login': login})
```

```
try:
```

```
    # Получаем всех пользователей
```

```
    users_response = requests.get(f'{API_BASE_URL}/Users', verify=False)
```

```
    if users_response.status_code == 200:
```

```
        users = users_response.json()
```

```
        # Ищем пользователя по зашифрованному логину
```

```
        encoded_login = base64.b64encode(login.encode()).decode()
```

```
        user = next((u for u in users if u['login'] == encoded_login), None)
```

```
if user is None:
```

```
    # Логин не существует
```

```
    return render(request, 'login.html', {'error': 'Такого логина не существует', 'login': login})
```

```
if not bcrypt.checkpw(password.encode(), user['password'].encode()):
```

```
    # Пароль неверный
```

```
    return render(request, 'login.html', {'error': 'Неверный пароль', 'login': login})
```

```
# Пароль верный
```

```
request.session.flush()
```

```
request.session['user_id'] = user.get('idUser')
```

```
request.session['role_id'] = user.get('idRole')
```

```
request.session['full_name'] = f'{user.get('firstName')} {user.get('secondName')}'
```

```
request.session['last_activity'] = str(datetime.now())
```

```
request.session.set_expiry(60 * 60 * 24 * 14)
```

```
request.session.save()
```

```
next_url = request.session.get('next_url')
```

```
if next_url:
```

```
    del request.session['next_url']
```

```
    return redirect(next_url)
```

```
# Изменяем логику перенаправления в зависимости от роли
```

```
role_id = user.get('idRole')
```

```
if role_id == 1: # Администратор
```

```
    return redirect('adminPage')
```

```
else: # Преподаватель или студент
```

```
    return redirect('AllSubjectPage')
```

```
        else:
            return render(request, 'login.html', {'error': 'Ошибка при проверке учетных данных', 'login':
login})

    except Exception as e:
        print(f"Exception during authentication: {str(e)}")
        return render(request, 'login.html', {'error': f'Ошибка при аутентификации: {str(e)}', 'login':
login})

    return render(request, 'login.html')
```

```
def logout(request):
    request.session.flush()
    return redirect('login')

import re
import base64
import bcrypt
import requests
from django.shortcuts import render, redirect

def register(request):
    if request.method == 'POST':
        request.session.flush()

        # Получаем данные из формы
        first_name = request.POST.get('firstName')
        second_name = request.POST.get('secondName')
        middle_name = request.POST.get('middleName')
        email = request.POST.get('email')
        login = request.POST.get('login')
        password = request.POST.get('password')
```

```

# Список для хранения ошибок
errors = []

# Проверка на минимальную длину логина и пароля
if len(login) < 8:
    errors.append('Логин должен содержать минимум 8 символов.')
if len(password) < 8:
    errors.append('Пароль должен содержать минимум 8 символов.')

# Проверка на наличие цифры и специального символа в пароле
if not re.search(r'\d', password):
    errors.append('Пароль должен содержать как минимум одну цифру.')
if not re.search(r'[@#$%^&*(),.?":{}|<>]', password):
    errors.append('Пароль должен содержать как минимум один специальный символ.')

# Если есть ошибки, возвращаем их в шаблон
if errors:
    return render(request, 'registration.html', {
        'error': '\n'.join(errors),
        'first_name': first_name,
        'second_name': second_name,
        'middle_name': middle_name,
        'email': email,
        'login': login
    })

# Проверяем, существует ли пользователь с таким email или логином
try:
    users_response = requests.get(f'{API_BASE_URL}/Users', verify=False)
    if users_response.status_code == 200:
        users = users_response.json()

        # Проверка на существование email
        if any(user['email'].lower() == email.lower() for user in users):
            return render(request, 'registration.html', {
                'error': 'Пользователь с таким email уже существует',
                'first_name': first_name,
                'second_name': second_name,
                'middle_name': middle_name,
                'email': email,
                'login': login
            })

```

```

# Проверка на существование логина
if any(base64.b64decode(user['login']).decode() == login for user in users):
    return render(request, 'registration.html', {
        'error': 'Пользователь с таким логином уже существует',
        'first_name': first_name,
        'second_name': second_name,
        'middle_name': middle_name,
        'email': email,
        'login': login
    })

# Шифруем логин в base64
encoded_login = base64.b64encode(login.encode()).decode()

# Хешируем пароль с помощью bcrypt
salt = bcrypt.gensalt()
hashed_password = bcrypt.hashpw(password.encode(), salt).decode()

new_user = {
    'firstName': first_name,
    'secondName': second_name,
    'middleName': middle_name,
    'email': email,
    'login': encoded_login,
    'password': hashed_password,
    'idRole': 4,
    'idGroup': 6
}

print("Отправляемые данные:", new_user)
response = requests.post(f'{API_BASE_URL}/Users', json=new_user, verify=False)
print("Статус ответа:", response.status_code)
print("Ответ:", response.text)

if response.status_code in [200, 201]:
    user_data = response.json()
    request.session['user_id'] = user_data.get('idUser')
    request.session['role_id'] = 4
    request.session['full_name'] = f'{first_name} {second_name} {middle_name}'
    return redirect('nullSubjects')
else:

```

```

        error_message = f"Ошибка при регистрации. Статус: {response.status_code}"
        return render(request, 'registration.html', {
            'error': error_message,
            'first_name': first_name,
            'second_name': second_name,
            'middle_name': middle_name,
            'email': email,
            'login': login
        })
    )

except requests.exceptions.RequestException as e:
    print(f"Ошибка при регистрации: {str(e)}")
    return render(request, 'registration.html', {
        'error': 'Ошибка при регистрации: ' + str(e),
        'first_name': first_name,
        'second_name': second_name,
        'middle_name': middle_name,
        'email': email,
        'login': login
    })

return render(request, 'registration.html')

def group_crud(request):
    response_from_api = requests.get(f'{API_BASE_URL}/Groups', verify=False)

    if response_from_api.status_code == 200:
        groups = response_from_api.json()
    else:
        groups = []

    return render(request, 'AdminPanel/GroupCRUD.html', {'groups': groups})

def create_group(request):
    if request.method == 'POST':
        name_group = request.POST.get('nameGroup')
        response = requests.post(f'{API_BASE_URL}/Groups', json={'nameGroup': name_group},
                               verify=False)

```

```

        return redirect('group_crud')
        return HttpResponse(status=405)

def edit_group(request, id):
    if request.method == 'POST':
        name_group = request.POST.get('nameGroup')
        response = requests.put(f'{API_BASE_URL}/Groups/{id}', json={'idGroup': id, 'nameGroup': name_group}, verify=False)
        return redirect('group_crud')
        return HttpResponse(status=405)

def delete_group(request, id):
    if request.method == 'POST':
        response = requests.delete(f'{API_BASE_URL}/Groups/{id}', verify=False)
        return redirect('group_crud')
    return HttpResponse(status=405)

def user_crud(request):
    response_from_api_users = requests.get(f'{API_BASE_URL}/Users', verify=False)
    users = response_from_api_users.json() if response_from_api_users.status_code == 200 else []

    response_from_api_roles = requests.get(f'{API_BASE_URL}/Roles', verify=False)
    roles = response_from_api_roles.json() if response_from_api_roles.status_code == 200 else []

    response_from_api_groups = requests.get(f'{API_BASE_URL}/Groups', verify=False)
    groups = response_from_api_groups.json() if response_from_api_groups.status_code == 200 else []

    selected_group_id = request.GET.get('group_id')
    if selected_group_id:
        users = [user for user in users if user.get('idGroup') == int(selected_group_id)]

    search_query = request.GET.get('search', '').strip()
    if search_query:
        users = [user for user in users if (
            search_query.lower() in user['firstName'].lower() or
            search_query.lower() in user['secondName'].lower() or
            search_query.lower() in user['middleName'].lower()
        )]

    sort_by = request.GET.get('sort_by', 'secondName')

```

```

if sort_by == 'secondName':
    users.sort(key=lambda x: x['secondName'])
elif sort_by == 'firstName':
    users.sort(key=lambda x: x['firstName'])
elif sort_by == 'middleName':
    users.sort(key=lambda x: x['middleName'])
elif sort_by == 'group':
    users.sort(key=lambda x: x.get('idGroup', ''))

return render(request, 'AdminPanel/UserCRUD.html', {'users': users, 'roles': roles, 'groups': groups})

```

```

from django.contrib import messages
import re

def edit_user(request, id):
    if request.method == 'POST':
        user_data = {
            'IdUser': id,
            'FirstName': request.POST.get('firstName'),
            'SecondName': request.POST.get('secondName'),
            'MiddleName': request.POST.get('middleName'),
            'Email': request.POST.get('email'),
            'IdRole': request.POST.get('idRole'),
            'IdGroup': request.POST.get('idGroup'),
        }
        print(user_data)

        response = requests.put(f'{API_BASE_URL}/Users/{id}', json=user_data, verify=False)

        if response.status_code == 204:
            return redirect('user_crud')
        else:
            print("Ошибка при обновлении пользователя:", response.status_code, response.text)
            return HttpResponse("Ошибка при обновлении данных пользователя.", status=response.status_code)

    return HttpResponse(status=405)

```

```

def delete_user(request, id):
    if request.method == 'POST':
        response = requests.delete(f'{API_BASE_URL}/Users/{id}', verify=False)
        return redirect('user_crud')
    return HttpResponse(status=405)

def subject_crud(request):
    if request.method == 'POST':
        new_subject = {
            'nameSubject': request.POST.get('nameSubject'),
            'idGroup': request.POST.get('idGroup'),
            'userId': request.POST.get('userId'),
        }
        response = requests.post(f'{API_BASE_URL}/Subjects', json=new_subject, verify=False)
        return redirect('subject_crud')

    response_from_api_groups = requests.get(f'{API_BASE_URL}/Groups', verify=False)
    groups = response_from_api_groups.json() if response_from_api_groups.status_code == 200 else []

    response_from_api_users = requests.get(f'{API_BASE_URL}/Users', verify=False)
    users = response_from_api_users.json() if response_from_api_users.status_code == 200 else []

    response_from_api_subjects = requests.get(f'{API_BASE_URL}/Subjects', verify=False)
    subjects = response_from_api_subjects.json() if response_from_api_subjects.status_code == 200 else []

    return render(request, 'AdminPanel/SubjectCRUD.html', {'groups': groups, 'users': users, 'subjects': subjects})

def edit_subject(request, id):
    if request.method == 'POST':
        updated_subject = {
            'idSubject': id,
            'nameSubject': request.POST.get('nameSubject'),
            'idGroup': request.POST.get('idGroup'),
            'userId': request.POST.get('userId'),
        }
        response = requests.put(f'{API_BASE_URL}/Subjects/{id}', json=updated_subject, verify=False)
        return redirect('subject_crud')

    response_from_api_subject = requests.get(f'{API_BASE_URL}/Subjects/{id}', verify=False)

```

```
subject = response_from_api_subject.json() if response_from_api_subject.status_code == 200 else {}

response_from_api_groups = requests.get(f'{API_BASE_URL}/Groups', verify=False)
groups = response_from_api_groups.json() if response_from_api_groups.status_code == 200 else []

response_from_api_users = requests.get(f'{API_BASE_URL}/Users', verify=False)
users = response_from_api_users.json() if response_from_api_users.status_code == 200 else []

return render(request, 'AdminPanel/SubjectEdit.html', {
    'subject': subject,
    'groups': groups,
    'users': users,
})
```

```
def delete_subject(request, id):
    if request.method == 'POST':
        response = requests.delete(f'{API_BASE_URL}/Subjects/{id}', verify=False)
        return redirect('subject_crud')
    return HttpResponseRedirect(status=405)
```

```
def create_practical_work(request, subject_id):
    if request.method == 'POST':
        name_practical_work = request.POST.get('namePracticalWork')
        description_practical_work = request.POST.get('descriptionPracticalWork')
        name_file_practical_work = request.POST.get('nameFilePracticalWork')
        url_file_practical_work = request.POST.get('urlFilePracticalWork')
        user_id = request.session.get('user_id')

        # Получаем текущую дату и время
        date_upload = date.today()
        time_upload = datetime.now().time().strftime('%H:%M:%S')

        # Данные для API
        api_url = f'{API_BASE_URL}/PracticalWorks'
        data = {
            'namePracticalWork': name_practical_work,
            'descriptionPracticalWork': description_practical_work,
            'dateUploadPracticalWork': str(date_upload),
            'timeUploadPracticalWork': str(time_upload),
            'nameFilePracticalWork': name_file_practical_work,
```

```

        'urlFilePracticalWork': url_file_practical_work,
        'userId': user_id,
        'subjectId': subject_id
    }

try:
    response = requests.post(api_url, json=data, verify=False)
    response.raise_for_status()
    return redirect('SubjectPage', id=subject_id)
except requests.exceptions.RequestException as e:
    return render(request, 'SubjectInfo/PracticalWork.html', {
        'error': 'Ошибка при добавлении практической работы: ' + str(e),
    })

return render(request, 'SubjectInfo/CRUDPracticalWork.html')

```

```

def process_completed_works(completed_works, students=None):
    """
    Обрабатывает список выполненных работ
    """

    processed_works = []
    student_works = {}

    # Группируем работы по студентам и практическим работам
    for work in completed_works:
        user_id = work.get('userId')

        # Добавляем информацию о студенте
        if students:
            full_name = "Неизвестный студент"
            for student in students:
                if student.get('idUser') == user_id:
                    full_name = f'{student.get('firstName')} {student.get('secondName')} {student.get('middleName')}'.strip()
                    break
                work['student_name'] = full_name

    # Проверяем, есть ли attemptCount в данных
    if 'attemptCount' in work:
        # Если уже есть - просто используем

```

```

        work['attempt_count'] = work['attemptCount']

    else:
        # Если нет - используем camelCase версию ключа
        if 'attemptCount' in work:
            work['attempt_count'] = work['attemptCount']
        else:
            # Если информации о попытках нет, устанавливаем по умолчанию 1
            work['attempt_count'] = 1

    # Группировка работ по студентам для отображения в админ-панели
    if user_id not in student_works:
        student_works[user_id] = {
            'works': [],
            'full_name': work.get('student_name', "Неизвестный студент")
        }

        student_works[user_id]['works'].append(work)
        processed_works.append(work)

    # Сортируем работы каждого студента по дате
    for user_id in student_works:
        student_works[user_id]['works'].sort(
            key=lambda x: (
                x.get('dateUploadCompletedWork', ''),
                x.get('timeUploadCompletedWork', '')
            ),
            reverse=True # Сначала новые
        )

    return processed_works, student_works

```

```

def PracticalWorkMaterial(request, idPracticalWork=None, id=None):
    # Use id parameter if idPracticalWork is not provided
    if idPracticalWork is None and id is not None:
        idPracticalWork = id

    # Проверяем и обновляем сессию
    if not ensure_session_valid(request):

```

```

request.session['next_url'] = request.path
return redirect('login')

user_id = request.session['user_id']
role_id = request.session.get('role_id')

# Инициализируем переменные для работ и попыток
latest_work = None
completed_works = []
used_attempts = []

try:
    # Получаем информацию о практической работе
    practical_work_response = make_api_request(f'PracticalWorks/{idPracticalWork}')
    if practical_work_response['status_code'] != 200:
        return render(request, 'nullPage.html', {'message': 'Практическая работа не найдена'})

    practical_work = practical_work_response['data']

    # Получаем информацию о предмете
    subject_response = make_api_request(f'Subjects/{practical_work["subjectId"]}')
    if subject_response['status_code'] == 200:
        subject = subject_response['data']
    else:
        subject = None

    # Получаем сданные работы по этой практической работе
    completed_works_response = make_api_request('CompletedWorks')

    # Для учителей - мы будем показывать все работы по этому заданию
    students_completed_works = {}

    if completed_works_response['status_code'] == 200:
        # Фильтруем только работы по данной практической работе
        all_completed_works = completed_works_response['data']
        all_works_for_practical = [work for work in all_completed_works if
str(work.get('practicalWorkId')) == str(idPracticalWork)]

        # Получаем информацию о студентах
        students_response = make_api_request('Users')
        students = {}

```

```

if students_response['status_code'] == 200:
    students_data = students_response['data']

    # Фильтруем только студентов (роль 3)
    students_data = [student for student in students_data if student.get('idRole') == 3]

    # Создаем словарь студентов для быстрого доступа
    for student in students_data:
        students[student['idUser']] = student

    # Обрабатываем работы в зависимости от роли пользователя
    if role_id == 3: # Если студент
        # Находим только работы этого студента
        user_works = [work for work in all_works_for_practical if str(work.get('userId')) ==
str(user_id)]

        # Добавляем информацию о попытке в каждую работу
        for work in user_works:
            if 'attemptCount' in work:
                attempt_count = int(work.get('attemptCount', 1))
                work['attempt_count'] = attempt_count

            # Собираем уникальные номера попыток
            if attempt_count not in used_attempts:
                used_attempts.append(attempt_count)

        # Сортируем номера попыток
        used_attempts.sort()

        # Выводим отладочную информацию
        print(f"DEBUG: Работы пользователя: {len(user_works)}")
        print(f"DEBUG: Использованные попытки: {used_attempts}")

        # Присваиваем работы в completed_works для шаблона
        completed_works = user_works

        if user_works:
            # Сортируем по attemptCount и статусу проверки
            # Сначала на проверке (если есть), затем с высшим значением attemptCount
            works_on_check = [work for work in user_works if work.get('grade') == "На проверке"]
            checked_works = [work for work in user_works if work.get('grade') != "На проверке"]

```

```

# Внутри каждой категории - сортировка по attemptCount
works_on_check.sort(key=lambda x: int(x.get('attemptCount', 0)), reverse=True)
checked_works.sort(key=lambda x: int(x.get('attemptCount', 0)), reverse=True)

# Если есть работы на проверке, берем первую из них (последнюю по времени)
if works_on_check:
    latest_work = works_on_check[0]
    print(f"DEBUG: Отображаем последнюю работу НА ПРОВЕРКЕ:")
    ID={latest_work.get('idCompletedWork')}, "
        f"Попытка={latest_work.get('attemptCount')}, Оценка={latest_work.get('grade')}")

    elif checked_works: # Иначе берем проверенную с высшим attemptCount
        latest_work = checked_works[0]
        print(f"DEBUG: Отображаем последнюю ПРОВЕРЕННУЮ работу:")
        ID={latest_work.get('idCompletedWork')}, "
            f"Попытка={latest_work.get('attemptCount')}, Оценка={latest_work.get('grade')}")

    elif role_id == 2 or role_id == 1: # Преподаватель или администратор
        # Группируем работы по студентам для удобного отображения в панели преподавателя
        for work in all_works_for_practical:
            student_id = work.get('userId')

            if student_id in students:
                student = students[student_id]
                student_name = f"{student.get('firstName', '')} {student.get('secondName', '')} {student.get('middleName', '')}"
                work['student_name'] = student_name

            # Проверяем, проверена ли работа
            is_checked = work.get('grade') and work.get('grade') != "На проверке"
            work['is_checked'] = is_checked

        # Добавляем в группировку по студентам
        if student_id not in students_completed_works:
            students_completed_works[student_id] = {
                'full_name': student_name,
                'works': []
            }
            students_completed_works[student_id]['works'].append(work)

        # Сортируем работы каждого студента (для преподавателя)
        for student_id in students_completed_works:
            students_completed_works[student_id]['works'].sort()

```

```

key=lambda x: (
    x.get('dateUploadCompletedWork', ''),
    x.get('timeUploadCompletedWork', '')
),
reverse=True # Сначала новые
)

# Статистика для преподавателя
passed_count = 0
not_passed_count = 0

if role_id == 2 or role_id == 1:
    # Считаем только для преподавателя/админа, так как студенту это не нужно
    for student_id, student_info in students_completed_works.items():
        for work in student_info['works']:
            if work.get('grade') in ['3', '4', '5']:
                passed_count += 1
            else:
                not_passed_count += 1

# Подготовка контекста с учетом роли пользователя
context = {
    'practical_work': practical_work,
    'subject': subject,
    'role_id': role_id,
    'user_role_id': role_id,
    'user_id': user_id,
    'is_teacher': role_id == 2,
    'is_student': role_id == 3, # Явно устанавливаем флаг для студента
    'is_admin': role_id == 1,
    'students_completed_works': students_completed_works,
    'passed_count': passed_count if 'passed_count' in locals() else 0,
    'not_passed_count': not_passed_count if 'not_passed_count' in locals() else 0,
    'latest_work': latest_work,
    'completed_works': completed_works,
    'used_attempts': used_attempts, # Добавляем список использованных попыток
    'total_attempts': 3, # Общее количество доступных попыток
    'remaining_attempts': 3 - len(used_attempts) # Оставшиеся попытки
}

# Добавляем отладочную информацию в контекст
context['debug_attempts'] = {

```

```

        'count': len(used_attempts),
        'values': used_attempts,
        'is_student_flag': role_id == 3
    }

    return render(request, 'SubjectInfo/PracticalWorkMaterial.html', context)

except Exception as e:
    print(f"Ошибка при получении данных практической работы: {e}")
    return redirect('AllSubjectPage') # Было return render('nullPage.html', ...)

```

```

def delete_practical_work(request, id):
    if request.method == 'POST':
        try:
            # Получаем информацию о практической работе перед удалением
            practical_work_url = f'{API_BASE_URL}/PracticalWorks/{id}'
            response_get = requests.get(practical_work_url, verify=False)

            if response_get.status_code == 200:
                practical_work = response_get.json()
                subject_id = practical_work.get('subjectId')

                # 1. Сначала получаем все выполненные работы для данной практической работы
                completed_works_url = f'{API_BASE_URL}/CompletedWorks'
                completed_works_response = requests.get(completed_works_url, verify=False)

                if completed_works_response.status_code == 200:
                    completed_works = completed_works_response.json()
                    # Фильтруем работы, относящиеся к удаляемой практической работе
                    related_works = [work for work in completed_works
                                     if work.get('practicalWorkId') == id]

                    # 2. Удаляем каждую выполненную работу
                    for work in related_works:
                        delete_completed_url
                        f'{API_BASE_URL}/CompletedWorks/{work["idCompletedWork"]}' =
                        delete_completed_response = requests.delete(delete_completed_url, verify=False)

```

```

        print(f"Удаление выполненной работы {work['idCompletedWork']}:

{delete_completed_response.status_code}")

# 3. После удаления всех связанных работ удаляем саму практическую работу
response_delete = requests.delete(practical_work_url, verify=False)

if response_delete.status_code == 204: # Успешное удаление
    messages.success(request, 'Практическая работа и все связанные с ней попытки успешно
удалены')
    return redirect('SubjectPage', id=subject_id)
else:
    messages.error(request, 'Ошибка при удалении практической работы')
else:
    messages.error(request, 'Практическая работа не найдена')

# В случае ошибки возвращаемся на страницу предмета
return redirect('SubjectPage', id=request.POST.get('subjectId'))

except Exception as e:
    print(f"Ошибка при удалении: {str(e)}")
    messages.error(request, f'Произошла ошибка: {str(e)}')
    return redirect('SubjectPage', id=request.POST.get('subjectId'))

# Если метод не POST
return HttpResponse("Метод не разрешен", status=405)

def edit_practical_work(request, id):
    practical_work_url = f'{API_BASE_URL}/PracticalWorks/{id}'

    try:
        response_from_api = requests.get(practical_work_url, verify=False)
        response_from_api.raise_for_status()
        practical_work = response_from_api.json()
    except requests.exceptions.RequestException as e:
        return render(request, 'SubjectInfo/PracticalWorkEdit.html', {
            'error': 'Ошибка при получении практической работы: ' + str(e),
        })

    if request.method == 'POST':
        updated_data = {

```

```

'idPracticalWork': id,
'namePracticalWork': request.POST.get('namePracticalWork',
practical_work['namePracticalWork']),
'descriptionPracticalWork': request.POST.get('descriptionPracticalWork',
practical_work['descriptionPracticalWork']),
'dateUploadPracticalWork': practical_work['dateUploadPracticalWork'],
'timeUploadPracticalWork': practical_work['timeUploadPracticalWork'],
'nameFilePracticalWork': request.POST.get('nameFilePracticalWork',
practical_work['nameFilePracticalWork']),
'urlFilePracticalWork': request.POST.get('urlFilePracticalWork',
practical_work['urlFilePracticalWork']),
'userId': practical_work['userId'],
'subjectId': practical_work['subjectId'],
}

try:
    response = requests.put(practical_work_url, json=updated_data, verify=False)
    response.raise_for_status()
    return redirect('SubjectPage', id=practical_work['subjectId'])
except requests.exceptions.RequestException as e:
    return render(request, 'SubjectInfo/PracticalWorkEdit.html', {
        'error': 'Ошибка при обновлении практической работы: ' + str(e),
        'practical_work': practical_work,
    })

return render(request, 'SubjectInfo/PracticalWorkEdit.html', {'practical_work': practical_work})

def AddCompleteWork(request, practical_work_id):
    """
    Функция для добавления выполненной работы
    """
    # Получение данных о практической работе
    try:
        # Получаем данные о практической работе
        practical_work_response = make_api_request(f"PracticalWorks/{practical_work_id}")

        if practical_work_response['status_code'] != 200:
            # Логирование ошибки
            print(f"Ошибка API: {practical_work_response['status_code']}")

            return HttpResponse("Ошибка при получении данных о практической работе")

        practical_work = practical_work_response['data']
    
```

```

# Проверяем, авторизован ли пользователь
if 'user_id' not in request.session:
    return redirect('login')

user_id = request.session['user_id']

# ИСПРАВЛЕНИЕ: Получаем все работы для подсчета попыток
print(f"Получение списка всех выполненных работ для определения номера попытки")
completed_works_response = make_api_request("CompletedWorks")

# По умолчанию - первая попытка
attempt_count = 1
existing_works = []

if completed_works_response['status_code'] == 200 and completed_works_response['data']:
    # Фильтруем работы для конкретного пользователя и практической работы
    all_works = completed_works_response['data']

    # ИСПРАВЛЕНО: Более точная фильтрация работ студента
    user_works = []
    for work in all_works:
        work_user_id = work.get('userId')
        work_practical_id = work.get('practicalWorkId')

        # Преобразуем идентификаторы к одному типу для сравнения
        if str(work_user_id) == str(user_id) and str(work_practical_id) == str(practical_work_id):
            user_works.append(work)
            print(f"Найдена           работа:           ID={work.get('idCompletedWork')},"
                  f"попытка={work.get('attemptCount')}")

    print(f"Найдено {len(user_works)} существующих работ пользователя {user_id} для"
          f"практической работы {practical_work_id}")

    if user_works:
        existing_works = user_works
        # ИСПРАВЛЕНО: Используем самый большой номер попытки из существующих работ
        max_attempt = 0
        for work in user_works:
            work_attempt = work.get('attemptCount')
            if work_attempt and int(work_attempt) > max_attempt:
                max_attempt = int(work_attempt)

```

```

# Следующая попытка будет на 1 больше максимальной существующей
attempt_count = max_attempt + 1
print(f"Максимальная существующая попытка: {max_attempt}")
print(f"Номер текущей попытки: {attempt_count}")

else:
    print(f"Не удалось получить список работ. Статус: {completed_works_response['status_code']}")

# Ограничиваем максимальное количество попыток
max_attempts_reached = attempt_count > 3
if max_attempts_reached:
    attempt_count = 3 # Отображаем максимум 3

if request.method == 'POST':
    if max_attempts_reached:
        # Если превышено количество попыток, показываем сообщение об ошибке
        return render(request, 'SubjectInfo/AddCompleteWork.html', {
            'practical_work': practical_work,
            'attempt_count': 3, # Показываем 3, так как больше быть не может
            'max_attempts_reached': True,
            'subject_id': practical_work.get('subjectId')
        })

file_name = request.POST.get('nameFileCompletedWork')
url_link = request.POST.get('urlFileCompletedWork')

if not file_name or not url_link:
    return render(request, 'SubjectInfo/AddCompleteWork.html', {
        'practical_work': practical_work,
        'attempt_count': attempt_count,
        'max_attempts_reached': max_attempts_reached,
        'subject_id': practical_work.get('subjectId'),
        'error': 'Необходимо указать название файла и ссылку'
    })

# Отправляем работу с корректным attempt_count
result = send_completed_work_direct(user_id, practical_work_id, file_name, url_link)

if result:
    # Исправляем перенаправление, используя параметр 'id' вместо 'idPracticalWork'
    return redirect('PracticalWorkMaterial', id=practical_work_id)

```

```

else:
    return render(request, 'SubjectInfo/AddCompleteWork.html', {
        'practical_work': practical_work,
        'attempt_count': attempt_count,
        'max_attempts_reached': max_attempts_reached,
        'subject_id': practical_work.get('subjectId'),
        'error': 'Ошибка при отправке работы'
    })
else:
    # GET запрос - показываем форму
    return render(request, 'SubjectInfo/AddCompleteWork.html', {
        'practical_work': practical_work,
        'attempt_count': attempt_count,
        'max_attempts_reached': max_attempts_reached,
        'subject_id': practical_work.get('subjectId')
    })

except Exception as e:
    print(f"Ошибка в AddCompleteWork: {e}")
    return redirect('AllSubjectPage') # Было return redirect('nullPage')

def send_completed_work_direct(user_id, practical_work_id, file_name, url_link):
    print("\n--- Отправка данных о выполненной работе ---")

    # Получаем все работы и фильтруем
    completed_works_response = make_api_request("CompletedWorks")

    # По умолчанию - первая попытка
    attempt_count = 1

    if completed_works_response['status_code'] == 200 and completed_works_response['data']:
        all_works = completed_works_response['data']

        # ИСПРАВЛЕНО: Более точная фильтрация и подсчет попыток
        user_works = []
        for work in all_works:
            work_user_id = work.get('userId')
            work_practical_id = work.get('practicalWorkId')

            # Преобразуем идентификаторы к одному типу для сравнения
            if str(work_user_id) == str(user_id) and str(work_practical_id) == str(practical_work_id):
                user_works.append(work)

```

```

        print(f"Найдена существующая работа: ID={work.get('idCompletedWork')},
попытка={work.get('attemptCount')}")

        print(f"Найдено {len(user_works)} работ пользователя {user_id}")

if user_works:
    # Находим максимальный номер попытки
    max_attempt = 0
    for work in user_works:
        work_attempt = work.get('attemptCount')
        if work_attempt and int(work_attempt) > max_attempt:
            max_attempt = int(work_attempt)

    # Следующая попытка будет на 1 больше максимальной существующей
    attempt_count = max_attempt + 1
    print(f"Максимальная существующая попытка: {max_attempt}")
    print(f"Номер новой попытки: {attempt_count}")

# Проверяем, не превышено ли максимальное число попыток
if attempt_count > 3:
    print(f"Превышено максимальное количество попыток: {attempt_count}")
    return False

print(f"Установлен номер попытки: {attempt_count}")

# Создаем текущую дату и время
import datetime
now = datetime.datetime.now()
date_str = now.strftime("%Y-%m-%d")
time_str = now.strftime("%H:%M:%S")

# Создаем данные для отправки
data = {
    "NameFileCompletedWork": file_name,
    "UrlFileCompletedWork": url_link,
    "Grade": "На проверке",
    "StatusId": 2,
    "UserId": user_id,
    "PracticalWorkId": practical_work_id,
    "AttemptCount": attempt_count,
    "DateUploadCompletedWork": date_str,
    "TimeUploadCompletedWork": time_str
}

```

```

}

# URL для отправки данных
url = f'{API_BASE_URL}/CompletedWorks'

try:
    # Отправляем данные на сервер
    response = requests.post(url, json=data, verify=False)
    print(f"Статус ответа: {response.status_code}")

    if response.status_code in [200, 201, 204]:
        print("Работа успешно добавлена!")
        return True

    else:
        print(f'Ошибка при добавлении работы: {response.text}')

    # Если первая попытка не удалась, пробуем другие варианты URL
    urls_to_try = [
        f'{API_BASE_URL}/completedworks', # нижний регистр
        f'{API_BASE_URL}/completedWorks', # смешанный регистр
        f'{API_BASE_URL}/CompletedWorks', # с закрывающим слешем
    ]

    for alt_url in urls_to_try:
        print(f'Пробуем альтернативный URL: {alt_url}')
        try:
            alt_response = requests.post(alt_url, json=data, verify=False)
            print(f"Статус ответа: {alt_response.status_code}")

            if alt_response.status_code in [200, 201, 204]:
                print(f"Работа успешно добавлена с URL: {alt_url}")
                return True

            except Exception as e:
                print(f'Ошибка при использовании URL {alt_url}: {str(e)}')

        return False

    except Exception as e:
        print(f'Исключение при отправке данных: {str(e)}')
        return False

def grade_completed_work(request, practical_work_id, user_id):
    try:

```

```

# Получение работ остается без изменений
completed_works_response = make_api_request(f"CompletedWorks")

if completed_works_response['status_code'] != 200 or not completed_works_response['data']:
    print(f"Не удалось получить список всех выполненных работ")
    return HttpResponse("Ошибка при получении данных о выполненных работах")

completed_works = completed_works_response['data']
user_works = [work for work in completed_works
              if work['userId'] == int(user_id) and work['practicalWorkId'] == int(practical_work_id)]

if not user_works:
    print(f"Работа пользователя с ID {user_id} для практической работы {practical_work_id} не
найдена")
    return HttpResponse("Выполненная работа не найдена")

# Берем последнюю работу
completed_work = max(user_works, key=lambda w: w.get('attemptCount', 0))
print(f"Найдена работа пользователя: {completed_work}")

# Получим список допустимых статусов
status_response = make_api_request(f"Status")
valid_statuses = []
if status_response['status_code'] == 200 and status_response['data']:
    valid_statuses = status_response['data']
    print(f"Доступные статусы: {valid_statuses}")

# Выведем статусы для отладки
for status in valid_statuses:
    print(f"Статус: {status['idStatus']} - {status['nameStatus']}")

if request.method == 'POST':
    grade = request.POST.get('grade')
    print(f"Полученная оценка: {grade}")

# ИСПРАВЛЕННАЯ ЛОГИКА определения статуса на основе оценки
# Вначале установим переменные для четкого понимания
STATUS_PASSED = 1 # ID статуса "Задание сдано"
STATUS_FAILED = 2 # ID статуса "Задание не сдано"

status_id = None
if grade:

```

```

try:
    # Преобразуем оценку в число если возможно, или оставляем как строку
    grade_value = int(grade) if grade.isdigit() else grade
    print(f"Преобразованная оценка: {grade_value}, тип: {type(grade_value)}")

    # Явно проверяем равенство и устанавливаем соответствующий статус
    if grade_value == 2:
        status_id = STATUS_FAILED # Оценка 2 -> "Задание не сдано"
        print(f"Оценка 2 -> устанавливаем статус {STATUS_FAILED} (Задание не сдано)")

    elif grade_value in [3, 4, 5]:
        status_id = STATUS_PASSED # Оценки 3,4,5 -> "Задание сдано"
        print(f"Оценка {grade_value} -> устанавливаем статус {STATUS_PASSED} (Задание сдано)")

    else:
        # Если оценка не 2,3,4,5 - берем статус из формы или оставляем текущий
        status_id = request.POST.get('statusId', completed_work['statusId'])
        print(f"Оценка {grade_value} не распознана -> используем статус из формы: {status_id}")

except Exception as e:
    print(f"Ошибка при определении статуса по оценке: {str(e)}")
    status_id = request.POST.get('statusId', completed_work['statusId'])

else:
    # Если оценка не указана, берем статус из формы
    status_id = request.POST.get('statusId', completed_work['statusId'])
    print(f"Оценка не указана -> используем статус из формы: {status_id}")

print(f"Итоговые данные: grade={grade}, statusId={status_id}")

# Создаем копию оригинального объекта, чтобы сохранить все поля
updated_work = completed_work.copy()

# Обновляем только необходимые поля
updated_work['grade'] = grade
if status_id:
    updated_work['statusId'] = int(status_id)

# В случае, если в базе идентификаторы называются по-другому
if 'statusId' not in updated_work and 'StatusId' in updated_work:
    updated_work['StatusId'] = int(status_id) if status_id else updated_work['StatusId']

print(f"Отправляем объект для обновления: {updated_work}")

```

```

# Используем правильный URL для обновления данных
update_url = f'{API_BASE_URL}/CompletedWorks/{completed_work['idCompletedWork']}'
print(f"Обновляем оценку по URL: {update_url}")

try:
    # Явно указываем заголовок Content-Type
    headers = {'Content-Type': 'application/json'}
    response = requests.put(update_url, json=updated_work, headers=headers, verify=False)
    print(f"Статус обновления: {response.status_code}")

    if response.status_code in [200, 201, 204]:
        # Исправленное перенаправление - используем 'id' вместо 'idPracticalWork'
        return redirect('PracticalWorkMaterial', id=practical_work_id)

    else:
        print(f"Ответ сервера: {response.text}")
        return HttpResponse(f"Ошибка при обновлении оценки: {response.status_code}, {response.text}")

    except Exception as e:
        print(f"Исключение при обновлении: {str(e)}")
        return HttpResponse(f"Ошибка при обновлении оценки: {str(e)}")

# Код для GET-запроса остается без изменений
practical_work_response = make_api_request(f"PracticalWorks/{practical_work_id}")
if practical_work_response['status_code'] != 200:
    return HttpResponse("Ошибка при получении данных о практической работе")

practical_work = practical_work_response['data']

context = {
    'completed_work': completed_work,
    'practical_work': practical_work,
    'user_id': user_id,
    'statuses': valid_statuses
}

return render(request, 'SubjectInfo/GradeCompletedWork.html', context)

except Exception as e:
    print(f"Исключение в функции grade_completed_work: {str(e)}")
    return HttpResponse(f"Произошла ошибка: {str(e)}")

```

```

def create_test(request, subject_id):
    if request.method == 'POST':
        try:
            # Создаем тест
            test_data = {
                'nameTest': request.POST.get('nameTest'),
                'descriptionTest': request.POST.get('descriptionTest'),
                'timeLimit': int(request.POST.get('timeLimit')),
                'subjectId': subject_id
            }

            test_url = f'{API_BASE_URL}/Tests'
            response = requests.post(test_url, json=test_data, verify=False)

            if response.status_code not in [200, 201]:
                messages.error(request, 'Ошибка при создании теста')
                return redirect('create_test', subject_id=subject_id)

            test = response.json()
            test_id = test.get('idTest')

            # Создаем критерии оценивания
            criteria_url = f'{API_BASE_URL}/GradingCriterions'

            # Создаем критерии для каждой оценки - используем только один способ создания критериев
            criteria_data = [
                {
                    'idTest': test_id,
                    'grade': 5,
                    'minPoints': int(float(request.POST.get('grade5Min'))), # Преобразуем в целое число
                    'maxPoints': int(float(request.POST.get('grade5Max')))) # Преобразуем в целое число
                },
                {
                    'idTest': test_id,
                    'grade': 4,
                    'minPoints': int(float(request.POST.get('grade4Min'))),
                    'maxPoints': int(float(request.POST.get('grade4Max'))))
                },
                {
                    'idTest': test_id,
                    'grade': 3,
                    'minPoints': int(float(request.POST.get('grade3Min')))),

```

```

        'maxPoints': int(float(request.POST.get('grade3Max'))))
    }
]

# Отправляем каждый критерий
for criterion in criteria_data:
    response = requests.post(criteria_url, json=criterion, verify=False)
    print(f"Создание критерия для оценки {criterion['grade']}:")
    print(f"Отправленные данные: {criterion}")
    print(f"Статус ответа: {response.status_code}")
    print(f"Ответ сервера: {response.text}")

    if response.status_code not in [200, 201]:
        messages.warning(request, f'Ошибка при создании критерия для оценки {criterion["grade"]}')

# Остальной код остается без изменений...

# Получаем вопросы из формы
questions_json = request.POST.get('questions')
if questions_json:
    try:
        questions = json.loads(questions_json)
        # Создаем вопросы
        questions_url = f'{API_BASE_URL}/QuestionsTests'
        for question in questions:
            question_data = {
                'questionText': question['text'],
                'questionType': question['type'],
                'points': question['points'],
                'correctAnswer': question['correctAnswer'],
                'answerVariants': ';' .join(question['answers']) if question['answers'] else '',
                'idTest': test_id
            }
            response = requests.post(questions_url, json=question_data, verify=False)
            if response.status_code not in [200, 201]:
                print(f'Ошибка при создании вопроса: {response.status_code}')
                print(f'Ответ сервера: {response.text}')
    except json.JSONDecodeError as e:
        print(f'Ошибка при разборе JSON вопросов: {e}')

messages.success(request, 'Тест успешно создан')

```

```

        return redirect('test_material', id=test_id)

    except Exception as e:
        print(f"Ошибка при создании теста: {str(e)}")
        messages.error(request, f'Ошибка при создании теста: {str(e)}')
        return redirect('create_test', subject_id=subject_id)

# GET запрос - показываем форму создания теста
return render(request, 'SubjectInfo/CreateTest.html', {
    'subject_id': subject_id
})

def get_test_questions(test_id):
    """Вспомогательная функция для получения вопросов теста"""
    questions_response = make_api_request('QuestionsTests')
    if questions_response['status_code'] == 200 and questions_response['data']:
        # Фильтруем вопросы для конкретного теста по idTest
        questions = [q for q in questions_response['data'] if q.get('idTest') == int(test_id)]
        print(f"Получено {len(questions)} вопросов для теста {test_id}")
        return questions
    return []

def test_material(request, id):
    # Получаем данные о тесте
    logger.info(f"Получение данных теста с ID: {id}")
    response = make_api_request(f'Tests/{id}')

    # Проверяем статус код в словаре
    status_code = response['status_code']
    test_data = response['data'] if status_code == 200 else None

    if status_code == 200 and test_data:
        test = test_data

        # Получаем имя предмета
        subject_name = ""
        subject_id = test.get('subjectId')
        if subject_id:
            subject_response = make_api_request(f'Subjects/{subject_id}')
            subject_status_code = subject_response['status_code']
            subject_data = subject_response['data'] if subject_status_code == 200 else None

```

```

if subject_status_code == 200 and subject_data:
    subject = subject_data
    subject_name = subject.get('nameSubject', '')

# Определяем роль пользователя
role_id = request.session.get('role_id')
is_teacher = role_id in [1, 2] # Администратор или преподаватель
is_student = role_id == 3      # Студент

# Инициализируем переменные
questions = []
grading_criteria = []

# Получаем вопросы теста ТОЛЬКО для преподавателей
if is_teacher:
    print(f"Получение вопросов для преподавателя, тест ID: {id}")
    questions = get_test_questions(id)

# Получаем критерии оценивания для конкретного теста
print(f"Получение критериев оценивания для теста ID: {id}")
criteria_response = make_api_request('GradingCriterions')
if criteria_response['status_code'] == 200 and criteria_response['data']:
    # Фильтруем критерии только для текущего теста
    all_criteria = criteria_response['data']
    grading_criteria = [
        criterion for criterion in all_criteria
        if criterion.get('idTest') == int(id) # Убедитесь, что типы совпадают
    ]
    print(f"Получено {len(grading_criteria)} критериев оценивания для теста {id}")
    print(f"Критерии: {grading_criteria}") # Добавляем для отладки
else:
    print(f"Ошибка при получении критериев оценивания: {criteria_response['status_code']}")
    print(f"Ответ сервера: {criteria_response.get('data')}") # Добавляем для отладки

# Сортируем критерии по оценке (от высшей к низшей)
if grading_criteria:
    grading_criteria.sort(key=lambda x: float(x.get('grade', 0)), reverse=True)

context = {
    'test': test,
    'questions': questions,
}

```

```

'grading_criteria': grading_criteria,
'teacher_view': is_teacher,
'student_view': is_student,
'subject_name': subject_name,
'subject_id': subject_id,
'show_debug': True, # Включаем отладочную информацию
'debug_info': {
    'criteria_response': criteria_response,
    'test_id': id,
    'found_criteria': grading_criteria
}
}

# Если пользователь - студент, добавляем информацию о попытках
if is_student:
    user_id = request.session.get('user_id')
    if user_id:
        # Получаем все попытки
        attempts_url = f'{API_BASE_URL}/TestAttempts'
        try:
            response = requests.get(attempts_url, verify=False)
            all_attempts = response.json() if response.status_code == 200 else []
        except requests.exceptions.RequestException as e:
            print(f"DEBUG: Ошибка при запросе к API: {e}")

        # Фильтруем попытки для конкретного теста и пользователя
        attempts = [
            attempt for attempt in all_attempts
            if str(attempt.get('testId')) == str(id)
            and str(attempt.get('userId')) == str(user_id)
        ]
    else:
        print(f"DEBUG: Пользователь не найден")

    print(f"DEBUG: Найдено {len(attempts)} попыток для теста {id} и пользователя {user_id}")

    for attempt in attempts:
        print(f"DEBUG: Попытка: {attempt}")

    # Инициализируем переменные
    attempts_count = len(attempts)
    max_attempts = 3
    remaining_attempts = max(0, max_attempts - attempts_count)
    latest_attempt = None
    can_retake = True

```

```

# Получаем последнюю попытку, если есть
if attempts:
    # Сортируем по времени окончания
    attempts.sort(
        key=lambda x: x.get('endTime', '0') or '0',
        reverse=True
    )
    latest_attempt = attempts[0]
    print(f"DEBUG: Последняя попытка: {latest_attempt}")

# Проверка на успешное прохождение
try:
    grade = latest_attempt.get('grade')
    if grade:
        grade = float(grade)
        if grade >= 3:
            can_retake = False
            print(f"DEBUG: Тест уже сдан с оценкой {grade}")
        else:
            can_retake = remaining_attempts > 0
            print(f"DEBUG: Тест не сдан, оценка {grade}, осталось попыток:
{remaining_attempts}")
    else:
        print("DEBUG: Оценка отсутствует")
except (ValueError, TypeError) as e:
    print(f"DEBUG: Ошибка при обработке оценки: {e}")
    can_retake = remaining_attempts > 0

context.update({
    'attempts': attempts,
    'attempts_count': attempts_count,
    'max_attempts': max_attempts,
    'remaining_attempts': remaining_attempts,
    'latest_attempt': latest_attempt,
    'can_retake': can_retake
})

except Exception as e:
    print(f"Ошибка при получении попыток теста: {e}")
    context.update({
        'attempts': [],
        'attempts_count': 0,
    })

```

```

        'max_attempts': 3,
        'remaining_attempts': 3,
        'latest_attempt': None,
        'can_retake': True
    })

return render(request, 'SubjectInfo/TestMaterial.html', context)
}

def start_test(request, id):
    """Функция для начала прохождения теста."""
    # Получаем информацию о тесте
    test_url = f'{API_BASE_URL}/Tests/{id}'

    try:
        response = requests.get(test_url, verify=False)
        test = response.json() if response.status_code == 200 else {}

        if not test:
            messages.error(request, 'Не удалось загрузить информацию о тесте.')
            return redirect('test_material', id=id)

    except:
        messages.error(request, 'Не удалось загрузить информацию о тесте.')
        return redirect('test_material', id=id)

    # Получаем вопросы теста
    questions_url = f'{API_BASE_URL}/QuestionsTests'
    try:
        # Получаем все вопросы
        response_questions = requests.get(questions_url, verify=False)
        all_questions = response_questions.json() if response_questions.status_code == 200 else []

        # Фильтруем вопросы только для текущего теста
        questions = [
            question for question in all_questions
            if str(question.get('idTest')) == str(id)
        ]

        print(f"DEBUG: Найдено {len(questions)} вопросов для теста {id}")

        if not questions:
            messages.error(request, 'Для этого теста не найдены вопросы.')
            return redirect('test_material', id=id)

    except:
        messages.error(request, 'Не удалось загрузить вопросы для теста.')
        return redirect('test_material', id=id)
}

```

```

except Exception as e:
    print(f"Ошибка при получении вопросов: {e}")
    messages.error(request, 'Не удалось загрузить вопросы теста.')
    return redirect('test_material', id=id)

# Создаем запись о начале тестирования
user_id = request.session.get('user_id')

# Текущая дата и время
current_date = datetime.now().strftime('%Y-%m-%d')
current_time = datetime.now().strftime('%H:%M:%S')

# Текущая дата и время в ISO формате
start_time = datetime.now().strftime("%Y-%m-%dT%H:%M:%S.%f")[:-3]

# Записываем начало попытки
attempt_data = {
    'dateTestAttempt': current_date,
    'timeTestAttempt': current_time,
    'score': 0,
    'testId': id,
    'userId': user_id,
    'startTime': start_time, # Добавляем время начала
    'endTime': None, # Время окончания пока null
    'status': 'InProgress' # Добавляем статус
}

# Создаем новую попытку через API
attempt_id = None
try:
    attempt_url = f'{API_BASE_URL}/TestAttempts'
    response = requests.post(attempt_url, json=attempt_data, verify=False)
    if response.status_code in [200, 201]:
        attempt = response.json()
        attempt_id = attempt.get('idTestAttempt')
except Exception as e:
    print(f"Ошибка при создании попытки: {e}")

# Выводим вопросы для тестирования
return render(request, 'SubjectInfo/TestProcess.html', {
    'test': test,
    'questions': questions,
})

```

```

'time_limit': test.get('timeLimit', 60),
'attempt_id': attempt_id,
'start_time': start_time # Передаем время начала в шаблон
})

def submit_test(request, id):
    print("\n==== НАЧАЛО ПРОВЕРКИ ТЕСТА ====")

    try:
        # Получаем user_id из сессии в начале функции
        user_id = request.session.get('user_id')
        if not user_id:
            raise Exception("Пользователь не авторизован")

        # Инициализируем переменные для подсчета баллов
        total_score = 0
        max_score = 0

        # Получаем все вопросы теста
        questions_response = make_api_request(f"QuestionsTests")
        if questions_response['status_code'] != 200:
            raise Exception("Не удалось получить вопросы теста")

        questions = [q for q in questions_response['data']
                     if str(q.get('idTest')) == str(id)]

        # Проверяем каждый вопрос
        for question in questions:
            question_id = question.get('idQuestion')
            question_type = question.get('questionType')
            max_score += question.get('points', 0)

            print(f"\nПроверка вопроса {question_id} (тип {question_type}):")
            print(f"Текст вопроса: {question.get('questionText')}")

            # Проверяем ответ в зависимости от типа вопроса
            if question_type == 1: # Один правильный ответ
                total_score += check_single_choice(request, question_id, question)
            elif question_type == 2: # Несколько правильных ответов
                total_score += check_multiple_choice(request, question_id, question)
            elif question_type == 3: # Сопоставление

```

```

        total_score += check_matching(request, question_id, question)
    elif question_type == 4: # Ввод ответа
        total_score += check_text_input(request, question_id, question)
    elif question_type == 5: # Установление порядка
        total_score += check_ordering(request, question_id, question)

print(f"\nИтоговый результат: {total_score} из {max_score} баллов")

# Сохраняем результаты
current_datetime = datetime.now()
end_time = current_datetime.strftime("%Y-%m-%dT%H:%M:%S.%f")[:-3]

# Рассчитываем оценку
grade = calculate_grade(total_score, max_score, id)

# Получаем ID попытки из формы
attempt_id = request.POST.get('attempt_id')

# Подготавливаем данные для сохранения
attempt_data = {
    'testId': int(id),
    'userId': int(user_id),
    'score': total_score,
    'maxScore': max_score,
    'grade': grade,
    'startTime': request.POST.get('start_time'),
    'endTime': end_time,
    'status': 'Completed'
}

# Если есть ID попытки, добавляем его
if attempt_id and attempt_id.lower() != 'none':
    attempt_data['idTestAttempt'] = int(attempt_id)
    # Обновляем существующую попытку
    response = requests.put(
        f'{API_BASE_URL}/TestAttempts/{attempt_id}',
        json=attempt_data,
        verify=False
    )
else:
    # Создаем новую попытку
    response = requests.post(

```

```

        f'{API_BASE_URL}/TestAttempts',
        json=attempt_data,
        verify=False
    )

if response.status_code in [200, 201, 204]:
    messages.success(
        request,
        f'Тест успешно завершен! Ваш результат: {total_score} из {max_score} баллов. Оценка: {grade}'
    )
    return redirect('test_results', id=id)
else:
    print(f'Ошибка сохранения результатов: {response.status_code}')
    print(f'Ответ сервера: {response.text}')
    raise Exception(f'Ошибка сохранения результатов: {response.status_code}')

except Exception as e:
    print(f'Ошибка при отправке результатов теста: {e}')
    messages.error(request, f'Произошла ошибка при отправке результатов теста: {str(e)}')
    return redirect('test_material', id=id)

def calculate_grade(score, max_score, test_id):
    """Функция для расчета оценки на основе критериев оценивания"""
    try:
        # Получаем критерии оценивания для конкретного теста
        criteria_response = make_api_request('GradingCriterions')
        if criteria_response['status_code'] != 200:
            return calculate_default_grade(score, max_score)

        # Фильтруем критерии только для текущего теста
        all_criteria = criteria_response['data']
        criteria = [
            criterion for criterion in all_criteria
            if str(criterion.get('idTest')) == str(test_id)
        ]

        if not criteria:
            return calculate_default_grade(score, max_score)

        print(f'\nПодсчет оценки для теста {test_id}:')
        print(f'Набрано баллов: {score}')
    
```

```

print(f"Максимум баллов: {max_score}")
print("Критерии оценивания:")
for criterion in criteria:
    print(f"Оценка {criterion['grade']}: {criterion['minPoints']}-{criterion['maxPoints']} баллов")

# Сортируем критерии по оценке (от высшей к низшей)
sorted_criteria = sorted(criteria, key=lambda x: float(x['grade']), reverse=True)

# Проходим по критериям и находим подходящую оценку
for criterion in sorted_criteria:
    min_points = float(criterion['minPoints'])
    max_points = float(criterion['maxPoints'])

    if min_points <= score <= max_points:
        print(f"Итоговая оценка: {criterion['grade']} (набрано {score} баллов)")
        return criterion['grade']

# Если набрано меньше минимума - ставим 2
print(f"Набрано баллов меньше минимума, оценка: 2")
return 2

except Exception as e:
    print(f"Ошибка при расчете оценки: {e}")
    return calculate_default_grade(score, max_score)

def calculate_default_grade(score, max_score):
    """Стандартная шкала оценок, если критерии недоступны"""
    if max_score == 0:
        return 2

    percentage = (score / max_score) * 100

    if percentage >= 85:
        return 5
    elif percentage >= 70:
        return 4
    elif percentage >= 50:
        return 3
    else:
        return 2

def test_results(request, id):

```

```

"""Функция для отображения результатов теста."""

try:
    # Получаем информацию о тесте
    test_url = f'{API_BASE_URL}/Tests/{id}'
    response = requests.get(test_url, verify=False)
    test = response.json() if response.status_code == 200 else {}

    user_id = request.session.get('user_id')

# Получаем попытки прохождения теста для текущего пользователя
try:
    attempts_url = f'{API_BASE_URL}/TestAttempts'
    response_attempts = requests.get(attempts_url, verify=False)

    if response_attempts.status_code == 200:
        all_attempts = response_attempts.json()

        # Фильтруем попытки только для текущего пользователя и теста
        attempts = [
            attempt for attempt in all_attempts
            if str(attempt.get('testId')) == str(id)
            and str(attempt.get('userId')) == str(user_id)
        ]

        print(f"DEBUG: Найдено {len(attempts)} попыток для теста {id} и пользователя {user_id}")

# Сортируем попытки по времени окончания (от новых к старым)
attempts.sort(
    key=lambda x: x.get('endTime', '0'),
    reverse=True
)

# Берем последнюю попытку ( первую в отсортированном списке )
latest_attempt = attempts[0] if attempts else None

attempts_count = len(attempts)
max_attempts = 3
remaining_attempts = max(0, max_attempts - attempts_count)

# Определяем, может ли пользователь пересдать тест
can_retake = True

if latest_attempt:

```

```

try:
    grade = float(latest_attempt.get('grade', 0))
    if grade >= 3:
        can_retake = False
    else:
        can_retake = remaining_attempts > 0
except (ValueError, TypeError):
    can_retake = remaining_attempts > 0

else:
    attempts = []
    latest_attempt = None
    attempts_count = 0
    remaining_attempts = 3
    can_retake = True

except Exception as e:
    print(f"Ошибка при получении попыток: {e}")
    attempts = []
    latest_attempt = None
    attempts_count = 0
    remaining_attempts = 3
    can_retake = True

# Получаем критерии оценивания только для текущего теста
try:
    criteria_url = f'{API_BASE_URL}/GradingCriterions'
    response_criteria = requests.get(criteria_url, verify=False)

    if response_criteria.status_code == 200:
        all_criteria = response_criteria.json()
        # Фильтруем критерии только для текущего теста
        grading_criteria = [
            criterion for criterion in all_criteria
            if str(criterion.get('idTest')) == str(id)
        ]
        # Сортируем по оценке (от высшей к низшей)
        grading_criteria.sort(key=lambda x: float(x.get('grade', 0)), reverse=True)

        print(f"DEBUG: Найдено {len(grading_criteria)} критериев для теста {id}")
    else:
        grading_criteria = []

```

```

        print(f"DEBUG: Ошибка при получении критериев: {response_criteria.status_code}")
    except Exception as e:
        print(f"DEBUG: Ошибка при обработке критериев: {e}")
        grading_criteria = []

    return render(request, 'SubjectInfo/TestResults.html', {
        'test': test,
        'attempts': attempts,
        'latest_attempt': latest_attempt,
        'can_retake': can_retake,
        'remaining_attempts': remaining_attempts,
        'grading_criteria': grading_criteria, # Отфильтрованные критерии
        'attempts_count': attempts_count,
        'max_attempts': max_attempts
    })
}

except Exception as e:
    print(f"Ошибка при получении результатов теста: {e}")
    messages.error(request, 'Произошла ошибка при получении результатов теста')
    return redirect('test_material', id=id)

def test_attempts(request, id):
    """
    Функция для отображения всех попыток прохождения теста для преподавателя.
    """

    # Получаем информацию о тесте
    test_url = f'{API_BASE_URL}/Tests/{id}'

    try:
        response = requests.get(test_url, verify=False)
        test = response.json() if response.status_code == 200 else {}
    except:
        messages.error(request, 'Не удалось загрузить информацию о тесте.')
        return redirect('SubjectPage', id=1)

    # Получаем все попытки прохождения теста
    attempts_url = f'{API_BASE_URL}/TestAttempts/test/{id}'
    try:
        response_attempts = requests.get(attempts_url, verify=False)
        attempts = response_attempts.json() if response_attempts.status_code == 200 else []
    except:
        attempts = []

```

```

# Группируем попытки по пользователям
user_attempts = {}

for attempt in attempts:
    user_id = attempt['userId']

    if user_id not in user_attempts:
        # Получаем информацию о пользователе
        user_url = f'{API_BASE_URL}/Users/{user_id}'

        try:
            response_user = requests.get(user_url, verify=False)
            user = response_user.json() if response_user.status_code == 200 else {}
            full_name = f'{user.get('firstName', 'Не указано')} {user.get('secondName', 'Не указано')}{user.get('middleName', 'Не указано")}'
        except:
            full_name = 'Неизвестный пользователь'

        user_attempts[user_id] = {
            'full_name': full_name,
            'attempts': []
        }

    user_attempts[user_id]['attempts'].append(attempt)

# Для каждого пользователя сортируем попытки по дате и времени
for user_id in user_attempts:
    user_attempts[user_id]['attempts'].sort(key=lambda x: (x['dateTestAttempt'], x['timeTestAttempt']), reverse=True)

return render(request, 'SubjectInfo/TestAttempts.html', {
    'test': test,
    'user_attempts': user_attempts
})
}

def test_students_results(request, id):
    """
    Функция для отображения результатов теста всех студентов для преподавателя.
    """

    # Получаем информацию о тесте
    test_url = f'{API_BASE_URL}/Tests/{id}'

    try:
        response = requests.get(test_url, verify=False)
        test = response.json() if response.status_code == 200 else {}
    
```

```

subject_id = test.get('subjectId')

except:
    messages.error(request, 'Не удалось загрузить информацию о тесте.')
    return redirect('SubjectPage', id=1)

# Получаем информацию о предмете
subject_url = f'{API_BASE_URL}/Subjects/{subject_id}'

try:
    response_subject = requests.get(subject_url, verify=False)
    subject = response_subject.json() if response_subject.status_code == 200 else {}
    group_id = subject.get('idGroup')
except:
    group_id = None

# Получаем студентов группы
students = []
if group_id:
    students_url = f'{API_BASE_URL}/Users'
    try:
        response_students = requests.get(students_url, verify=False)
        all_users = response_students.json() if response_students.status_code == 200 else []
        # Фильтруем только студентов (role_id = 3) из нужной группы
        students = [user for user in all_users if user.get('idRole') == 3 and user.get('idGroup') == group_id]
    except:
        students = []

# Получаем все попытки прохождения теста
attempts_url = f'{API_BASE_URL}/TestAttempts'

try:
    response_attempts = requests.get(attempts_url, verify=False)
    all_attempts = response_attempts.json() if response_attempts.status_code == 200 else []

    print(f'Получено попыток: {len(all_attempts)}')
    print(f'ID теста для фильтрации: {id}')

    # Фильтруем попытки только для текущего теста
    test_attempts = [
        attempt for attempt in all_attempts
        if str(attempt.get('testId')) == str(id)
    ]

    print(f'Отфильтровано попыток для теста {id}: {len(test_attempts)}')

```

```

except Exception as e:
    print(f"Ошибка при получении попыток: {e}")
    test_attempts = []

# Для каждого студента находим его попытки
for student in students:
    student_id = student['idUser']

    # Фильтруем попытки только для текущего студента и текущего теста
    student_attempts = [
        attempt for attempt in test_attempts
        if str(attempt.get('userId')) == str(student_id)
    ]

    if student_attempts:
        # Сортируем попытки по дате и времени (от новых к старым)
        student_attempts.sort(
            key=lambda x: (x.get('endTime', '') or ''),
            reverse=True
        )

        # Обрабатываем каждую попытку
        for attempt in student_attempts:
            if 'endTime' in attempt:
                try:
                    # Разбираем строку ISO datetime
                    end_time = attempt['endTime'].split('T')
                    attempt['dateTestAttempt'] = end_time[0] # Дата
                    attempt['timeTestAttempt'] = end_time[1].split('.')[0] # Время
                except (IndexError, AttributeError):
                    attempt['dateTestAttempt'] = 'Нет данных'
                    attempt['timeTestAttempt'] = 'Нет данных'

            else:
                attempt['dateTestAttempt'] = 'Нет данных'
                attempt['timeTestAttempt'] = 'Нет данных'

        # Находим лучшую попытку (с максимальным баллом)
        best_attempt = max(student_attempts, key=lambda x: float(x.get('score', 0)))
        student['best_attempt'] = best_attempt
        student['all_attempts'] = student_attempts
        student['attempts_count'] = len(student_attempts)
        student['has_passed'] = float(best_attempt.get('grade', 0)) >= 3

    else:

```

```

student['best_attempt'] = None
student['all_attempts'] = []
student['attempts_count'] = 0
student['has_passed'] = False

# Статистика
total_students = len(students)
passed_students = sum(1 for student in students if student['has_passed'])
not_passed_students = total_students - passed_students

context = {
    'test': test,
    'students': students,
    'total_students': total_students,
    'passed_students': passed_students,
    'not_passed_students': not_passed_students,
    'subject': subject
}

return render(request, 'SubjectInfo/TestStudentsResults.html', context)

```

```

def edit_test(request, id):
    if request.method == 'POST':
        try:
            # Получаем данные из формы
            test_data = {
                'idTest': id,
                'nameTest': request.POST.get('nameTest'),
                'descriptionTest': request.POST.get('descriptionTest'),
                'timeLimit': int(request.POST.get('timeLimit')),
                'subjectId': int(request.POST.get('subjectId'))
            }

            # Обновляем тест
            test_url = f'{API_BASE_URL}/Tests/{id}'
            response = requests.put(test_url, json=test_data, verify=False)

            if response.status_code not in [200, 204]:
                messages.error(request, 'Ошибка при обновлении теста')
                return redirect('edit_test', id=id)
        
```

```

# Обновляем критерии оценивания
criteria_url = f'{API_BASE_URL}/GradingCriterions'

# Получаем существующие критерии
existing_criteria_response = make_api_request('GradingCriterions')
if existing_criteria_response['status_code'] == 200:
    existing_criteria = [
        c for c in existing_criteria_response['data']
        if c.get('idTest') == int(id)
    ]

# Удаляем существующие критерии
for criterion in existing_criteria:
    # Используем правильное имя поля idCriteria
    delete_url = f'{API_BASE_URL}/GradingCriterions/{criterion["idCriteria"]}'
    requests.delete(delete_url, verify=False)

# Создаем новые критерии без поля idCriteria
new_criteria = [
    {
        'idTest': id,
        'grade': 5,
        'minPoints': int(request.POST.get('grade5Min')),
        'maxPoints': int(request.POST.get('grade5Max'))
    },
    {
        'idTest': id,
        'grade': 4,
        'minPoints': int(request.POST.get('grade4Min')),
        'maxPoints': int(request.POST.get('grade4Max'))
    },
    {
        'idTest': id,
        'grade': 3,
        'minPoints': int(request.POST.get('grade3Min')),
        'maxPoints': int(request.POST.get('grade3Max'))
    }
]

# Отправляем новые критерии
for criterion in new_criteria:
    response = requests.post(criteria_url, json=criterion, verify=False)

```

```

        if response.status_code not in [200, 201]:
            print(f"Ошибка при создании критерия: {response.status_code}")
            print(f"Отправленные данные: {criterion}")
            print(f"Ответ сервера: {response.text}")
            messages.warning(request, f'Ошибка при обновлении критерия для оценки
{criterion["grade"]}')

# Остальной код для обработки вопросов остается без изменений...

messages.success(request, 'Тест успешно обновлен')
return redirect('test_material', id=id)

except Exception as e:
    print(f"Ошибка при сохранении теста: {str(e)}")
    messages.error(request, f'Ошибка при сохранении теста: {str(e)}')
    return redirect('edit_test', id=id)

# GET запрос
try:
    # Получаем данные о тесте
    test_response = make_api_request(f'Tests/{id}')
    if test_response['status_code'] != 200:
        messages.error(request, 'Тест не найден')
        return redirect('index')

    test = test_response['data']

    # Получаем вопросы теста используя общую функцию
    questions = get_test_questions(id)

    # Получаем критерии оценивания
    criteria_response = make_api_request('GradingCriterions')
    if criteria_response['status_code'] == 200:
        # Фильтруем критерии только для текущего теста
        grading_criteria = [
            c for c in criteria_response['data']
            if c.get('idTest') == int(id)
        ]
        # Сортируем по оценке (от высшей к низшей)
        grading_criteria.sort(key=lambda x: float(x.get('grade', 0)), reverse=True)
    else:
        grading_criteria = []

```

```

context = {
    'test': test,
    'questions': questions,
    'grading_criteria': grading_criteria
}

return render(request, 'SubjectInfo/EditTest.html', context)

except Exception as e:
    print(f"Ошибка при загрузке теста: {str(e)}")
    messages.error(request, f'Ошибка при загрузке теста: {str(e)}')
    return redirect('index')

def delete_test(request, id):
    """
    Функция для удаления теста.
    """

    if request.method == 'POST':
        try:
            # Получаем информацию о тесте для получения subject_id
            test_url = f'{API_BASE_URL}/Tests/{id}'
            response = requests.get(test_url, verify=False)

            if response.status_code != 200:
                messages.error(request, 'Тест не найден')
                return redirect('AllSubjectPage')

            test = response.json()
            subject_id = test.get('subjectId')

            # Сначала удаляем все попытки прохождения теста
            attempts_url = f'{API_BASE_URL}/TestAttempts'
            attempts_response = requests.get(attempts_url, verify=False)
            if attempts_response.status_code == 200:
                attempts = attempts_response.json()
                # Фильтруем попытки для данного теста
                test_attempts = [a for a in attempts if str(a.get('testId')) == str(id)]

                # Удаляем каждую попытку
                for attempt in test_attempts:
                    attempt_id = attempt.get('idTestAttempt')

```

```

        delete_attempt_url = f'{API_BASE_URL}/TestAttempts/{attempt_id}'
        requests.delete(delete_attempt_url, verify=False)

        # Удаляем все вопросы теста
        questions_url = f'{API_BASE_URL}/QuestionsTests'
        questions_response = requests.get(questions_url, verify=False)
        if questions_response.status_code == 200:
            questions = questions_response.json()
            # Фильтруем вопросы для данного теста
            test_questions = [q for q in questions if str(q.get('idTest')) == str(id)]

            # Удаляем каждый вопрос
            for question in test_questions:
                question_id = question.get('idQuestion')
                delete_question_url = f'{API_BASE_URL}/QuestionsTests/{question_id}'
                requests.delete(delete_question_url, verify=False)

            # Удаляем критерии оценивания
            criteria_url = f'{API_BASE_URL}/GradingCriterions'
            criteria_response = requests.get(criteria_url, params={'testId': id}, verify=False)
            if criteria_response.status_code == 200:
                criteria = criteria_response.json()
                # Фильтруем критерии для данного теста
                test_criteria = [c for c in criteria if str(c.get('testId')) == str(id)]

                # Удаляем каждый критерий
                for criterion in test_criteria:
                    criterion_id = criterion.get('idGradingCriterion')
                    delete_criterion_url = f'{API_BASE_URL}/GradingCriterions/{criterion_id}'
                    requests.delete(delete_criterion_url, verify=False)

            # Наконец, удаляем сам тест
            response = requests.delete(test_url, verify=False)
            if response.status_code == 204:
                messages.success(request, 'Тест успешно удален')
                return redirect('SubjectPage', id=subject_id)
            else:
                messages.error(request, 'Ошибка при удалении теста')
                return redirect('test_material', id=id)

        except Exception as e:
            messages.error(request, f'Ошибка при удалении теста: {str(e)}')

```

```

        return redirect('AllSubjectPage')

    return HttpResponse(status=405) # Method Not Allowed

def check_api_access(request):
    """
    Функция для проверки соединения с API
    """

    try:
        # Проверяем доступ к основным эндпоинтам
        users_response = make_api_request('Users')
        subjects_response = make_api_request('Subjects')
        groups_response = make_api_request('Groups')

        status = {
            'users': {
                'status': users_response.status_code,
                'message': 'OK' if users_response.status_code == 200 else 'Failed'
            },
            'subjects': {
                'status': subjects_response.status_code,
                'message': 'OK' if subjects_response.status_code == 200 else 'Failed'
            },
            'groups': {
                'status': groups_response.status_code,
                'message': 'OK' if groups_response.status_code == 200 else 'Failed'
            }
        }

        return JsonResponse(status)
    except Exception as e:
        return JsonResponse({'error': str(e)}, status=500)

def ensure_session_valid(request):
    """
    Функция для проверки валидности сессии и продления времени жизни
    """

    if 'user_id' not in request.session:
        return False

    # Обновляем время последней активности
    request.session['last_activity'] = str(datetime.now())

```

```

# Устанавливаем флаг модификации сессии
request.session.modified = True
# Принудительно сохраняем сессию
request.session.save()

return True

def test_submit(request, test_id):
    """
    Функция для отправки результатов теста.
    """

    import json

    # Подробное логирование полученных данных
    print("\n===== ДАННЫЕ ФОРМЫ =====")
    for key, value in request.POST.items():
        print(f"POST: {key} = {value}")
    print("=====\\n")

    # Анализируем данные сопоставления и упорядочивания
    matching_questions = {}
    order_questions = {}

    for key in request.POST.keys():
        # Поиск данных сопоставления
        if '_matching_' in key:
            parts = key.split('_matching_')
            question_id = parts[0].replace('question_', '')
            if question_id not in matching_questions:
                matching_questions[question_id] = []
            matching_questions[question_id].append(request.POST[key])

        # Поиск данных упорядочивания
        elif '_order_' in key:
            parts = key.split('_order_')
            question_id = parts[0].replace('question_', '')
            if question_id not in order_questions:
                order_questions[question_id] = []
            order_questions[question_id].append(request.POST[key])

    print("Найденные данные сопоставления:")
    for question_id, values in matching_questions.items():

```

```

print(f"Вопрос {question_id}: {values}")

print("Найденные данные упорядочивания:")
for question_id, values in order_questions.items():
    print(f"Вопрос {question_id}: {values}")

def check_single_choice(request, question_id, question):
    """Проверка вопроса с одним правильным ответом"""
    print("\nОтладка вопроса с одним ответом:")
    print(f"Весь вопрос: {question}") # Добавляем вывод всего вопроса

    user_answer = request.POST.get(f'question_{question_id}')
    correct_answer = question.get('correctAnswer')
    answer_variants = question.get('answerVariants', '').split(';')

    print(f"ID вопроса: {question_id}")
    print(f"Варианты ответов: {answer_variants}")
    print(f"Полученный ответ: {user_answer}")
    print(f"Правильный ответ: {correct_answer}")

    if not user_answer or not correct_answer:
        print("Ошибка: Отсутствует ответ пользователя или правильный ответ")
        return 0

    # Нормализуем ответы перед сравнением
    user_answer = user_answer.strip().lower()
    correct_answer = correct_answer.strip().lower()

    is_correct = user_answer == correct_answer
    points = question.get('points', 0) if is_correct else 0

    print(f"Результат проверки: {'Верно' if is_correct else 'Неверно'}")
    print(f"Начислено баллов: {points}")

    return points

def check_multiple_choice(request, question_id, question):
    """Проверка вопроса с множественным выбором"""
    print(f"\nПроверка вопроса с множественным выбором (ID: {question_id}):")
    print(f"Текст вопроса: {question.get('questionText')}")

    # Получаем все возможные варианты ответов

```

```

answer_variants = question.get('answerVariants', '').split(';')
answer_variants = [v.strip() for v in answer_variants if v.strip()]

# Получаем правильные ответы и нормализуем их
correct_answers = question.get('correctAnswer', '').split(';')
correct_answers = [ans.strip() for ans in correct_answers if ans.strip()]

# Получаем ответы пользователя
user_answers = []
for i in range(len(answer_variants)):
    answer_key = f'question_{question_id}_option_{i+1}'
    if answer_key in request.POST:
        answer = request.POST[answer_key].strip()
        if answer:
            user_answers.append(answer)

print(f"Варианты ответов: {answer_variants}")
print(f"Правильные ответы (до нормализации): {correct_answers}")
print(f"Ответы пользователя (до нормализации): {user_answers}")

# Функция для нормализации строки
def normalize_string(s):
    # Приводим к нижнему регистру
    s = s.lower()
    # Убираем начальные и конечные пробелы
    s = s.strip()
    # Заменяем множественные пробелы на один
    s = ' '.join(s.split())
    return s

# Нормализуем оба набора ответов
normalized_correct = {normalize_string(ans) for ans in correct_answers}
normalized_user = {normalize_string(ans) for ans in user_answers}

print(f"Нормализованные правильные ответы: {normalized_correct}")
print(f"Нормализованные ответы пользователя: {normalized_user}")

# Сравниваем множества
is_correct = normalized_correct == normalized_user
points = question.get('points', 0) if is_correct else 0

print(f"Результат проверки: {'Верно' if is_correct else 'Неверно'}")

```

```
print(f"Начислено баллов: {points}")
```

```
return points
```

```
def check_text_input(request, question_id, question):
```

```
    """Проверка вопроса с вводом ответа"""
    user_answer = request.POST.get(f'question_{question_id}', "").strip()
```

```
    correct_answer = question.get('correctAnswer', "").strip()
```

```
    print(f"Вопрос {question_id}:")
```

```
    print(f"- Введенный ответ: '{user_answer}'")
```

```
    print(f"- Правильный ответ: '{correct_answer}'")
```

```
    # Нормализация ответов для сравнения
```

```
    user_normalized = user_answer.lower().replace(' ', "")
```

```
    correct_normalized = correct_answer.lower().replace(' ', "")
```

```
    return question.get('points', 0) if user_normalized == correct_normalized else 0
```

```
def check_matching(request, question_id, question):
```

```
    """Проверка вопроса на сопоставление"""
    print(f"\nПроверка вопроса на сопоставление (ID: {question_id}):")
```

```
    print(f"Текст вопроса: {question.get('questionText')}")
```

```
    # Получаем правильные пары из correctAnswer
```

```
    correct_pairs = {}
```

```
    if question.get('correctAnswer'):
```

```
        pairs = question.get('correctAnswer').split(';')
```

```
        for pair in pairs:
```

```
            if ' - ' in pair:
```

```
                left, right = pair.split(' - ', 1)
```

```
                correct_pairs[left.strip().lower()] = right.strip().lower()
```

```
    print(f"Правильные пары: {correct_pairs}")
```

```
    # Получаем ответы пользователя
```

```
    user_pairs = {}
```

```
    i = 0
```

```
    while True:
```

```
        left_key = f'question_{question_id}_left_{i}'
```

```
        match_key = f'question_{question_id}_match_{i}'
```

```

if left_key not in request.POST:
    break

left = request.POST.get(left_key, "").strip().lower()
match = request.POST.get(match_key, "").strip()

if left and match:
    if ':' in match: # Если формат "left:right"
        _, right = match.split(':', 1)
        user_pairs[left] = right.strip().lower()
    else:
        user_pairs[left] = match.lower()

i += 1

print(f'Пары пользователя: {user_pairs}')

# Проверяем все соединения
is_correct = True
for left, right in user_pairs.items():
    if left not in correct_pairs or correct_pairs[left] != right:
        is_correct = False
        break

# Проверяем, что количество пар совпадает
if len(user_pairs) != len(correct_pairs):
    is_correct = False

points = question.get('points', 0) if is_correct else 0

print(f"Результат проверки: {'Верно' if is_correct else 'Неверно'}")
print(f"Начислено баллов: {points}")

return points

def check_ordering(request, question_id, question):
    """Проверка вопроса на установление порядка"""
    print(f"\nПроверка вопроса на установление порядка (ID: {question_id}):")
    print(f"Весь вопрос: {question}") # Добавляем вывод всего вопроса
    print(f"Текст вопроса: {question.get('questionText')}")

    # Получаем правильный порядок

```

```

correct_order = []
if question.get('correctAnswer'):
    correct_order = [item.strip() for item in question.get('correctAnswer').split(';') if item.strip()]

print(f"Правильный порядок: {correct_order}")

# Получаем все POST параметры для отладки
print("Все POST параметры для этого вопроса:")
for key, value in request.POST.items():
    if f'question_{question_id}' in key:
        print(f'{key}: {value}')

# Получаем ответ пользователя
user_order = []
i = 0
while True:
    key = f'question_{question_id}_order_{i}'
    if key not in request.POST:
        break
    value = request.POST.get(key).strip()
    if value:
        user_order.append(value)
    i += 1

print(f"Порядок пользователя: {user_order}")

# Проверяем совпадение порядка
is_correct = user_order == correct_order
points = question.get('points', 0) if is_correct else 0

print(f"Результат проверки: {'Верно' if is_correct else 'Неверно'}")
print(f"Начислено баллов: {points}")

return points

```

## ПРИЛОЖЕНИЕ В. Программа испытаний

### АННОТАЦИЯ

В данном программном документе приведены сценарии тестовых испытаний и результаты тестовых испытаний.

В разделе «Цель испытаний» указана цель проведения испытаний.

В разделе «Объект испытаний» указаны наименование работы и цель её выполнения.

В разделе «Требования к тестированию» указаны функциональные возможности, подлежащие проверке во время испытаний и заданные в пояснительной записке на программу.

В разделе «Средства и порядок испытаний» указаны технические и программные средства, используемые во время испытаний, а также порядок проведения испытаний.

В разделе «Методы испытаний» приведено описание используемых методов испытаний.

В разделе «Тестовые примеры и результаты» приведены таблицы с тестовыми сценариями и результатами тестовых испытаний.

## СОДЕРЖАНИЕ

ЦЕЛЬ ИСПЫТАНИЙ.....	4
ОБЪЕКТ ИСПЫТАНИЙ .....	5
1.1. Наименование объекта .....	5
1.2. Область применения объекта .....	5
1.3. Обозначение испытуемой программы.....	5
ТРЕБОВАНИЯ К ТЕСТИРОВАНИЮ .....	6
1.4. Схема тестирования.....	6
1.5. Требования, подлежащие проверке .....	6
СРЕДСТВА И ПОРЯДОК ИСПЫТАНИЙ .....	8
МЕТОДЫ ИСПЫТАНИЙ .....	9
ТЕСТОВЫЕ ПРИМЕРЫ И РЕЗУЛЬТАТЫ.....	10
1.6. Возможности интерфейса .....	10
1.7. Возможности неавторизованного пользователя.....	13
1.7.1. Возможность авторизации .....	13
1.7.2. Возможность регистрации.....	14
1.8. Администратор системы .....	1
1.8.1. Возможность добавления новой группы.....	1
1.8.2. Возможность изменения группы.....	1
1.8.3. Возможность изменения пользователя.....	2
1.8.4. Возможность добавления предмета .....	3
1.8.5. Возможность изменения предмета .....	4
1.9. Преподаватель.....	5
1.9.1. Возможность добавления новой лекции .....	5
1.9.2. Возможность изменения лекции .....	7

1.9.3. Возможность добавления новой практической работы.....	9
1.9.4. Возможность добавления оценки к практической работе....	11
1.9.5. Возможность создания теста .....	12
1.9.6. Возможность добавления вопросов в тест .....	16
1.9.7. Возможность изменения теста .....	29
1.9.8. Возможность изменения вопросов в тесте.....	33
1.10. Студент .....	47
1.10.1. Возможность сдать практическую работу .....	47
1.10.2. Возможность пройти тест .....	48

## **ЦЕЛЬ ИСПЫТАНИЙ**

Целью проведения испытаний является проверка соответствия разработанного программного изделия требованиям, изложенным в документе «Пояснительная записка» для разработки приложения «YumlSchool».

## **ОБЪЕКТ ИСПЫТАНИЙ**

### **1.1. Наименование объекта**

Наименование - «Веб приложение YumlSchool».

### **1.2. Область применения объекта**

Приложение предназначено к использованию в учебных заведениях. Веб-приложение позволит оптимизировать учебный процесс и автоматизировать проведение практических работ и тестирования студентов по изученным темам.

### **1.3. Обозначение испытуемой программы**

Обозначение – «YumlSchool».

## ТРЕБОВАНИЯ К ТЕСТИРОВАНИЮ

### 1.4. Схема тестирования

Тестирование разработанного программного решения проводилось по схеме тестирования, представленного на рисунке 1.

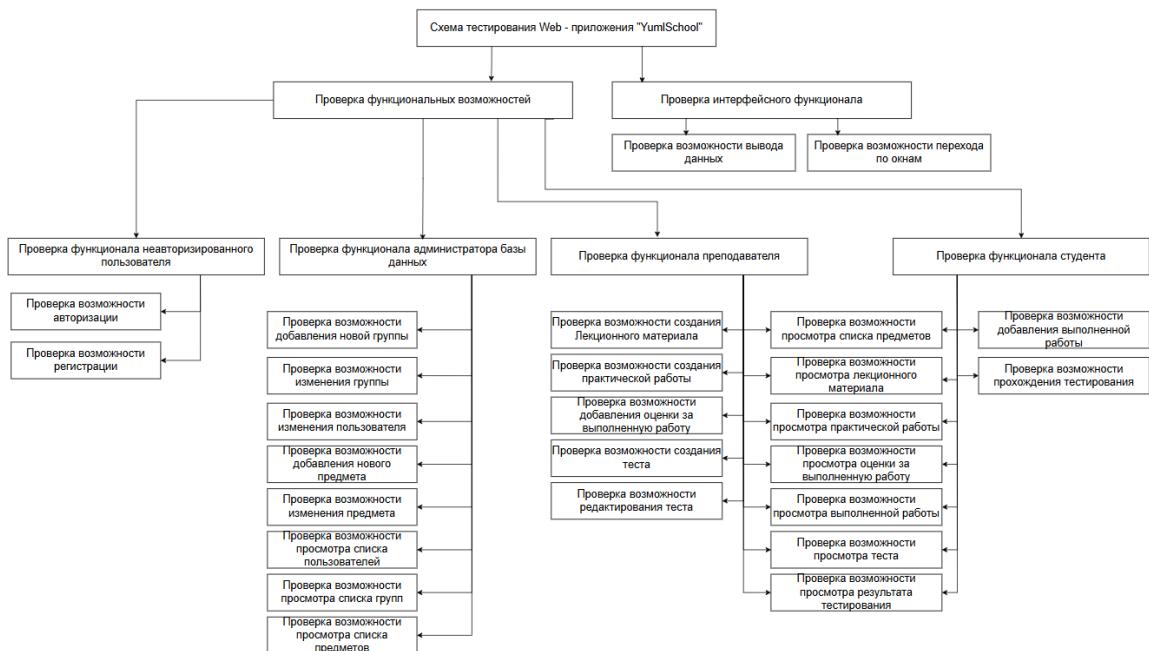


Рисунок 1 - Схема тестирования

### 1.5. Требования, подлежащие проверке

Функциональные возможности, подлежащие проверке при проведении тестирования:

Функциональные возможности интерфейса: Проверка возможности вывода данных, Проверка возможности перехода по страницам

Функциональные возможности неавторизированного пользователя: Проверка возможности авторизации, регистрации.

Функциональные возможности администратора базы данных: Проверка возможности добавления новой группы, Проверка возможности изменения группы, Проверка возможности изменения пользователя, Проверка возможности добавления нового предмета, Проверка возможности изменения предмета, Проверка возможности просмотра списка пользователей, групп, предметов.

Функциональные возможности преподавателя: Проверка возможности создания Лекционного материала, Проверка возможности создания практической работы, Проверка возможности добавления оценки за выполненную работу, Проверка возможности просмотра списка предметов, Проверка возможности просмотра лекционного материала, Проверка возможности просмотра практической работы, Проверка возможности просмотра оценки за выполненную работу, Проверка возможности просмотра выполненной работы студента, Проверка возможности создания теста, Проверка возможности редактирования теста, Проверка возможности просмотра теста, Проверка возможности просмотра результата тестирования.

Функциональные возможности студента: Проверка возможности просмотра списка предметов, Проверка возможности просмотра лекционного материала, Проверка возможности просмотра практической работы, Проверка возможности просмотра оценки, Проверка возможности добавления выполненной работы, Проверка возможности просмотра выполненной работы, Проверка возможности прохождения тестирования, Проверка возможности просмотра результата прохождения теста

## СРЕДСТВА И ПОРЯДОК ИСПЫТАНИЙ

В качестве средств вычислительной техники при разработке ПО и для использования веб приложения использовался ноутбук Asus Zenbook 14 UM433IQ. Характеристики представлены в таблице 5.

Таблица 1 - Характеристики вычислительной техники для разработки ПО

№	Тип средства	Название средства
1	2	3
Для разработки		
Ноутбук Asus Zenbook 14 UM433IQ		
1	Размер экрана:	14
2	Разрешение экрана:	1920x1080
3	Линейка процессора:	AMD Ryzen 7 4700U with Radeon Graphics
4	Количество ядер процессора:	8
5	Оперативная память:	16 ГБ
6	Видеокарта:	NVIDIA GeForce MX350
7	Конфигурация накопителей:	SSD
8	Общий объем всех накопителей:	1000

## МЕТОДЫ ИСПЫТАНИЙ

### 1. По формальности тестирования.

Тестирование по тестам – тестирование по предварительно написанным тест-кейсам.

### 2. По исполнению кода.

Динамическое тестирование - во время тестирования код исполняется.

### 3. По уровню тестирования.

Системное – проверка работы всей системы на соответствие заявленным требованиям к программному продукту.

### 4. По целям.

Функциональное тестирование направлено на проверку того, какие функции ПО реализованы, и того, насколько верно они реализованы.

### 5. По степени автоматизации.

Ручное – без использования дополнительных программных средств.

### 6. По позитивности сценария.

Позитивным – проверка ПО на соответствие ожидаемому поведению. Негативным – проверяет, будет ли ПО работать в случае, когда поведение пользователя отличается от ожидаемого.

### 7. По знанию системы.

Тестирование «белого ящика» – тестирование программного продукта с доступом к коду.

Тестирование «черного ящика» – тестирование без доступа к коду продукта.

### 8. По разработке тестовых испытаний.

На основе требований – требование было определено до начала тестирования.

## ТЕСТОВЫЕ ПРИМЕРЫ И РЕЗУЛЬТАТЫ

Все тестовые данные, действия и результаты для проведения тестирования Веб приложения «YumlSchool» представлены в таблицах:

### 1.6. Возможности интерфейса

Таблица 2 - Тест возможности переключения между окнами и отображения данных

№	Окно	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
1	Окно авторизации	Доступно неавторизованному пользователю	Доступно неавторизованному пользователю	Тест пройден
2	Окно регистрации	Доступно неавторизованному пользователю	Доступно неавторизованному пользователю	Тест пройден
4	Окно предметов	Доступно после успешного прохождения авторизации под ролью «Студент».	Доступно после успешного прохождения авторизации под ролью «Студент».	Тест пройден
5		Доступно после успешного прохождения авторизации под ролью «Преподаватель».	Доступно после успешного прохождения авторизации под ролью «Преподаватель».	Тест пройден
6	Окна административной панели	Доступно после успешного прохождения авторизации под ролью «Администратор базы данных».	Доступно после успешного прохождения авторизации под ролью «Администратор базы данных».	Тест пройден
7	Окно предмета	Доступно из окна «Предметы» авторизированного пользователя под ролью «Студент» и «Преподаватель»	Доступно из окна «Предметы» авторизированного пользователя под ролью «Студент» и «Преподаватель»	Тест пройден

№	Окно	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
8	Окно практической работы	Доступно из окна «Предмет» авторизированного пользователя под ролью «Студент»	Доступно из окна «Предмет» авторизированного пользователя под ролью «Студент»	Тест пройден
9		Доступно из окна «Предмет» авторизированного пользователя под ролью «Преподаватель»	Доступно из окна «Предмет» авторизированного пользователя под ролью «Преподаватель»	Тест пройден
10	Окно создания практической работы	Доступно из окна «Предмет» авторизированного пользователя под ролью «Преподаватель»	Доступно из окна «Предмет» авторизированного пользователя под ролью «Преподаватель»	Тест пройден
11	Окно лекционного материала	Доступно из окна «Предмет» авторизированного пользователя под ролью «Преподаватель»	Доступно из окна «Предмет» авторизированного пользователя под ролью «Преподаватель»	Тест пройден
12		Доступно из окна «Предмет» авторизированного пользователя под ролью «Студент»	Доступно из окна «Предмет» авторизированного пользователя под ролью «Студент»	Тест пройден
13	Окно редактирования лекционного материала	Доступно из окна «Лекция» авторизированного пользователя под ролью «Преподаватель»	Доступно из окна «Лекция» авторизированного пользователя под ролью «Преподаватель»	Тест пройден
14	Окно создания лекционного материала	Доступно из окна «Предмет» авторизированного пользователя под ролью «Преподаватель»	Доступно из окна «Предмет» авторизированного пользователя под ролью «Преподаватель»	Тест пройден

№	Окно	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
15	Окно создания теста	Доступно из окна «Предмет» авторизированного пользователя под ролью «Преподаватель»	Доступно из окна «Предмет» авторизированного пользователя под ролью «Преподаватель»	Тест пройден
16	Окно редактирования теста	Доступно из окна «Тест» авторизированного пользователя под ролью «Преподаватель»	Доступно из окна «Тест» авторизированного пользователя под ролью «Преподаватель»	Тест пройден
17	Окно теста	Доступно из окна «Предмет» авторизированного пользователя под ролью «Преподаватель»	Доступно из окна «Предмет» авторизированного пользователя под ролью «Преподаватель»	Тест пройден
18		Доступно из окна «Предмет» авторизированного пользователя под ролью «Студент»	Доступно из окна «Предмет» авторизированного пользователя под ролью «Студент»	Тест пройден
19	Окно результата теста	Доступно из окна «Предмет» авторизированного пользователя под ролью «Преподаватель»	Доступно из окна «Предмет» авторизированного пользователя под ролью «Преподаватель»	Тест пройден
20		Доступно из окна «Предмет» авторизированного пользователя под ролью «Студент»	Доступно из окна «Предмет» авторизированного пользователя под ролью «Студент»	Тест пройден
22	Окно прохождения теста	Доступно из окна «Предмет» авторизированного пользователя под ролью «Студент»	Доступно из окна «Предмет» авторизированного пользователя под ролью «Студент»	Тест пройден

## 1.7. Возможности неавторизованного пользователя.

### 1.7.1. Возможность авторизации

Таблица 3 - Тест возможности авторизации

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/н е пройден)
1	2	3	4	5
1	Логин: vasin3485 Пароль: vasim4ek	Успешная авторизация	Успешная авторизация	Тест пройден
2	Логин: vasin3485 Пароль: 123	Авторизация не прошла. Над полями ввода отображается надпись «Неверный пароль»	Авторизация не прошла. Над полями ввода отображается надпись «Неверный пароль»	Тест пройден
3	Логин: vasin34851 Пароль: vasim4ek	Авторизация не прошла. Над полями ввода отображается надпись «Такого логина не существует»	Авторизация не прошла. Над полями ввода отображается надпись «Такого логина не существует»	Тест пройден
4	Логин: пустота Пароль: пустота	Авторизация не прошла. Над полем ввода логина и пароля отображается сообщение «Заполните это поле.»	Авторизация не прошла. Над полем ввода логина и пароля отображается сообщение «Заполните это поле.»	Тест пройден
5	Логин: vasin3485 Пароль: пустота	Авторизация не прошла. Над полем ввода пароля отображается сообщение «Заполните это поле»	Авторизация не прошла. Над полем ввода пароля отображается сообщение «Заполните это поле»	Тест пройден
6	Логин: пустота Пароль: vasim4ek	Авторизация не прошла. Над полем ввода логина	Авторизация не прошла. Над полем ввода логина отображается сообщение «Заполните это поле.»	Тест Пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/н е пройден)
1	2	3	4	5
		отображается сообщение «Заполните это поле.»		

### 1.7.2. Возможность регистрации.

Таблица 4 - Тест возможности регистрации

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не про <sup>йден</sup> )
1	2	3	4	5
1	Фамилия: Ицков Имя: Дмитрий Отчество: Олегович Email: kaiangel@gmail.com Логин: kaiangel1245 Пароль: lav39mice!	Успешная регистрация	Успешная регистрация	Тест пройден
2	Фамилия: Ицков Имя: Дмитрий Отчество: «Пустое поле» Email: kaiangel@gmail.com Логин: kaiangel1245 Пароль: lav39mice!	Успешная регистрация	Успешная регистрация	Тест пройден
3	Фамилия: «Пустое поле» Имя: Дмитрий Отчество: «Олегович» Email: kaiangel@gmail.com Логин: kaiangel1245 Пароль: lav39mice!	Регистрация не прошла. Над полем ввода фамилии отображается сообщение «Заполните это поле.»	Регистрация не прошла. Над полем ввода фамилии отображается сообщение «Заполните это поле.»	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не прощен)
1	2	3	4	5
4	Фамилия: Ицков Имя: «Пустое поле» Отчество: Олегович Email: kaiangel@gmail.com Логин: kaiangel1245 Пароль: lav39mice!	Регистрация не прошла. Над полем ввода имени отображается сообщение «Заполните это поле.»	Регистрация не прошла. Над полем ввода имени отображается сообщение «Заполните это поле.»	Тест пройден
5	Фамилия: Ицков Имя: Дмитрий Отчество: Олегович Email: «Пустое поле» Логин: kaiangel1245 Пароль: lav39mice!	Регистрация не прошла. Над полем ввода электронной почты отображается сообщение «Заполните это поле.»	Регистрация не прошла. Над полем ввода электронной почты отображается сообщение «Заполните это поле.»	Тест пройден
6	Фамилия: Ицков Имя: Дмитрий Отчество: Олегович Email: kaiangel@gmail.com Логин: «Пустое поле» Пароль: lav39mice!	Регистрация не прошла. Над полем ввода логина отображается сообщение «Заполните это поле.»	Регистрация не прошла. Над полем ввода логина отображается сообщение «Заполните это поле.»	Тест пройден
7	Фамилия: Ицков Имя: Дмитрий Отчество: Олегович Email: kaiangel@gmail.com Логин: kaiangel1245 Пароль: «Пустое поле»	Регистрация не прошла. Над полем ввода пароля отображается сообщение «Заполните это поле.»	Регистрация не прошла. Над полем ввода пароля отображается сообщение «Заполните это поле.»	Тест Пройден
8	Фамилия: Ицков Имя: Дмитрий Отчество: Олегович Email: kaiangel@gmail.com Логин: kai Пароль: lav39mice!	Регистрация не прошла. Над полями ввода отображается сообщение «Логин должен содержать минимум 8 символов.»	Регистрация не прошла. Над полями ввода отображается сообщение «Логин должен содержать минимум 8 символов.»	Тест Пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не прощен)
1	2	3	4	5
		символов.»		
9	Фамилия: Ицков Имя: Дмитрий Отчество: Олегович Email: kaiangel@gmail.com Логин: kai Пароль: lavmice	Регистрация не прошла. Над полями ввода отображается сообщение «Логин должен содержать минимум 8 символов. Пароль должен содержать минимум 8 символов. Пароль должен содержать как минимум одну цифру. Пароль должен содержать как минимум один специальный символ.»	Регистрация не прошла. Над полями ввода отображается сообщение «Логин должен содержать минимум 8 символов. Пароль должен содержать минимум 8 символов. Пароль должен содержать как минимум одну цифру. Пароль должен содержать как минимум один специальный символ.»	Тест пройден
10	Фамилия: Ицков Имя: Дмитрий Отчество: Олегович Email: kaiangel@gmail.com Логин: kaiangel1245 Пароль: lavmice	Регистрация не прошла. Над полями ввода отображается сообщение «Пароль должен содержать минимум 8 символов. Пароль должен содержать как минимум одну цифру. Пароль должен содержать как минимум один специальный символ.»	Регистрация не прошла. Над полями ввода отображается сообщение «Пароль должен содержать минимум 8 символов. Пароль должен содержать как минимум одну цифру. Пароль должен содержать как минимум один специальный символ.»	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не прощен)
1	2	3	4	5
		символ.»		
11	Фамилия: Ицков Имя: Дмитрий Отчество: Олегович Email: kaiangel@gmail.com Логин: kaiangel1245 Пароль: lavemice	Регистрация не прошла. Над полями ввода отображается сообщение «Пароль должен содержать минимум 8 символов. Пароль должен содержать как минимум одну цифру. Пароль должен содержать как минимум один специальный символ.»	Регистрация не прошла. Над полями ввода отображается сообщение «Пароль должен содержать минимум 8 символов. Пароль должен содержать как минимум одну цифру. Пароль должен содержать как минимум один специальный символ.»	Тест пройден
12	Фамилия: Ицков Имя: Дмитрий Отчество: Олегович Email: kaiangel@gmail.com Логин: kaiangel1245 Пароль: lav39mice	Регистрация не прошла. Над полями ввода отображается сообщение «Пароль должен содержать минимум 8 символов. Пароль должен содержать как минимум одну цифру. Пароль должен содержать как минимум один специальный символ.»	Регистрация не прошла. Над полями ввода отображается сообщение «Пароль должен содержать минимум 8 символов. Пароль должен содержать как минимум одну цифру. Пароль должен содержать как минимум один специальный символ.»	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не прощен)
1	2	3	4	5
13	Фамилия: Ицков Имя: Дмитрий Отчество: Олегович Email: lilkristall@gmail.com Логин: kaiangel1245 Пароль: lav39mice!	Регистрация не прошла. Над полями ввода отображается сообщение «Пользователь с таким email уже существует»	Регистрация не прошла. Над полями ввода отображается сообщение «Пользователь с таким email уже существует»	Тест пройден
14	Фамилия: Ицков Имя: Дмитрий Отчество: Олегович Email: lilkristall@gmail.com Логин: ogbuda1234edd Пароль: lav39mice!	Регистрация не прошла. Над полями ввода отображается сообщение «Пользователь с таким логином уже существует»	Регистрация не прошла. Над полями ввода отображается сообщение «Пользователь с таким логином уже существует»	Тест пройден

## 1.8. Администратор системы

### 1.8.1. Возможность добавления новой группы

Таблица 5 - Тест возможности добавления новой группы

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройде н)
1	2	3	4	5
1	Название новой группы: АНГ- 16-06	Успешное добавление новой группы	Успешное добавление новой группы	Тест пройден
2	Название новой группы: «Пустое поле»	Добавление новой группы не прошло. Над полем ввода названия новой группы выводится сообщение «Заполните это поле.»	Добавление новой группы не прошло. Над полем ввода названия новой группы выводится сообщение «Заполните это поле.»	Тест пройден
3	Название новой группы: АНГ- 15-06	Добавление новой группы не прошло. Над полем ввода названия новой группы отображается сообщение «Группа с таким названием уже существует»	Добавление новой группы не прошло. Над полем ввода названия новой группы отображается сообщение «Группа с таким названием уже существует»	Тест пройден

### 1.8.2. Возможность изменения группы

Таблица 6 - Тест возможности изменения группы

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
1	Новое название группы: ПИ- 14-08	Успешное изменение названия группы	Успешное изменение названия группы	Тест пройден
2	Новое название группы: «Пустое поле»	Изменение названия группы не прошло. Над полем ввода	Изменение названия группы не прошло. Над полем ввода	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
		нового названия группы отображается сообщение «Заполните это поле.»	нового названия группы отображается сообщение «Заполните это поле.»	
3	Новое название группы: АНГ- 15-06	Изменение названия группы не прошло. Над полем ввода нового названия группы отображается сообщение «Группа с таким названием уже существует»	Изменение названия группы не прошло. Над полем ввода нового названия группы отображается сообщение «Группа с таким названием уже существует»	Тест пройден

### 1.8.3. Возможность изменения пользователя

Таблица 7 - Тест возможности изменения пользователя

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
1	Фамилия: Ицков Имя: Иван Отчество: Александрович Электронная почта: <a href="mailto:facepublic@gmail.com">facepublic@gmail.com</a> Роль: Студент Группа: ПИ-14-09	Успешное изменение данных пользователя	Успешное изменение данных пользователя	Тест пройден
2	Фамилия: Дремин Имя: Иван Отчество: «Пустое поле» Электронная почта: <a href="mailto:facepublic@gmail.com">facepublic@gmail.com</a> Роль: Студент Группа: ПИ-14-09	Успешное изменение данных пользователя	Успешное изменение данных пользователя	Тест пройден
3	Фамилия: «Пустое поле»	Изменение данных	Изменение данных	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	Имя: Иван Отчество: Александрович Электронная почта: <a href="mailto:facepublic@gmail.com">facepublic@gmail.com</a> Роль: Студент Группа: ПИ-14-09	пользователя не прошло. Над полем ввода фамилии отображается сообщение «Заполните это поле.»	пользователя не прошло. Над полем ввода фамилии отображается сообщение «Заполните это поле.»	
4	Фамилия: Дремин Имя: «Пустое поле» Отчество: Александрович Электронная почта: <a href="mailto:facepublic@gmail.com">facepublic@gmail.com</a> Роль: Студент Группа: ПИ-14-09	Изменение данных пользователя не прошло. Над полем ввода имени отображается сообщение «Заполните это поле.»	Изменение данных пользователя не прошло. Над полем ввода имени отображается сообщение «Заполните это поле.»	Тест пройден
5	Фамилия: Дремин Имя: Иван Отчество: Александрович Электронная почта: «Пустое поле» Роль: Студент Группа: ПИ-14-09	Изменение данных пользователя не прошло. Над полем ввода электронной почты отображается сообщение «Заполните это поле.»	Изменение данных пользователя не прошло. Над полем ввода электронной почты отображается сообщение «Заполните это поле.»	Тест пройден
6	Фамилия: Дремин Имя: Иван Отчество: Александрович Электронная почта: <a href="mailto:kaiangel@gmail.com">kaiangel@gmail.com</a> Роль: Студент Группа: ПИ-14-09	Изменение данных не прошло. Над полями ввода отображается сообщение «Пользовател ь с таким email уже существует»	Изменение данных не прошло. Над полями ввода отображается сообщение «Пользователь с таким email уже существует»	Тест пройден

#### 1.8.4. Возможность добавления предмета

Таблица 8 - Тест возможности добавления пользователя

№	Действие (входные данные)	Ожидаемый результат	Фактически й результат	Статус теста (пройден/не пройден)
---	------------------------------	------------------------	------------------------------	--------------------------------------

1	2	3	4	5
1	Название: SC музыка Группа: Администрация Преподаватель: Горбутова Маргарита Витальевна	Успешное добавление предмета	Успешное добавление предмета	Тест пройден
2	Название: «Пустое поле» Группа: Администрация Преподаватель: Горбутова Маргарита Витальевна	Добавление предмета не прошло. Над полем ввода названия отображается сообщение «Заполните это поле.»	Добавление предмета не прошло. Над полем ввода названия отображается сообщение «Заполните это поле.»	Тест пройден

#### 1.8.5. Возможность изменения предмета

Таблица 9 - Тест возможности изменения предмета

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
1	Название: SC реперы Группа: Администрация Преподаватель: Горбутова Маргарита Витальевна	Успешное изменение предмета	Успешное изменение предмета	Тест пройден
2	Название: «Пустое поле» Группа: Администрация Преподаватель: Горбутова Маргарита Витальевна	Изменение предмета не прошло. Над полем ввода названия отображается сообщение «Заполните это поле.»	Изменение предмета не прошло. Над полем ввода названия отображается сообщение «Заполните это поле.»	Тест пройден

## 1.9. Преподаватель

### 1.9.1. Возможность добавления новой лекции

Таблица 10 - Тест возможности добавления новой лекции

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/ не пройден)
1	2  Название лекции: Феномен madk1d Описание лекции: В рамках лекции разберем феномен успеха исполнителя madk1d Название файла лекции: madk1d Ссылка на файл лекции: <a href="https://docs.google.com/document/d/19sCVEaF6-0WVC0TSLCf4WkcSuM4r8X6a/edit?usp=drive_link&amp;oid=104502585184625130868&amp;rtpof=true&amp;sd=true">https://docs.google.com/document/d/19sCVEaF6-0WVC0TSLCf4WkcSuM4r8X6a/edit?usp=drive_link&amp;oid=104502585184625130868&amp;rtpof=true&amp;sd=true</a>	Успешное добавление новой лекции	Успешное добавление новой лекции	Тест пройден
2	  Название лекции: «Пустое поле» Описание лекции: В рамках лекции разберем феномен успеха исполнителя madk1d Название файла лекции: madk1d Ссылка на файл лекции: <a href="https://docs.google.com/document/d/19sCVEaF6-0WVC0TSLCf4WkcSuM4r8X6a/edit?usp=drive_link&amp;oid=104502585184625130868&amp;rtpof=true&amp;sd=true">https://docs.google.com/document/d/19sCVEaF6-0WVC0TSLCf4WkcSuM4r8X6a/edit?usp=drive_link&amp;oid=104502585184625130868&amp;rtpof=true&amp;sd=true</a>	Добавление новой лекции не прошло. Над полем ввода называния лекции отображается сообщение «Заполните это поле»	Добавление новой лекции не прошло. Над полем ввода называния лекции отображается сообщение «Заполните это поле»	Тест пройден
3	  Название лекции: Феномен madk1d Описание лекции: «Пустое поле» Название файла лекции: madk1d Ссылка на файл лекции: <a href="https://docs.google.com/document/d/19sCVEaF6-0WVC0TSLCf4WkcSuM4r8X6a/edit?usp=drive_link&amp;oid=104502585184625130868&amp;rtpof=true&amp;sd=true">https://docs.google.com/document/d/19sCVEaF6-0WVC0TSLCf4WkcSuM4r8X6a/edit?usp=drive_link&amp;oid=104502585184625130868&amp;rtpof=true&amp;sd=true</a>	Добавление новой лекции не прошло. Над полем ввода описания лекции отображается сообщени	Добавление новой лекции не прошло. Над полем ввода описания лекции отображается сообщени	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/ не пройден)
1	2	3	4	5
		отображается сообщение «Заполните это поле»	«Заполните это поле»	
4	<p>Название лекции: Феномен madk1d</p> <p>Описание лекции: В рамках лекции разберем феномен успеха исполнителя madk1d</p> <p>Название файла лекции: «Пустое поле»</p> <p>Ссылка на файл лекции:</p> <p><a href="https://docs.google.com/document/d/19sCVEaF6-0WVC0TSLCf4WkcSuM4r8X6a/edit?usp=drive_link&amp;ouid=104502585184625130868&amp;rtpof=true&amp;sd=true">https://docs.google.com/document/d/19sCVEaF6-0WVC0TSLCf4WkcSuM4r8X6a/edit?usp=drive_link&amp;ouid=104502585184625130868&amp;rtpof=true&amp;sd=true</a></p>	<p>Добавление новой лекции не прошло. Над полем ввода названия файла отображается сообщение «Заполните это поле»</p>	<p>Добавление новой лекции не прошло. Над полем ввода названия файла отображается сообщение «Заполните это поле»</p>	Тест пройден
5	<p>Название лекции: Феномен madk1d</p> <p>Описание лекции: В рамках лекции разберем феномен успеха исполнителя madk1d</p> <p>Название файла лекции: madk1d</p> <p>Ссылка на файл лекции: «Пустое поле»</p>	<p>Добавление новой лекции не прошло. Над полем ввода ссылки на файл лекции отображается сообщение «Заполните это поле»</p>	<p>Добавление новой лекции не прошло. Над полем ввода ссылки на файл лекции отображается сообщение «Заполните это поле»</p>	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
		ните это поле»		

### 1.9.2. Возможность изменения лекции

Таблица 11 - Тест возможности изменения лекции

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
1	<p>Название лекции: Феномен madk1d</p> <p>Описание лекции: В рамках лекции разберем феномен успеха исполнителя madk1d</p> <p>Название файла лекции: madk1d</p> <p>Ссылка на файл лекции:</p> <p><a href="https://docs.google.com/document/d/19sCVEaF6-0WVC0TSLCf4WkcSuM4r8X6a/edit?usp=drive_link&amp;ouid=104502585184625130868&amp;rtpof=true&amp;sd=true">https://docs.google.com/document/d/19sCVEaF6-0WVC0TSLCf4WkcSuM4r8X6a/edit?usp=drive_link&amp;ouid=104502585184625130868&amp;rtpof=true&amp;sd=true</a></p>	Успешное изменение лекции	Успешное изменение лекции	Тест пройден
2	<p>Название лекции: «Пустое поле»</p> <p>Описание лекции: В рамках лекции разберем феномен успеха исполнителя madk1d</p> <p>Название файла лекции: madk1d</p> <p>Ссылка на файл лекции:</p> <p><a href="https://docs.google.com/document/d/19sCVEaF6-0WVC0TSLCf4WkcSuM4r8X6a/edit?usp=drive_link&amp;ouid=104502585184625130868&amp;rtpof=true&amp;sd=true">https://docs.google.com/document/d/19sCVEaF6-0WVC0TSLCf4WkcSuM4r8X6a/edit?usp=drive_link&amp;ouid=104502585184625130868&amp;rtpof=true&amp;sd=true</a></p>	Изменение лекции не прошл о. Над полем ввода назван ия лекции отобра жается сообщ ение «Запол ните это поле»	Изменение лекции не прошл о. Над полем ввода назван ия лекции отобра жается сообщ ение «Запол ните это поле»	Тест пройден
3	<p>Название лекции: Феномен madk1d</p> <p>Описание лекции: «Пустое поле»</p> <p>Название файла лекции: madk1d</p> <p>Ссылка на файл лекции:</p> <p><a href="https://docs.google.com/document/d/19sCVEaF6-0WVC0TSLCf4WkcSuM4r8X6a/edit?usp=drive_link&amp;ouid=104502585184625130868&amp;rtpof=true&amp;sd=true">https://docs.google.com/document/d/19sCVEaF6-0WVC0TSLCf4WkcSuM4r8X6a/edit?usp=drive_link&amp;ouid=104502585184625130868&amp;rtpof=true&amp;sd=true</a></p>	Изменение лекции не прошл	Изменение лекции не прошл	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	<a href="https://docs.google.com/document/d/19sCVEaF6-0WVC0TSLCf4WkcSuM4r8X6a/edit?usp=drive_link&amp;ouid=104502585184625130868&amp;rtpof=true&amp;sd=true">0WVC0TSLCf4WkcSuM4r8X6a/edit?usp=drive_link&amp;ouid=104502585184625130868&amp;rtpof=true&amp;sd=true</a>	о. Над полем ввода описан ия лекции отобра жается сообщение «Заполните это поле»	о. Над полем ввода описан ия лекции отобра жается сообщение «Заполните это поле»	
4	<p>Название лекции: Феномен madk1d</p> <p>Описание лекции: В рамках лекции разберем феномен успеха исполнителя madk1d</p> <p>Название файла лекции: «Пустое поле»</p> <p>Ссылка на файл лекции:</p> <p><a href="https://docs.google.com/document/d/19sCVEaF6-0WVC0TSLCf4WkcSuM4r8X6a/edit?usp=drive_link&amp;ouid=104502585184625130868&amp;rtpof=true&amp;sd=true">https://docs.google.com/document/d/19sCVEaF6-0WVC0TSLCf4WkcSuM4r8X6a/edit?usp=drive_link&amp;ouid=104502585184625130868&amp;rtpof=true&amp;sd=true</a></p>	Изменение лекции не прошло.	Изменение лекции не прошло.	Тест пройден
5	<p>Название лекции: Феномен madk1d</p> <p>Описание лекции: В рамках лекции разберем феномен успеха исполнителя madk1d</p> <p>Название файла лекции: madk1d</p> <p>Ссылка на файл лекции: «Пустое поле»</p>	Изменение лекции не прошло.	Изменение лекции не прошло.	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
		ожается сообщение «Заполните это поле»	ожается сообщение «Заполните это поле»	

### 1.9.3. Возможность добавления новой практической работы

Таблица 12 - Тест возможности добавления новой практической работы

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
1	<p>Название практической работы: Практическая работа №1</p> <p>Описание практической работы: В рамках практической работы необходимо записаться под fortuna812 type бит</p> <p>Название файла: Практическая работа №1</p> <p>Ссылка на файл:</p> <p><a href="https://docs.google.com/document/d/19sCVEaF6-0WVC0TSLCf4WkcSuM4r8X6a/edit?usp=drive_link&amp;ouid=104502585184625130868&amp;rtpof=true&amp;sd=true">https://docs.google.com/document/d/19sCVEaF6-0WVC0TSLCf4WkcSuM4r8X6a/edit?usp=drive_link&amp;ouid=104502585184625130868&amp;rtpof=true&amp;sd=true</a></p>	Успешное добавление новой практической работы	Успешное добавление новой практической работы	Тест пройден
2	<p>Название практической работы: «Пустое поле»</p> <p>Описание практической работы: В рамках практической работы необходимо записаться под fortuna812 type бит</p> <p>Название файла: Практическая работа №1</p> <p>Ссылка на файл:</p> <p><a href="https://docs.google.com/document/d/19sCVEaF6-0WVC0TSLCf4WkcSuM4r8X6a/edit?usp=drive_link&amp;ouid=104502585184625130868&amp;rtpof=true&amp;sd=true">https://docs.google.com/document/d/19sCVEaF6-0WVC0TSLCf4WkcSuM4r8X6a/edit?usp=drive_link&amp;ouid=104502585184625130868&amp;rtpof=true&amp;sd=true</a></p>	Добавление новой практической работы не прошл о. Над полем ввода названи я практической работы отобра жается	Добавление новой практической работы не прошл о. Над полем ввода названи я практической работы отобра жается	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
		сообщение «Заполните это поле»	сообщение «Заполните это поле»	
3	<p>Название практической работы: Практическая работа №1</p> <p>Описание практической работы: «Пустое поле»</p> <p>Название файла: Практическая работа №1</p> <p>Ссылка на файл:</p> <p><a href="https://docs.google.com/document/d/19sCVEaF6-0WVC0TSLCf4WkcSuM4r8X6a/edit?usp=drive_link&amp;ouid=104502585184625130868&amp;rtpof=true&amp;sd=true">https://docs.google.com/document/d/19sCVEaF6-0WVC0TSLCf4WkcSuM4r8X6a/edit?usp=drive_link&amp;ouid=104502585184625130868&amp;rtpof=true&amp;sd=true</a></p>	<p>Добавление новой практической работы не прошл о. Над полем ввода описан ия практической работы отобра жается сообщение «Заполните это поле»</p>	<p>Добавление новой практической работы не прошл о. Над полем ввода описан ия практической работы отобра жается сообщение «Заполните это поле»</p>	Тест пройден
4	<p>Название практической работы: Практическая работа №1</p> <p>Описание практической работы: В рамках практической работы необходимо записаться под fortuna812 type бит</p> <p>Название файла: «Пустое поле»</p> <p>Ссылка на файл:</p> <p><a href="https://docs.google.com/document/d/19sCVEaF6-0WVC0TSLCf4WkcSuM4r8X6a/edit?usp=drive_link&amp;ouid=104502585184625130868&amp;rtpof=true&amp;sd=true">https://docs.google.com/document/d/19sCVEaF6-0WVC0TSLCf4WkcSuM4r8X6a/edit?usp=drive_link&amp;ouid=104502585184625130868&amp;rtpof=true&amp;sd=true</a></p>	<p>Добавление новой практической работы не прошл о. Над полем ввода назван ия файла отобра жается</p>	<p>Добавление новой практической работы не прошл о. Над полем ввода назван ия файла отобра жается</p>	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
		сообщение «Заполните это поле»	сообщение «Заполните это поле»	
5	Название практической работы: Практическая работа №1  Описание практической работы: В рамках практической работы необходимо записаться под fortuna812 type бит  Название файла: Практическая работа №1 Ссылка на файл: «Пустое поле»	Добавление новой практической работы не прошл о. Над полем ввода ссылки на файл отобра жается сообщение «Заполните это поле»	Добавление новой практической работы не прошл о. Над полем ввода ссылки на файл отобра жается сообщение «Заполните это поле»	Тест пройден

#### 1.9.4. Возможность добавления оценки к практической работе

Таблица 13 - Тест возможности добавления оценки к практической работе

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	Оценка:5 (Отлично)	Успешное добавление оценки к практической работе	Успешное добавление оценки к практической работе	Тест пройден
	Оценка: «Не выбрана»	Добавление оценки к практической работе не	Добавление оценки к практической работе не	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
		прошло. Над полем выбора оценки отображается сообщение «Выберите один из пунктов списка.»	прошло. Над полем выбора оценки отображается сообщение «Выберите один из пунктов списка.»	

#### 1.9.5. Возможность создания теста

Таблица 14 - Тест возможности создания теста

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
1	Название теста: Вводный тест Описание теста: Вводный тест для проверки знаний Ограничение по времени: 10 Оценка 5 от: 9 Оценка 5 до: 10 Оценка 4 от: 7 Оценка 4 до: 8 Оценка 3 от: 5 Оценка 3 до: 6 Вопросы: Созданы	Тест успешно создан	Тест успешно создан	Тест пройден
2	Название теста: «Пустое поле» Описание теста: Вводный тест для проверки знаний Ограничение по времени: 10 Оценка 5 от: 9 Оценка 5 до: 10 Оценка 4 от: 7 Оценка 4 до: 8 Оценка 3 от: 5	Создание теста не прошло. Над полем ввода названия теста отображается сообщение «Заполните это поле.»	Создание теста не прошло. Над полем ввода названия теста отображается сообщение «Заполните это поле.»	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	Оценка 3 до:6 Вопросы: Созданы			
3	Название теста: Вводный тест Описание теста: «Пустое поле» Ограничение по времени: 10 Оценка 5 от: 9 Оценка 5 до: 10 Оценка 4 от:7 Оценка 4 до:8 Оценка 3 от:5 Оценка 3 до:6 Вопросы: Созданы	Создание теста не прошло. Над полем ввода описания теста отображается сообщение «Заполните это поле.»	Создание теста не прошло. Над полем ввода описания теста отображается сообщение «Заполните это поле.»	Тест пройден
4	Название теста: Вводный тест Описание теста: Вводный тест для проверки знаний Ограничение по времени: «Пустое поле» Оценка 5 от: 9 Оценка 5 до: 10 Оценка 4 от:7 Оценка 4 до:8 Оценка 3 от:5 Оценка 3 до:6 Вопросы: Созданы	Создание теста не прошло. Над полем ввода ограничения по времени отображается сообщение «Заполните это поле.»	Создание теста не прошло. Над полем ввода ограничения по времени отображается сообщение «Заполните это поле.»	Тест пройден
5	Название теста: Вводный тест Описание теста: Вводный тест для проверки знаний Ограничение по времени: 10 Оценка 5 от: «Пустое поле» Оценка 5 до: 10	Создание теста не прошло. Над полем ввода начала диапазона на оценку 5 отображается сообщение «Заполните это поле.»	Создание теста не прошло. Над полем ввода начала диапазона на оценку 5 отображается сообщение «Заполните это поле.»	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	Оценка 4 от:7 Оценка 4 до:8 Оценка 3 от:5 Оценка 3 до:6 Вопросы: Созданы			
6	Название теста: Вводный тест Описание теста: Вводный тест для проверки знаний Ограничение по времени: 10 Оценка 5 от: 9 Оценка 5 до: «Пустое поле» Оценка 4 от:7 Оценка 4 до:8 Оценка 3 от:5 Оценка 3 до:6 Вопросы: Созданы	Создание теста не прошло. Над полем ввода конца диапазона на оценку 5 отображается сообщение «Заполните это поле.»	Создание теста не прошло. Над полем ввода конца диапазона на оценку 5 отображается сообщение «Заполните это поле.»	Тест пройден
7	Название теста: Вводный тест Описание теста: Вводный тест для проверки знаний Ограничение по времени: 10 Оценка 5 от: 9 Оценка 5 до: 10 Оценка 4 от: «Пустое поле» Оценка 4 до:8 Оценка 3 от:5 Оценка 3 до:6 Вопросы: Созданы	Создание теста не прошло. Над полем ввода начала диапазона на оценку 4 отображается сообщение «Заполните это поле.»	Создание теста не прошло. Над полем ввода начала диапазона на оценку 4 отображается сообщение «Заполните это поле.»	Тест пройден
8	Название теста: Вводный тест Описание теста: Вводный тест для проверки	Создание теста не прошло. Над полем ввода конца диапазона на оценку 4	Создание теста не прошло. Над полем ввода конца диапазона на оценку 4	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	знаний Ограничение по времени: 10 Оценка 5 от: 9 Оценка 5 до: 10 Оценка 4 от:7 Оценка 4 до: «Пустое поле» Оценка 3 от:5 Оценка 3 до:6 Вопросы: Созданы	отображается сообщение «Заполните это поле.»	отображается сообщение «Заполните это поле.»	
9	Название теста: Вводный тест Описание теста: Вводный тест для проверки знаний Ограничение по времени: 10 Оценка 5 от: 9 Оценка 5 до: 10 Оценка 4 от:7 Оценка 4 до:8 Оценка 3 от: «Пустое поле» Оценка 3 до:6 Вопросы: Созданы	Создание теста не прошло. Над полем ввода начала диапазона на оценку 3 отображается сообщение «Заполните это поле.»	Создание теста не прошло. Над полем ввода начала диапазона на оценку 3 отображается сообщение «Заполните это поле.»	Тест пройден
10	Название теста: Вводный тест Описание теста: Вводный тест для проверки знаний Ограничение по времени: 10 Оценка 5 от: 9 Оценка 5 до: 10 Оценка 4 от:7 Оценка 4 до:8 Оценка 3 от:5 Оценка 3 до: «Пустое поле» Вопросы: Созданы	Создание теста не прошло. Над полем ввода конца диапазона на оценку 3 отображается сообщение «Заполните это поле.»	Создание теста не прошло. Над полем ввода конца диапазона на оценку 3 отображается сообщение «Заполните это поле.»	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
11	<p>Название теста: Вводный тест Описание теста: Вводный тест для проверки знаний Ограничение по времени: 10 Оценка 5 от: 9 Оценка 5 до: 10 Оценка 4 от: 7 Оценка 4 до: 8 Оценка 3 от: 5 Оценка 3 до: 6 Вопросы: Не созданы</p>	<p>Создание теста не прошло. Над полями ввода отображается сообщение «Чтобы создать тест нужен хотя бы 1 вопрос»</p>	<p>Создание теста не прошло. Над полями ввода отображается сообщение «Чтобы создать тест нужен хотя бы 1 вопрос»</p>	<p>Тест пройден</p>

#### 1.9.6. Возможность добавления вопросов в тест

Таблица 15 - Тест возможности добавления вопросов в тест

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
1	<p>Текст вопроса: Выберите правильный перевод слова what Тип вопроса: Один правильный ответ Баллы за вопрос: 1 Вариант 1: что Вариант 2: кто Вариант 3: когда Вариант 4: почему</p>	<p>Вопрос успешно добавлен</p>	<p>Вопрос успешно добавлен</p>	<p>Тест пройден</p>
2	<p>Текст вопроса: «Пустое поле» Тип вопроса: Один правильный</p>	<p>Добавление вопроса не прошло. Над полем ввода текста вопроса</p>	<p>Добавление вопроса не прошло. Над полем ввода текста вопроса</p>	<p>Тест пройден</p>

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	ответ Баллы за вопрос: 1 Вариант 1: что Вариант 2: кто Вариант 3: когда Вариант 4: почему	отображается сообщение «Заполните это поле.»	отображается сообщение «Заполните это поле.»	
3	Текст вопроса: Выберите правильный перевод слова what Тип вопроса: Один правильный ответ Баллы за вопрос: «Пустое поле» Вариант 1: что Вариант 2: кто Вариант 3: когда Вариант 4: почему	Добавление вопроса не прошло. Над полем ввода балла за вопрос отображается сообщение «Заполните это поле.»	Добавление вопроса не прошло. Над полем ввода балла за вопрос отображается сообщение «Заполните это поле.»	Тест пройден
4	Текст вопроса: Выберите правильный перевод слова what Тип вопроса: Один правильный ответ Баллы за вопрос: 1 Вариант 1: «Пустое поле» Вариант 2: кто Вариант 3: когда Вариант 4: почему	Добавление вопроса не прошло. Над полем ввода вариант 1 отображается сообщение «Заполните это поле.»	Добавление вопроса не прошло. Над полем ввода вариант 1 отображается сообщение «Заполните это поле.»	Тест пройден
5	Текст вопроса: Выберите правильный	Добавление вопроса не прошло. Над	Добавление вопроса не прошло. Над	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	<p>перевод слова what</p> <p>Тип вопроса: Один правильный ответ</p> <p>Баллы за вопрос: 1</p> <p>Вариант 1: что</p> <p>Вариант 2: «Пустое поле»</p> <p>Вариант 3: когда</p> <p>Вариант 4: почему</p>	<p>полем ввода вариант 2 отображается сообщение «Заполните это поле.»</p>	<p>полем ввода вариант 2 отображается сообщение «Заполните это поле.»</p>	
6	<p>Текст вопроса: Выберите правильный перевод слова what</p> <p>Тип вопроса: Один правильный ответ</p> <p>Баллы за вопрос: 1</p> <p>Вариант 1: что</p> <p>Вариант 2: кто</p> <p>Вариант 3: «Пустое поле»</p> <p>Вариант 4: почему</p>	<p>Добавление вопроса не прошло. Над полем ввода вариант 3 отображается сообщение «Заполните это поле.»</p>	<p>Добавление вопроса не прошло. Над полем ввода вариант 3 отображается сообщение «Заполните это поле.»</p>	Тест пройден
7	<p>Текст вопроса: Выберите правильный перевод слова what</p> <p>Тип вопроса: Один правильный ответ</p> <p>Баллы за вопрос: 1</p> <p>Вариант 1: что</p> <p>Вариант 2: кто</p> <p>Вариант 3: когда</p> <p>Вариант 4:</p>	<p>Добавление вопроса не прошло. Над полем ввода вариант 4 отображается сообщение «Заполните это поле.»</p>	<p>Добавление вопроса не прошло. Над полем ввода вариант 4 отображается сообщение «Заполните это поле.»</p>	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	«Пустое поле»			
8	<p>Текст вопроса: Как переводится what и where</p> <p>Тип вопроса: Несколько правильных ответов</p> <p>Баллы за вопрос: 1</p> <p>Вариант 1: что</p> <p>Вариант 2: где</p> <p>Вариант 3: когда</p> <p>Вариант 4: почему</p>	Вопрос успешно добавлен	Вопрос успешно добавлен	Тест пройден
9	<p>Текст вопроса: «Пустое поле»</p> <p>Тип вопроса: Несколько правильных ответов</p> <p>Баллы за вопрос: 1</p> <p>Вариант 1: что</p> <p>Вариант 2: где</p> <p>Вариант 3: когда</p> <p>Вариант 4: почему</p>	Добавление вопроса не прошло. Над полем ввода текста вопроса отображается сообщение «Заполните это поле.»	Добавление вопроса не прошло. Над полем ввода текста вопроса отображается сообщение «Заполните это поле.»	Тест пройден
10	<p>Текст вопроса: Как переводится what и where</p> <p>Тип вопроса: Несколько правильных ответов</p> <p>Баллы за вопрос:</p> <p>«Пустое поле»</p> <p>Вариант 1: что</p> <p>Вариант 2: где</p> <p>Вариант 3: когда</p> <p>Вариант 4: почему</p>	Добавление вопроса не прошло. Над полем ввода балла за вопрос отображается сообщение «Заполните это поле.»	Добавление вопроса не прошло. Над полем ввода балла за вопрос отображается сообщение «Заполните это поле.»	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
11	<p>Текст вопроса: Как переводится what и where Тип вопроса: Несколько правильных ответов Баллы за вопрос: 1 Вариант 1: «Пустое поле» Вариант 2: где Вариант 3: когда Вариант 4: почему</p>	<p>Добавление вопроса не прошло. Над полем ввода вариант 1 отображается сообщение «Заполните это поле.»</p>	<p>Добавление вопроса не прошло. Над полем ввода вариант 1 отображается сообщение «Заполните это поле.»</p>	Тест пройден
12	<p>Текст вопроса: Как переводится what и where Тип вопроса: Несколько правильных ответов Баллы за вопрос: 1 Вариант 1: что Вариант 2: «Пустое поле» Вариант 3: когда Вариант 4: почему</p>	<p>Добавление вопроса не прошло. Над полем ввода вариант 2 отображается сообщение «Заполните это поле.»</p>	<p>Добавление вопроса не прошло. Над полем ввода вариант 2 отображается сообщение «Заполните это поле.»</p>	Тест пройден
13	<p>Текст вопроса: Как переводится what и where Тип вопроса: Несколько правильных ответов Баллы за вопрос: 1 Вариант 1: что Вариант 2: где Вариант 3: «Пустое поле»</p>	<p>Добавление вопроса не прошло. Над полем ввода вариант 3 отображается сообщение «Заполните это поле.»</p>	<p>Добавление вопроса не прошло. Над полем ввода вариант 3 отображается сообщение «Заполните это поле.»</p>	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	Вариант 4: почему			
14	Текст вопроса: Как переводится what и where Тип вопроса: Несколько правильных ответов Баллы за вопрос: 1 Вариант 1: что Вариант 2: где Вариант 3: когда Вариант 4: «Пустое поле»	Добавление вопроса не прошло. Над полем ввода вариант 4 отображается сообщение «Заполните это поле.»	Добавление вопроса не прошло. Над полем ввода вариант 4 отображается сообщение «Заполните это поле.»	Тест пройден
15	Текст вопроса: Введите в алфавитном порядке b c a d Тип вопроса: Установление порядка Баллы за вопрос: 1 Элемент 1: b Элемент 2: c Элемент 3: a Элемент 4: d	Вопрос успешно добавлен	Вопрос успешно добавлен	Тест пройден
16	Текст вопроса: «Пустое поле» Тип вопроса: Установление порядка Баллы за вопрос: 1 Элемент 1: b Элемент 2: c Элемент 3: a Элемент 4: d	Добавление вопроса не прошло. Над полем ввода текста вопроса отображается сообщение «Заполните это поле.»	Добавление вопроса не прошло. Над полем ввода текста вопроса отображается сообщение «Заполните это поле.»	Тест пройден
17	Текст вопроса: Введите в алфавитном порядке b c a d Тип вопроса: Установление	Добавление вопроса не прошло. Над полем ввода балла за вопрос отображается	Добавление вопроса не прошло. Над полем ввода балла за вопрос отображается	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	порядка Баллы за вопрос: «Пустое поле» Элемент 1: b Элемент 2: c Элемент 3: a Элемент 4: d	сообщение «Заполните это поле.»	сообщение «Заполните это поле.»	
18	Текст вопроса: Ведите в алфавитном порядке b c a d Тип вопроса: Установление порядка Баллы за вопрос: 1 Элемент 1: «Пустое поле» Элемент 2: c Элемент 3: a Элемент 4: d	Добавление вопроса не прошло. Над полем ввода элемент 1 отображается сообщение «Заполните это поле.»	Добавление вопроса не прошло. Над полем ввода элемент 1 отображается сообщение «Заполните это поле.»	Тест пройден
19	Текст вопроса: Ведите в алфавитном порядке b c a d Тип вопроса: Установление порядка Баллы за вопрос: 1 Элемент 1: b Элемент 2: «Пустое поле» Элемент 3: a Элемент 4: d	Добавление вопроса не прошло. Над полем ввода элемент 2 отображается сообщение «Заполните это поле.»	Добавление вопроса не прошло. Над полем ввода элемент 2 отображается сообщение «Заполните это поле.»	Тест пройден
20	Текст вопроса: Ведите в алфавитном порядке b c a d Тип вопроса: Установление порядка Баллы за вопрос: 1 Элемент 1: b Элемент 2: c Элемент 3:	Добавление вопроса не прошло. Над полем ввода элемент 3 отображается сообщение «Заполните это поле.»	Добавление вопроса не прошло. Над полем ввода элемент 3 отображается сообщение «Заполните это поле.»	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	«Пустое поле» Элемент 4: d			
21	Текст вопроса: Ведите в алфавитном порядке b с a d Тип вопроса: Установление порядка Баллы за вопрос: 1 Элемент 1: b Элемент 2: c Элемент 3: a Элемент 4: «Пустое поле»	Добавление вопроса не прошло. Над полем ввода элемент 4 отображается сообщение «Заполните это поле.»	Добавление вопроса не прошло. Над полем ввода элемент 4 отображается сообщение «Заполните это поле.»	Тест пройден
22	Текст вопроса: Напишите как переводится слово which Тип вопроса: Ввод ответа Баллы за вопрос: 1 Правильный ответ: который	Вопрос успешно добавлен	Вопрос успешно добавлен	Тест пройден
23	Текст вопроса: «Пустое поле» Тип вопроса: Ввод ответа Баллы за вопрос: 1 Правильный ответ: который	Добавление вопроса не прошло. Над полем ввода текста вопроса отображается сообщение «Заполните это поле.»	Добавление вопроса не прошло. Над полем ввода текста вопроса отображается сообщение «Заполните это поле.»	Тест пройден
24	Текст вопроса: Напишите как переводится слово which Тип вопроса: Ввод ответа Баллы за вопрос: «Пустое поле» Правильный ответ: который	Добавление вопроса не прошло. Над полем ввода балла за вопрос отображается сообщение «Заполните это поле.»	Добавление вопроса не прошло. Над полем ввода балла за вопрос отображается сообщение «Заполните это поле.»	Тест пройден
25	Текст вопроса: Напишите как	Добавление вопроса не	Добавление вопроса не	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	переводится слово which Тип вопроса: Ввод ответа Баллы за вопрос: 1 Правильный ответ: «Пустое поле»	прошло. Над полем ввода правильный ответ отображается сообщение «Заполните это поле.»	прошло. Над полем ввода правильный ответ отображается сообщение «Заполните это поле.»	
26	Текст вопроса: Сопоставьте правильный перевод слов Тип вопроса: Сопоставление Баллы за вопрос: 1 Пара 1 (Левая часть): what Пара 1 (Правая часть): что Пара 2 (Левая часть): where Пара 2 (Правая часть): где Пара 3 (Левая часть): which Пара 3 (Правая часть): который Пара 4 (Левая часть): who Пара 4 (Правая часть): кто	Вопрос успешно добавлен	Вопрос успешно добавлен	Тест пройден
27	Текст вопроса: «Пустое поле» Тип вопроса: Сопоставление Баллы за вопрос: 1 Пара 1 (Левая часть): what Пара 1 (Правая часть): что Пара 2 (Левая часть): where Пара 2 (Правая часть): где	Добавление вопроса не прошло. Над полем ввода текст вопроса отображается сообщение «Заполните это поле.»	Добавление вопроса не прошло. Над полем ввода текст вопроса отображается сообщение «Заполните это поле.»	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	Пара 3 (Левая часть): which Пара 3 (Правая часть): который Пара 4 (Левая часть): who Пара 4 (Правая часть): кто			
28	Текст вопроса: «Пустое поле» Тип вопроса: Сопоставление Баллы за вопрос: «Пустое поле» Пара 1 (Левая часть): what Пара 1 (Правая часть): что Пара 2 (Левая часть): where Пара 2 (Правая часть): где Пара 3 (Левая часть): which Пара 3 (Правая часть): который Пара 4 (Левая часть): who Пара 4 (Правая часть): кто	Добавление вопроса не прошло. Над полем ввода баллы за вопрос отображается сообщение «Заполните это поле.»	Добавление вопроса не прошло. Над полем ввода баллы за вопрос отображается сообщение «Заполните это поле.»	Тест пройден
29	Текст вопроса: «Пустое поле» Тип вопроса: Сопоставление Баллы за вопрос: 1 Пара 1 (Левая часть): «Пустое поле» Пара 1 (Правая часть): что Пара 2 (Левая часть): where Пара 2 (Правая часть): где	Добавление вопроса не прошло. Над полем ввода левой части пары 1 отображается сообщение «Заполните это поле.»	Добавление вопроса не прошло. Над полем ввода левой части пары 1 отображается сообщение «Заполните это поле.»	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	Пара 3 (Левая часть): which Пара 3 (Правая часть): который Пара 4 (Левая часть): who Пара 4 (Правая часть): кто			
30	Текст вопроса: «Пустое поле» Тип вопроса: Сопоставление Баллы за вопрос: 1 Пара 1 (Левая часть): what Пара 1 (Правая часть): «Пустое поле» Пара 2 (Левая часть): where Пара 2 (Правая часть): где Пара 3 (Левая часть): which Пара 3 (Правая часть): который Пара 4 (Левая часть): who Пара 4 (Правая часть): кто	Добавление вопроса не прошло. Над полем ввода правой части пары 1 отображается сообщение «Заполните это поле.»	Добавление вопроса не прошло. Над полем ввода правой части пары 1 отображается сообщение «Заполните это поле.»	Тест пройден
31	Текст вопроса: «Пустое поле» Тип вопроса: Сопоставление Баллы за вопрос: 1 Пара 1 (Левая часть): what Пара 1 (Правая часть): что Пара 2 (Левая часть): «Пустое поле» Пара 2 (Правая часть): где	Добавление вопроса не прошло. Над полем ввода левой части пары 2 отображается сообщение «Заполните это поле.»	Добавление вопроса не прошло. Над полем ввода левой части пары 2 отображается сообщение «Заполните это поле.»	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	Пара 3 (Левая часть): which Пара 3 (Правая часть): который Пара 4 (Левая часть): who Пара 4 (Правая часть): кто			
32	Текст вопроса: «Пустое поле» Тип вопроса: Сопоставление Баллы за вопрос: 1 Пара 1 (Левая часть): what Пара 1 (Правая часть): что Пара 2 (Левая часть): where Пара 2 (Правая часть): «Пустое поле» Пара 3 (Левая часть): which Пара 3 (Правая часть): который Пара 4 (Левая часть): who Пара 4 (Правая часть): кто	Добавление вопроса не прошло. Над полем ввода правой части пары 2 отображается сообщение «Заполните это поле.»	Добавление вопроса не прошло. Над полем ввода правой части пары 2 отображается сообщение «Заполните это поле.»	Тест пройден
33	Текст вопроса: «Пустое поле» Тип вопроса: Сопоставление Баллы за вопрос: 1 Пара 1 (Левая часть): what Пара 1 (Правая часть): что Пара 2 (Левая часть): where Пара 2 (Правая часть): где Пара 3 (Левая	Добавление вопроса не прошло. Над полем ввода левой части пары 3 отображается сообщение «Заполните это поле.»	Добавление вопроса не прошло. Над полем ввода левой части пары 3 отображается сообщение «Заполните это поле.»	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	часть): «Пустое поле» Пара 3 (Правая часть): который Пара 4 (Левая часть): who Пара 4 (Правая часть): кто			
34	Текст вопроса: «Пустое поле» Тип вопроса: Сопоставление Баллы за вопрос: 1 Пара 1 (Левая часть): what Пара 1 (Правая часть): что Пара 2 (Левая часть): where Пара 2 (Правая часть): где Пара 3 (Левая часть): which Пара 3 (Правая часть): «Пустое поле» Пара 4 (Левая часть): who Пара 4 (Правая часть): кто	Добавление вопроса не прошло. Над полем ввода правой части пары 3 отображается сообщение «Заполните это поле.»	Добавление вопроса не прошло. Над полем ввода правой части пары 3 отображается сообщение «Заполните это поле.»	Тест пройден
35	Текст вопроса: «Пустое поле» Тип вопроса: Сопоставление Баллы за вопрос: 1 Пара 1 (Левая часть): what Пара 1 (Правая часть): что Пара 2 (Левая часть): where Пара 2 (Правая часть): где Пара 3 (Левая часть): which	Добавление вопроса не прошло. Над полем ввода левой части пары 4 отображается сообщение «Заполните это поле.»	Добавление вопроса не прошло. Над полем ввода левой части пары 4 отображается сообщение «Заполните это поле.»	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	Пара 3 (Правая часть): который Пара 4 (Левая часть): «Пустое поле» Пара 4 (Правая часть): кто			
36	Текст вопроса: «Пустое поле» Тип вопроса: Сопоставление Баллы за вопрос: 1 Пара 1 (Левая часть): what Пара 1 (Правая часть): что Пара 2 (Левая часть): where Пара 2 (Правая часть): где Пара 3 (Левая часть): which Пара 3 (Правая часть): который Пара 4 (Левая часть): who Пара 4 (Правая часть): «Пустое поле»	Добавление вопроса не прошло. Над полем ввода правой части пары 4 отображается сообщение «Заполните это поле.»	Добавление вопроса не прошло. Над полем ввода правой части пары 4 отображается сообщение «Заполните это поле.»	Тест пройден

#### 1.9.7. Возможность изменения теста

Таблица 16 – Тест возможности изменения теста

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
1	Название теста: Вводный тест Описание теста: Вводный тест для проверки знаний Ограничение	Тест успешно изменен	Тест успешно изменен	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	по времени: 10 Оценка 5 от: 9 Оценка 5 до: 10 Оценка 4 от:7 Оценка 4 до:8 Оценка 3 от:5 Оценка 3 до:6 Вопросы: Созданы			
2	Название теста: «Пустое поле» Описание теста: Вводный тест для проверки знаний Ограничение по времени: 10 Оценка 5 от: 9 Оценка 5 до: 10 Оценка 4 от:7 Оценка 4 до:8 Оценка 3 от:5 Оценка 3 до:6 Вопросы: Созданы	Изменение теста не прошло. Над полем ввода названия теста отображается сообщение «Заполните это поле.»	Изменение теста не прошло. Над полем ввода названия теста отображается сообщение «Заполните это поле.»	Тест пройден
3	Название теста: Вводный тест Описание теста: «Пустое поле» Ограничение по времени: 10 Оценка 5 от: 9 Оценка 5 до: 10 Оценка 4 от:7 Оценка 4 до:8 Оценка 3 от:5 Оценка 3 до:6 Вопросы: Созданы	Изменение теста не прошло. Над полем ввода описания теста отображается сообщение «Заполните это поле.»	Изменение теста не прошло. Над полем ввода описания теста отображается сообщение «Заполните это поле.»	Тест пройден
4	Название теста: Вводный тест Описание теста: Вводный тест для проверки знаний	Изменение теста не прошло. Над полем ввода ограничения по времени отображается	Изменение теста не прошло. Над полем ввода ограничения по времени отображается	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	<p>Ограничение по времени: «Пустое поле»</p> <p>Оценка 5 от: 9</p> <p>Оценка 5 до: 10</p> <p>Оценка 4 от:7</p> <p>Оценка 4 до:8</p> <p>Оценка 3 от:5</p> <p>Оценка 3 до:6</p> <p>Вопросы: Созданы</p>	сообщение «Заполните это поле.»	сообщение «Заполните это поле.»	
5	<p>Название теста: Вводный тест</p> <p>Описание теста: Вводный тест для проверки знаний</p> <p>Ограничение по времени: 10</p> <p>Оценка 5 от: «Пустое поле»</p> <p>Оценка 5 до: 10</p> <p>Оценка 4 от:7</p> <p>Оценка 4 до:8</p> <p>Оценка 3 от:5</p> <p>Оценка 3 до:6</p> <p>Вопросы: Созданы</p>	Изменение теста не прошло. Над полем ввода начала диапазона на оценку 5 отображается сообщение «Заполните это поле.»	Изменение теста не прошло. Над полем ввода начала диапазона на оценку 5 отображается сообщение «Заполните это поле.»	Тест пройден
6	<p>Название теста: Вводный тест</p> <p>Описание теста: Вводный тест для проверки знаний</p> <p>Ограничение по времени: 10</p> <p>Оценка 5 от: 9</p> <p>Оценка 5 до: «Пустое поле»</p> <p>Оценка 4 от:7</p> <p>Оценка 4 до:8</p> <p>Оценка 3 от:5</p> <p>Оценка 3 до:6</p> <p>Вопросы: Созданы</p>	Изменение теста не прошло. Над полем ввода конца диапазона на оценку 5 отображается сообщение «Заполните это поле.»	Изменение теста не прошло. Над полем ввода конца диапазона на оценку 5 отображается сообщение «Заполните это поле.»	Тест пройден
7	Название теста:	Изменение	Изменение	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	<p>Вводный тест</p> <p>Описание теста: Вводный тест для проверки знаний</p> <p>Ограничение по времени: 10</p> <p>Оценка 5 от: 9</p> <p>Оценка 5 до: 10</p> <p>Оценка 4 от: «Пустое поле»</p> <p>Оценка 4 до:8</p> <p>Оценка 3 от:5</p> <p>Оценка 3 до:6</p> <p>Вопросы: Созданы</p>	<p>теста не прошло. Над полем ввода начала диапазона на оценку 4 отображается сообщение «Заполните это поле.»</p>	<p>теста не прошло. Над полем ввода начала диапазона на оценку 4 отображается сообщение «Заполните это поле.»</p>	
8	<p>Название теста: Вводный тест</p> <p>Описание теста: Вводный тест для проверки знаний</p> <p>Ограничение по времени: 10</p> <p>Оценка 5 от: 9</p> <p>Оценка 5 до: 10</p> <p>Оценка 4 от:7</p> <p>Оценка 4 до: «Пустое поле»</p> <p>Оценка 3 от:5</p> <p>Оценка 3 до:6</p> <p>Вопросы: Созданы</p>	<p>Изменение теста не прошло. Над полем ввода конца диапазона на оценку 4 отображается сообщение «Заполните это поле.»</p>	<p>Изменение теста не прошло. Над полем ввода конца диапазона на оценку 4 отображается сообщение «Заполните это поле.»</p>	Тест пройден
9	<p>Название теста: Вводный тест</p> <p>Описание теста: Вводный тест для проверки знаний</p> <p>Ограничение по времени: 10</p> <p>Оценка 5 от: 9</p> <p>Оценка 5 до: 10</p> <p>Оценка 4 от:7</p> <p>Оценка 4 до:8</p>	<p>Изменение теста не прошло. Над полем ввода начала диапазона на оценку 3 отображается сообщение «Заполните это поле.»</p>	<p>Изменение теста не прошло. Над полем ввода начала диапазона на оценку 3 отображается сообщение «Заполните это поле.»</p>	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	Оценка 3 от: «Пустое поле» Оценка 3 до:6 Вопросы: Созданы			
10	Название теста: Вводный тест Описание теста: Вводный тест для проверки знаний Ограничение по времени: 10 Оценка 5 от: 9 Оценка 5 до: 10 Оценка 4 от:7 Оценка 4 до:8 Оценка 3 от:5 Оценка 3 до: «Пустое поле» Вопросы: Созданы	Изменение теста не прошло. Над полем ввода конца диапазона на оценку 3 отображается сообщение «Заполните это поле.»	Изменение теста не прошло. Над полем ввода конца диапазона на оценку 3 отображается сообщение «Заполните это поле.»	Тест пройден
11	Название теста: Вводный тест Описание теста: Вводный тест для проверки знаний Ограничение по времени: 10 Оценка 5 от: 9 Оценка 5 до: 10 Оценка 4 от:7 Оценка 4 до:8 Оценка 3 от:5 Оценка 3 до:6 Вопросы: Не созданы	Изменение теста не прошло. Над полями ввода отображается сообщение «Чтобы создать тест нужен хотя бы 1 вопрос»	Изменение теста не прошло. Над полями ввода отображается сообщение «Чтобы создать тест нужен хотя бы 1 вопрос»	Тест пройден

#### 1.9.8. Возможность изменения вопросов в тесте

Таблица 17 - Тест возможности изменения вопросов в тесте

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
---	---------------------------------	------------------------	--------------------------	--------------------------------------

1	2	3	4	5
1	<p>Текст вопроса: Выберите правильный перевод слова what Тип вопроса: Один правильный ответ Баллы за вопрос: 1 Вариант 1: что Вариант 2: кто Вариант 3: когда Вариант 4: почему</p>	Вопрос успешно изменен	Вопрос успешно изменен	Тест пройден
2	<p>Текст вопроса: «Пустое поле» Тип вопроса: Один правильный ответ Баллы за вопрос: 1 Вариант 1: что Вариант 2: кто Вариант 3: когда Вариант 4: почему</p>	Изменение вопроса не прошло. Над полем ввода текста вопроса отображается сообщение «Заполните это поле.»	Изменение вопроса не прошло. Над полем ввода текста вопроса отображается сообщение «Заполните это поле.»	Тест пройден
3	<p>Текст вопроса: Выберите правильный перевод слова what Тип вопроса: Один правильный ответ Баллы за вопрос: «Пустое поле» Вариант 1: что Вариант 2: кто Вариант 3: когда Вариант 4: почему</p>	Изменение вопроса не прошло. Над полем ввода балла за вопрос отображается сообщение «Заполните это поле.»	Изменение вопроса не прошло. Над полем ввода балла за вопрос отображается сообщение «Заполните это поле.»	Тест пройден
4	Текст вопроса: Выберите	Изменение вопроса не	Изменение вопроса не	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	<p>правильный перевод слова what</p> <p>Тип вопроса: Один правильный ответ</p> <p>Баллы за вопрос: 1</p> <p>Вариант 1: «Пустое поле»</p> <p>Вариант 2: кто</p> <p>Вариант 3: когда</p> <p>Вариант 4: почему</p>	<p>прошло. Над полем ввода вариант 1 отображается сообщение «Заполните это поле.»</p>	<p>прошло. Над полем ввода вариант 1 отображается сообщение «Заполните это поле.»</p>	
5	<p>Текст вопроса: Выберите правильный перевод слова what</p> <p>Тип вопроса: Один правильный ответ</p> <p>Баллы за вопрос: 1</p> <p>Вариант 1: что</p> <p>Вариант 2: «Пустое поле»</p> <p>Вариант 3: когда</p> <p>Вариант 4: почему</p>	<p>Изменение вопроса не прошло. Над полем ввода вариант 2 отображается сообщение «Заполните это поле.»</p>	<p>Изменение вопроса не прошло. Над полем ввода вариант 2 отображается сообщение «Заполните это поле.»</p>	Тест пройден
6	<p>Текст вопроса: Выберите правильный перевод слова what</p> <p>Тип вопроса: Один правильный ответ</p> <p>Баллы за вопрос: 1</p> <p>Вариант 1: что</p> <p>Вариант 2: кто</p> <p>Вариант 3:</p>	<p>Изменение вопроса не прошло. Над полем ввода вариант 3 отображается сообщение «Заполните это поле.»</p>	<p>Изменение вопроса не прошло. Над полем ввода вариант 3 отображается сообщение «Заполните это поле.»</p>	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	«Пустое поле» Вариант 4: почему			
7	Текст вопроса: Выберите правильный перевод слова what Тип вопроса: Один правильный ответ Баллы за вопрос: 1 Вариант 1: что Вариант 2: кто Вариант 3: когда Вариант 4: «Пустое поле»	Изменение вопроса не прошло. Над полем ввода вариант 4 отображается сообщение «Заполните это поле.»	Изменение вопроса не прошло. Над полем ввода вариант 4 отображается сообщение «Заполните это поле.»	Тест пройден
8	Текст вопроса: Как переводится what и where Тип вопроса: Несколько правильных ответов Баллы за вопрос: 1 Вариант 1: что Вариант 2: где Вариант 3: когда Вариант 4: почему	Вопрос успешно изменен	Вопрос успешно изменен	Тест пройден
9	Текст вопроса: «Пустое поле» Тип вопроса: Несколько правильных ответов Баллы за вопрос: 1 Вариант 1: что Вариант 2: где Вариант 3: когда	Изменение вопроса не прошло. Над полем ввода текста вопроса отображается сообщение «Заполните это поле.»	Изменение вопроса не прошло. Над полем ввода текста вопроса отображается сообщение «Заполните это поле.»	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	Вариант 4: почему			
10	Текст вопроса: Как переводится what и where Тип вопроса: Несколько правильных ответов Баллы за вопрос: «Пустое поле» Вариант 1: что Вариант 2: где Вариант 3: когда Вариант 4: почему	Изменение вопроса не прошло. Над полем ввода балла за вопрос отображается сообщение «Заполните это поле.»	Изменение вопроса не прошло. Над полем ввода балла за вопрос отображается сообщение «Заполните это поле.»	Тест пройден
11	Текст вопроса: Как переводится what и where Тип вопроса: Несколько правильных ответов Баллы за вопрос: 1 Вариант 1: «Пустое поле» Вариант 2: где Вариант 3: когда Вариант 4: почему	Изменение вопроса не прошло. Над полем ввода вариант 1 отображается сообщение «Заполните это поле.»	Изменение вопроса не прошло. Над полем ввода вариант 1 отображается сообщение «Заполните это поле.»	Тест пройден
12	Текст вопроса: Как переводится what и where Тип вопроса: Несколько правильных ответов Баллы за вопрос: 1 Вариант 1: что Вариант 2:	Изменение вопроса не прошло. Над полем ввода вариант 2 отображается сообщение «Заполните это поле.»	Изменение вопроса не прошло. Над полем ввода вариант 2 отображается сообщение «Заполните это поле.»	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	«Пустое поле» Вариант 3: когда Вариант 4: почему			
13	Текст вопроса: Как переводится what и where Тип вопроса: Несколько правильных ответов Баллы за вопрос: 1 Вариант 1: что Вариант 2: где Вариант 3: «Пустое поле» Вариант 4: почему	Изменение вопроса не прошло. Над полем ввода вариант 3 отображается сообщение «Заполните это поле.»	Изменение вопроса не прошло. Над полем ввода вариант 3 отображается сообщение «Заполните это поле.»	Тест пройден
14	Текст вопроса: Как переводится what и where Тип вопроса: Несколько правильных ответов Баллы за вопрос: 1 Вариант 1: что Вариант 2: где Вариант 3: когда Вариант 4: «Пустое поле»	Изменение вопроса не прошло. Над полем ввода вариант 4 отображается сообщение «Заполните это поле.»	Изменение вопроса не прошло. Над полем ввода вариант 4 отображается сообщение «Заполните это поле.»	Тест пройден
15	Текст вопроса: Введите в алфавитном порядке b c a d Тип вопроса: Установление порядка Баллы за вопрос: 1 Элемент 1: b Элемент 2: c	Вопрос успешно изменен	Вопрос успешно изменен	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	Элемент 3: a Элемент 4: d			
16	Текст вопроса: «Пустое поле» Тип вопроса: Установление порядка Баллы за вопрос: 1 Элемент 1: b Элемент 2: c Элемент 3: a Элемент 4: d	Изменение вопроса не прошло. Над полем ввода текста вопроса отображается сообщение «Заполните это поле.»	Изменение вопроса не прошло. Над полем ввода текста вопроса отображается сообщение «Заполните это поле.»	Тест пройден
17	Текст вопроса: Ведите в алфавитном порядке b c a d Тип вопроса: Установление порядка Баллы за вопрос: «Пустое поле» Элемент 1: b Элемент 2: c Элемент 3: a Элемент 4: d	Изменение вопроса не прошло. Над полем ввода балла за вопрос отображается сообщение «Заполните это поле.»	Изменение вопроса не прошло. Над полем ввода балла за вопрос отображается сообщение «Заполните это поле.»	Тест пройден
18	Текст вопроса: Ведите в алфавитном порядке b c a d Тип вопроса: Установление порядка Баллы за вопрос: 1 Элемент 1: «Пустое поле» Элемент 2: c Элемент 3: a Элемент 4: d	Изменение вопроса не прошло. Над полем ввода элемент 1 отображается сообщение «Заполните это поле.»	Изменение вопроса не прошло. Над полем ввода элемент 1 отображается сообщение «Заполните это поле.»	Тест пройден
19	Текст вопроса: Ведите в алфавитном порядке b c a d Тип вопроса: Установление порядка	Изменение вопроса не прошло. Над полем ввода элемент 2 отображается сообщение	Изменение вопроса не прошло. Над полем ввода элемент 2 отображается сообщение	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	Баллы за вопрос: 1 Элемент 1: b Элемент 2: «Пустое поле» Элемент 3: a Элемент 4: d	«Заполните это поле.»	«Заполните это поле.»	
20	Текст вопроса: Ведите в алфавитном порядке b c a d Тип вопроса: Установление порядка Баллы за вопрос: 1 Элемент 1: b Элемент 2: c Элемент 3: «Пустое поле» Элемент 4: d	Изменение вопроса не прошло. Над полем ввода элемент 3 отображается сообщение «Заполните это поле.»	Изменение вопроса не прошло. Над полем ввода элемент 3 отображается сообщение «Заполните это поле.»	Тест пройден
21	Текст вопроса: Ведите в алфавитном порядке b c a d Тип вопроса: Установление порядка Баллы за вопрос: 1 Элемент 1: b Элемент 2: c Элемент 3: a Элемент 4: «Пустое поле»	Изменение вопроса не прошло. Над полем ввода элемент 4 отображается сообщение «Заполните это поле.»	Изменение вопроса не прошло. Над полем ввода элемент 4 отображается сообщение «Заполните это поле.»	Тест пройден
22	Текст вопроса: Напишите как переводится слово which Тип вопроса: Ввод ответа Баллы за вопрос: 1 Правильный ответ: который	Вопрос успешно изменен	Вопрос успешно изменен	Тест пройден
23	Текст вопроса: «Пустое поле» Тип вопроса:	Изменение вопроса не прошло. Над	Изменение вопроса не прошло. Над	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	Ввод ответа Баллы за вопрос: 1 Правильный ответ: который	полем ввода текста вопроса отображается сообщение «Заполните это поле.»	полем ввода текста вопроса отображается сообщение «Заполните это поле.»	
24	Текст вопроса: Напишите как переводится слово which Тип вопроса: Ввод ответа Баллы за вопрос: «Пустое поле» Правильный ответ: который	Изменение вопроса не прошло. Над полем ввода балла за вопрос отображается сообщение «Заполните это поле.»	Изменение вопроса не прошло. Над полем ввода балла за вопрос отображается сообщение «Заполните это поле.»	Тест пройден
25	Текст вопроса: Напишите как переводится слово which Тип вопроса: Ввод ответа Баллы за вопрос: 1 Правильный ответ: «Пустое поле»	Изменение вопроса не прошло. Над полем ввода правильный ответ отображается сообщение «Заполните это поле.»	Изменение вопроса не прошло. Над полем ввода правильный ответ отображается сообщение «Заполните это поле.»	Тест пройден
26	Текст вопроса: Сопоставьте правильный перевод слов Тип вопроса: Сопоставление Баллы за вопрос: 1 Пара 1 (Левая часть): what Пара 1 (Правая часть): что Пара 2 (Левая часть): where Пара 2 (Правая часть): где Пара 3 (Левая часть): which Пара 3 (Правая часть):	Вопрос успешно изменен	Вопрос успешно изменен	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	который Пара 4 (Левая часть): who Пара 4 (Правая часть): кто			
27	Текст вопроса: «Пустое поле» Тип вопроса: Сопоставление Баллы за вопрос: 1 Пара 1 (Левая часть): what Пара 1 (Правая часть): что Пара 2 (Левая часть): where Пара 2 (Правая часть): где Пара 3 (Левая часть): which Пара 3 (Правая часть): который Пара 4 (Левая часть): who Пара 4 (Правая часть): кто	Изменение вопроса не прошло. Над полем ввода текст вопроса отображается сообщение «Заполните это поле.»	Изменение вопроса не прошло. Над полем ввода текст вопроса отображается сообщение «Заполните это поле.»	Тест пройден
28	Текст вопроса: «Пустое поле» Тип вопроса: Сопоставление Баллы за вопрос: «Пустое поле» Пара 1 (Левая часть): what Пара 1 (Правая часть): что Пара 2 (Левая часть): where Пара 2 (Правая часть): где Пара 3 (Левая часть): which Пара 3 (Правая часть): который	Изменение вопроса не прошло. Над полем ввода баллы за вопрос отображается сообщение «Заполните это поле.»	Изменение вопроса не прошло. Над полем ввода баллы за вопрос отображается сообщение «Заполните это поле.»	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	Пара 4 (Левая часть): who Пара 4 (Правая часть): кто			
29	Текст вопроса: «Пустое поле» Тип вопроса: Сопоставление Баллы за вопрос: 1 Пара 1 (Левая часть): «Пустое поле» Пара 1 (Правая часть): что Пара 2 (Левая часть): where Пара 2 (Правая часть): где Пара 3 (Левая часть): which Пара 3 (Правая часть): который Пара 4 (Левая часть): who Пара 4 (Правая часть): кто	Изменение вопроса не прошло. Над полем ввода левой части пары 1 отображается сообщение «Заполните это поле.»	Изменение вопроса не прошло. Над полем ввода левой части пары 1 отображается сообщение «Заполните это поле.»	Тест пройден
30	Текст вопроса: «Пустое поле» Тип вопроса: Сопоставление Баллы за вопрос: 1 Пара 1 (Левая часть): what Пара 1 (Правая часть): «Пустое поле» Пара 2 (Левая часть): where Пара 2 (Правая часть): где Пара 3 (Левая часть): which Пара 3 (Правая часть): который	Изменение вопроса не прошло. Над полем ввода правой части пары 1 отображается сообщение «Заполните это поле.»	Изменение вопроса не прошло. Над полем ввода правой части пары 1 отображается сообщение «Заполните это поле.»	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	Пара 4 (Левая часть): who Пара 4 (Правая часть): кто			
31	Текст вопроса: «Пустое поле» Тип вопроса: Сопоставление Баллы за вопрос: 1 Пара 1 (Левая часть): what Пара 1 (Правая часть): что Пара 2 (Левая часть): «Пустое поле» Пара 2 (Правая часть): где Пара 3 (Левая часть): which Пара 3 (Правая часть): который Пара 4 (Левая часть): who Пара 4 (Правая часть): кто	Изменение вопроса не прошло. Над полем ввода левой части пары 2 отображается сообщение «Заполните это поле.»	Изменение вопроса не прошло. Над полем ввода левой части пары 2 отображается сообщение «Заполните это поле.»	Тест пройден
32	Текст вопроса: «Пустое поле» Тип вопроса: Сопоставление Баллы за вопрос: 1 Пара 1 (Левая часть): what Пара 1 (Правая часть): что Пара 2 (Левая часть): where Пара 2 (Правая часть): «Пустое поле» Пара 3 (Левая часть): which Пара 3 (Правая часть): который	Изменение вопроса не прошло. Над полем ввода правой части пары 2 отображается сообщение «Заполните это поле.»	Изменение вопроса не прошло. Над полем ввода правой части пары 2 отображается сообщение «Заполните это поле.»	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	Пара 4 (Левая часть): who Пара 4 (Правая часть): кто			
33	Текст вопроса: «Пустое поле» Тип вопроса: Сопоставление Баллы за вопрос: 1 Пара 1 (Левая часть): what Пара 1 (Правая часть): что Пара 2 (Левая часть): where Пара 2 (Правая часть): где Пара 3 (Левая часть): «Пустое поле» Пара 3 (Правая часть): который Пара 4 (Левая часть): who Пара 4 (Правая часть): кто	Изменение вопроса не прошло. Над полем ввода левой части пары 3 отображается сообщение «Заполните это поле.»	Изменение вопроса не прошло. Над полем ввода левой части пары 3 отображается сообщение «Заполните это поле.»	Тест пройден
34	Текст вопроса: «Пустое поле» Тип вопроса: Сопоставление Баллы за вопрос: 1 Пара 1 (Левая часть): what Пара 1 (Правая часть): что Пара 2 (Левая часть): where Пара 2 (Правая часть): где Пара 3 (Левая часть): which Пара 3 (Правая часть): «Пустое поле» Пара 4 (Левая	Изменение вопроса не прошло. Над полем ввода правой части пары 3 отображается сообщение «Заполните это поле.»	Изменение вопроса не прошло. Над полем ввода правой части пары 3 отображается сообщение «Заполните это поле.»	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	часть): who Пара 4 (Правая часть): кто			
35	Текст вопроса: «Пустое поле» Тип вопроса: Сопоставление Баллы за вопрос: 1 Пара 1 (Левая часть): what Пара 1 (Правая часть): что Пара 2 (Левая часть): where Пара 2 (Правая часть): где Пара 3 (Левая часть): which Пара 3 (Правая часть): который Пара 4 (Левая часть): «Пустое поле» Пара 4 (Правая часть): кто	Изменение вопроса не прошло. Над полем ввода левой части пары 4 отображается сообщение «Заполните это поле.»	Изменение вопроса не прошло. Над полем ввода левой части пары 4 отображается сообщение «Заполните это поле.»	Тест пройден
36	Текст вопроса: «Пустое поле» Тип вопроса: Сопоставление Баллы за вопрос: 1 Пара 1 (Левая часть): what Пара 1 (Правая часть): что Пара 2 (Левая часть): where Пара 2 (Правая часть): где Пара 3 (Левая часть): which Пара 3 (Правая часть): который Пара 4 (Левая часть): who	Изменение вопроса не прошло. Над полем ввода правой части пары 4 отображается сообщение «Заполните это поле.»	Изменение вопроса не прошло. Над полем ввода правой части пары 4 отображается сообщение «Заполните это поле.»	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
	Пара 4 (Правая часть): «Пустое поле»			

## 1.10. Студент

### 1.10.1. Возможность сдать практическую работу

Таблица 18 - Тест возможности сдать практическую работу

№	Действие (входные данные)	Ожидаeмый результат	Фактический результат	Статус теста (пройден/нe пройден)
1	2	3	4	5
1	Название файла: Практическая работа №1.docx Ссылка на файл: <a href="https://docs.google.com/document/d/19sCVEaF6-0WVC0TSLCf4WkcSuM4r8X6a/edit?usp=drive_link&amp;oid=104502585184625130868&amp;rtpof=true&amp;sd=true">https://docs.google.com/document/d/19sCVEaF6-0WVC0TSLCf4WkcSuM4r8X6a/edit?usp=drive_link&amp;oid=104502585184625130868&amp;rtpof=true&amp;sd=true</a>	Практическая работа сдана успешно	Практическая работа сдана успешно	Тест пройден
2	Название файла: «Пустое поле» Ссылка на файл: <a href="https://docs.google.com/document/d/19sCVEaF6-0WVC0TSLCf4WkcSuM4r8X6a/edit?usp=drive_link&amp;oid=104502585184625130868&amp;rtpof=true&amp;sd=true">https://docs.google.com/document/d/19sCVEaF6-0WVC0TSLCf4WkcSuM4r8X6a/edit?usp=drive_link&amp;oid=104502585184625130868&amp;rtpof=true&amp;sd=true</a>	Сдать практическую работу не удалось. Над полем ввода назван ия файла отобра жается сообще ние «Запол ните это поле.»	Сдать практическую работу не удалось. Над полем ввода назван ия файла отобра жается сообщение «Заполните это поле.»	Тест пройден
3	Название файла: Практическая работа №1.docx Ссылка на файл: «Пустое поле»	Сдать практическую работу не удалось. Над полем ввода	Сдать практическую работу не удалось. Над полем ввода	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
		ссылки на файл отображается сообщение «Заполните это поле.»	ссылки на файл отображается сообщение «Заполните это поле.»	

### 1.10.2. Возможность пройти тест

Таблица 19 - Тест возможности пройти тест

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
1	Даны ответы на все вопросы: Да	Тест пройден, происходит переход на страницу результатов теста	Тест пройден, происходит переход на страницу результатов теста	Тест пройден
2	Даны ответы на все вопросы: Нет	Тест пройден, происходит переход на страницу результатов теста	Тест пройден, происходит переход на страницу результатов теста	Тест пройден
3	Время, данное на прохождение теста, вышло	Попытка зачтена, отображается модульное окно с сообщением «Время истекло! Тест будет автоматически отправлен». По нажатию на «OK» происходит переход на страницу	Попытка зачтена, отображается модульное окно с сообщением «Время истекло! Тест будет автоматически отправлен». По нажатию на «OK» происходит переход на страницу	Тест пройден

№	Действие (входные данные)	Ожидаемый результат	Фактический результат	Статус теста (пройден/не пройден)
1	2	3	4	5
		страницу результатов теста	результатов теста	

## ПРИЛОЖЕНИЕ Г. Руководство пользователя

### АННОТАЦИЯ

Настоящее руководство пользователя предназначено для ознакомления с программой YumlSchool (далее – программа). Программа предназначена к использованию в учебных заведениях. Веб-приложение позволит оптимизировать учебный процесс и автоматизировать проведение практических работ и тестирования студентов по изученным темам.

## СОДЕРЖАНИЕ

1. Назначение программы .....	4
1.1. Назначение.....	4
1.2. Основные возможности программы .....	4
2. Условия выполнения программы .....	5
2.1. Уровень подготовки пользователя.....	5
2.2. Выполняемые функции .....	5
2.3. Программные и аппаратные требования к системе .....	5
2.3.1. Требования к программному обеспечению .....	5
2.3.2. Требования к аппаратному обеспечению.....	5
3. Выполнение программы .....	7
3.1. Подготовка к работе .....	7
3.1.1. Установка программы .....	7
3.1.2. Настройка программы .....	7
3.1.3. Запуск программы.....	7
3.2. Описание функций программы .....	8
3.2.1. Элементы основного окна программы .....	8
3.2.1.1. Элементы основного окна для преподавателя.....	8
3.2.1.2. Элементы основного окна для студента.....	8
3.2.1.3. Элементы основного окна для администратора БД.....	9
3.3. Основные функции программы.....	10
3.3.1. Функции неавторизованного пользователя .....	10
3.3.2. Функции преподавателя.....	12
3.3.3. Функции студента.....	25
3.3.4. Функции администратора БД .....	31

4. Возможные неполадки и способы устранения.....	36
5. Рекомендации по освоению .....	37

## 1. НАЗНАЧЕНИЕ ПРОГРАММЫ

### 1.1. Назначение

Настоящее руководство предназначено для ознакомления с программой YumlSchool. Программа предназначена к использованию в учебных заведениях. Веб-приложение позволит оптимизировать учебный процесс и автоматизировать проведение практических работ и тестирования студентов по изученным темам. Руководство содержит описание работы и сведения о функциях программы

### 1.2. Основные возможности программы

- Создание предметов в процессе учебы;
- Создание лекционных материалов, практических работ, тестов;
- Осуществление автоматической проверки тестов у студентов;
- Вывод данных об успеваемости студентов в формате PDF;
- Вывод результатов тестирования и проведения практических работ

## **2. УСЛОВИЯ ВЫПОЛНЕНИЯ ПРОГРАММЫ**

### **2.1. Уровень подготовки пользователя**

Пользователь программы должен иметь навыки работы с ПК, выполнять базовые операции на стандартных сайтах Google, Firefox, Edge.

### **2.2. Выполняемые функции**

Программа выполняет функции создания предметов в процессе учебы, создания лекционных материалов, практических работ, тестов, осуществления автоматической проверки тестов у студентов, вывода данных об успеваемости студентов в формате PDF, вывода результатов тестирования и проведения практических работ.

### **2.3. Программные и аппаратные требования к системе**

#### **2.3.1. Требования к программному обеспечению**

Программа предназначена для работы под управлением браузера (Google Chrome 80+, Mozilla Firefox 78+, Microsoft Edge 80+, Safari 12+)

#### **2.3.2. Требования к аппаратному обеспечению**

Программа предназначена для работы на персональных компьютерах, совместимых с компьютерами семейства x86, имеющих следующие характеристики:

- Процессор:

Минимум: 1 ГГц, двухъядерный процессор

Рекомендуется: 2 ГГц и выше, четырехъядерный процессор

- Оперативная память (RAM):

Минимум: 4 ГБ

Рекомендуется: 8 ГБ и выше

- Место на жестком диске:

Минимум: 500 МБ свободного пространства

Рекомендуется: 1 ГБ и выше

- Графическая карта:

Поддержка OpenGL 2.0 или выше

Рекомендуется: видеокарта с поддержкой современных графических API

- Система охлаждения:

Эффективная система охлаждения для предотвращения перегрева при длительной работе

- Сетевое соединение:

Рекомендуется подключение к интернету со скоростью не менее 1 Мбит/с для оптимальной работы

### 3. ВЫПОЛНЕНИЕ ПРОГРАММЫ

#### 3.1. Подготовка к работе

##### 3.1.1. Установка программы

Для того, чтобы установить WEB-приложение, необходимо перенести эксплуатационный пакет с флэш-накопителя в папку на компьютере, в которой будут храниться все файлы WEB приложения, после чего открыть эксплуатационный пакет, который установит файлы проекта в текущую директорию

##### 3.1.2. Настройка программы

Перед запуском WEB-приложения, необходимо произвести настройку:

Открыть папку, в которой хранится API, открыть конфигурационный файл и изменить в нем параметры строки подключения к базе данных, на те, что считаются действительными на компьютере (хост, порт, название базы данных, логин от сервера БД, пароль от сервера БД).

Установить библиотеки в виртуальную среду разработки pip, использовав команду pip install, перечень библиотек: asgiref, bcrypt, certifi, cffi, charset-normalizer, Django, idna, logger, rfc3986, requests, sqlparse, tzdata, urllib3

##### 3.1.3. Запуск программы

Чтобы произвести запуск WEB-приложения, необходимо:

- Провести запросы из скрипта БД на сервере БД, для создания таблиц и заполнения некоторых из них данными по умолчанию;
- Запустить проект API через среду разработки Microsoft Visual Studio 2022;
- Запустить WEB-приложение, запустив файл «.exe» в папке YumlSchoolWEB.

## 3.2. Описание функций программы

### 3.2.1. Элементы основного окна программы

#### 3.2.1.1. Элементы основного окна для преподавателя

Элементами основного окна для преподавателя являются (см. Рисунок 1), здесь видны:

- Данные о сессии пользователя, его фамилия и имя, кнопка «Выход» - для выхода из системы и возврата на страницу авторизации.
- Данные о предметах пользователя в которых он состоит. Здесь видны название предмета, группа, преподаватель предмета. По нажатию на предмет происходит переход на страницу предмета.

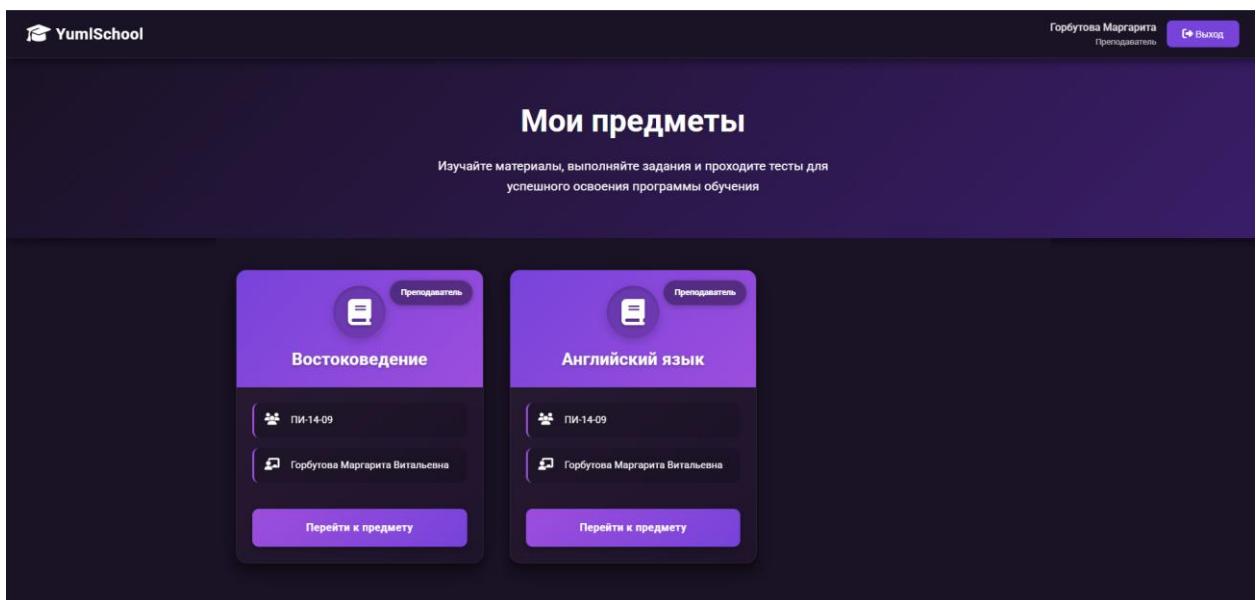


Рисунок 1 - Элементы основного окна у преподавателя

#### 3.2.1.2. Элементы основного окна для студента

Элементами основного окна для студента являются (см. Рисунок 2), здесь видны:

- Данные о сессии пользователя, его фамилия и имя, кнопка «Выход» - для выхода из системы и возврата на страницу авторизации.
- Данные о предметах пользователя в которых он состоит. Здесь видны название предмета, группа, преподаватель предмета. По нажатию на предмет происходит переход на страницу предмета.

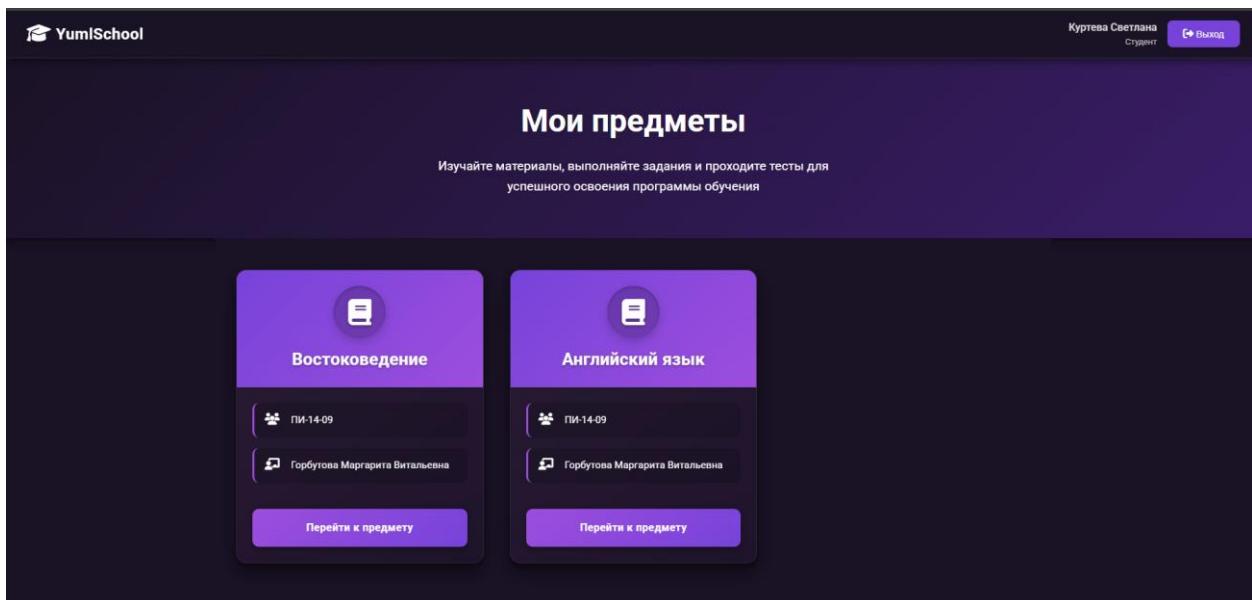


Рисунок 2 - Элементы основного окна у студента

### 3.2.1.3. Элементы основного окна для администратора БД

Элементами основного окна для администратора БД являются (см. Рисунок 2), здесь видны:

- Данные о сессии пользователя, его фамилия и имя, кнопка «Выход» - для выхода из системы и возврата на страницу авторизации.
- Административная панель, здесь:
  - Кнопка для перехода на страницу «Управление группами»
  - Кнопка для перехода на страницу «Управление пользователями»
  - Кнопка для перехода на страницу «Управление предметами»
- Выгрузка данных о группе в формате JSON.

Здесь необходимо нажать на кнопку «Выгрузить группы в JSON», которая выгрузит данные о группах;

Нажать на «Ваш файл» и выбрать файл с группами;

Нажать на «Загрузить группы из JSON» и данные вгрусятся.

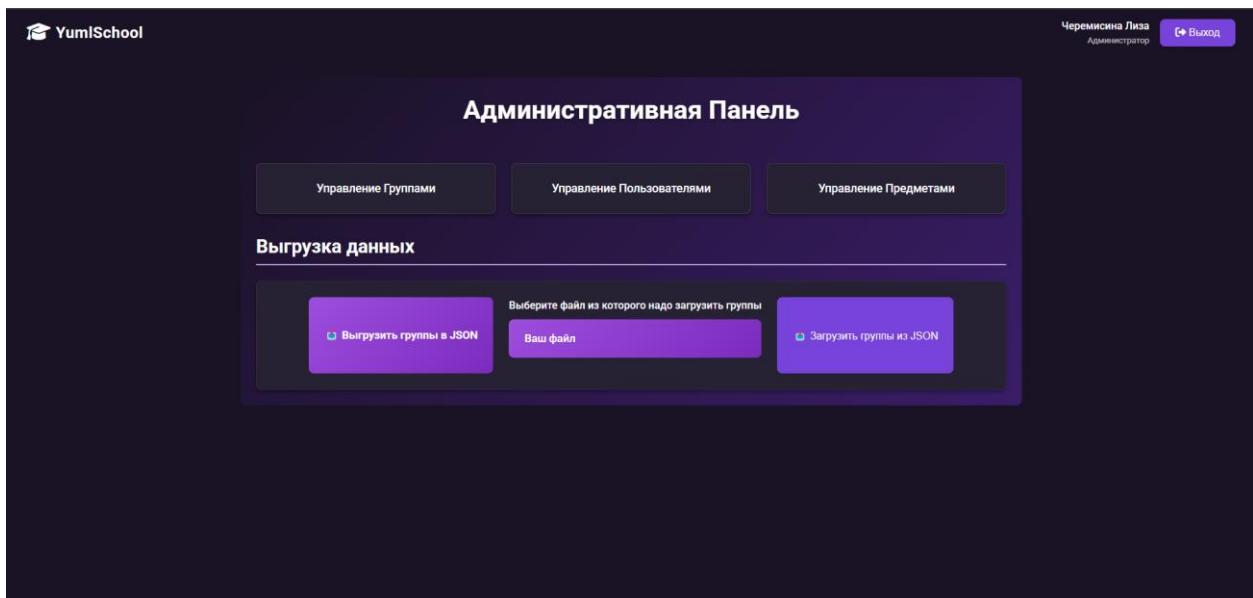


Рисунок 3 - Элементы основного окна у администратора БД

### 3.3. Основные функции программы

#### 3.3.1. Функции неавторизованного пользователя

После запуска программы и открытия сайта, пользователю доступна авторизация (Рисунок 4) и регистрация (Рисунок 5)

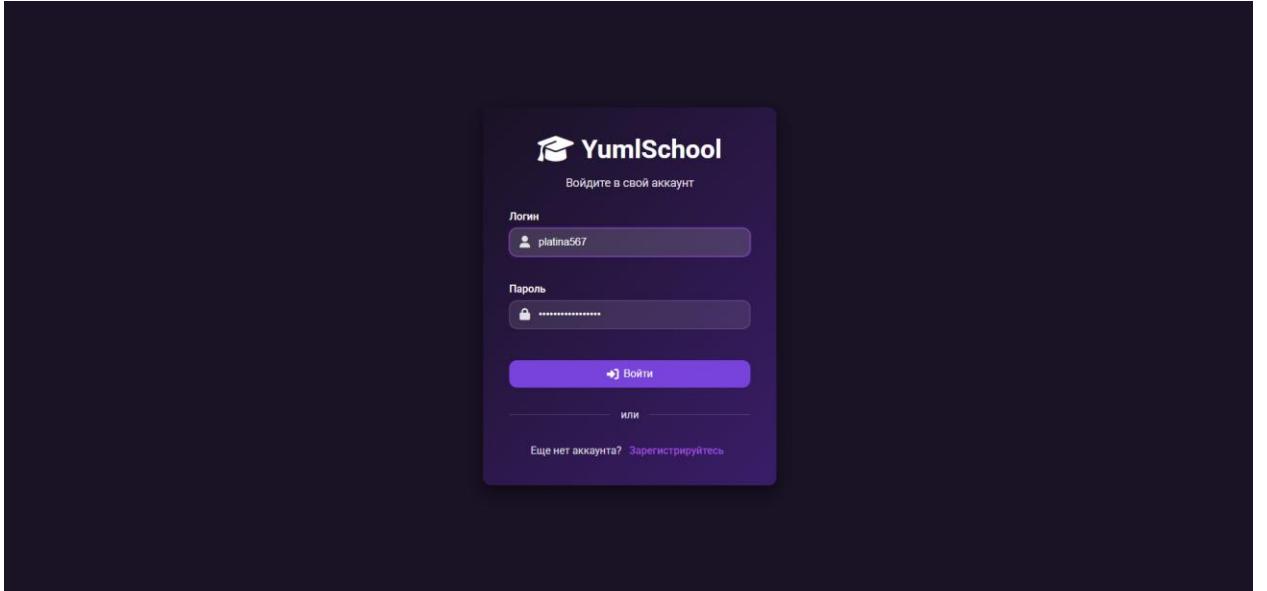


Рисунок 4 – Авторизация

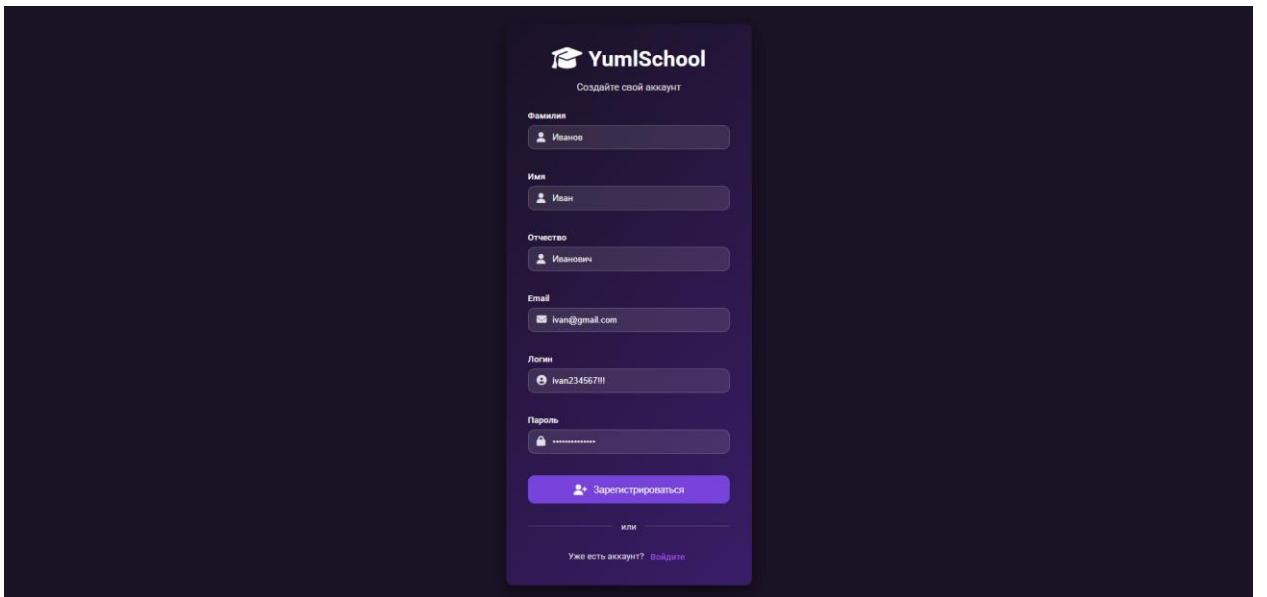


Рисунок 5 – Регистрация

После авторизации неавторизованный пользователь попадает на страницу с оповещением о том что пользователь пока не имеет роли и ему недоступны предметы.

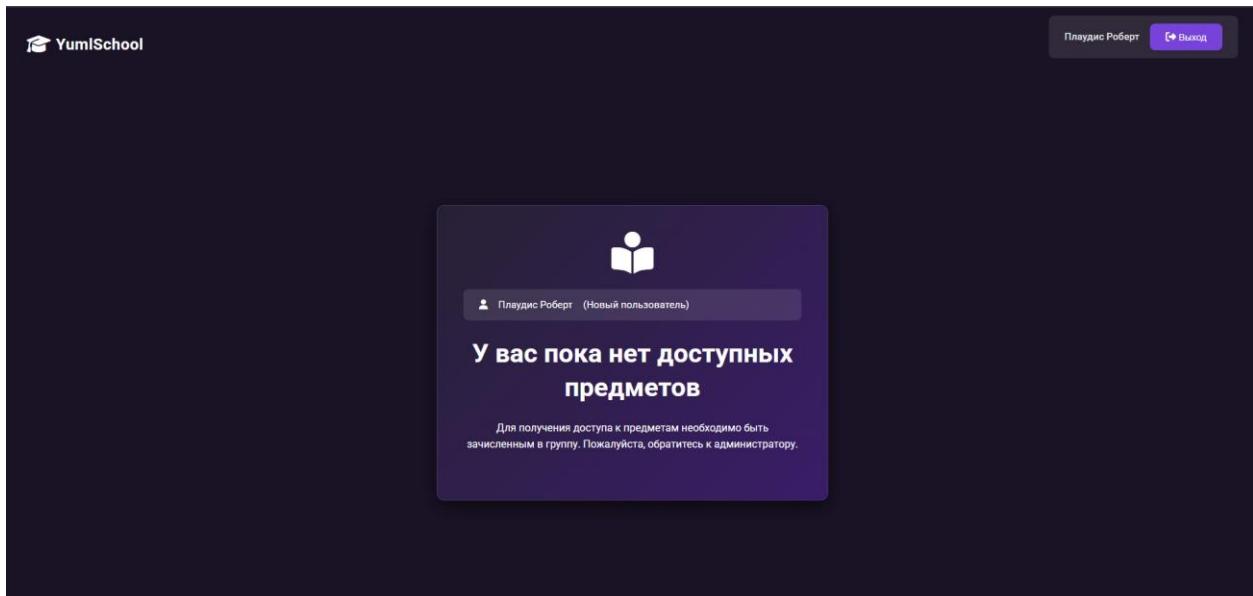


Рисунок 6 - Страница для неавторизованного пользователя

### 3.3.2. Функции преподавателя

#### 3.3.2.1. Функции элемента «Лекция»

На странице предметов у преподавателя есть доступ к лекциям. Они выводятся в блоке «Лекции»

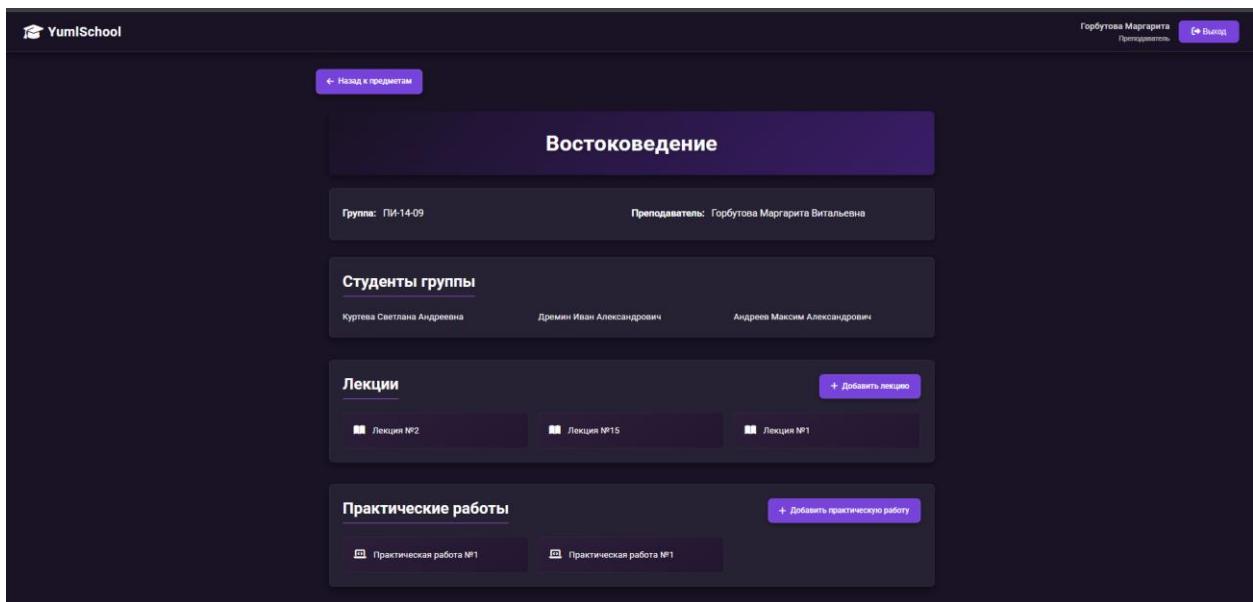


Рисунок 7 - Страница с лекциями

Для добавления лекции необходимо нажать на «Добавить лекцию». В открывшемся окне будут находиться поля для добавления лекции. После заполнения необходимо нажать на кнопку «Добавить лекцию»

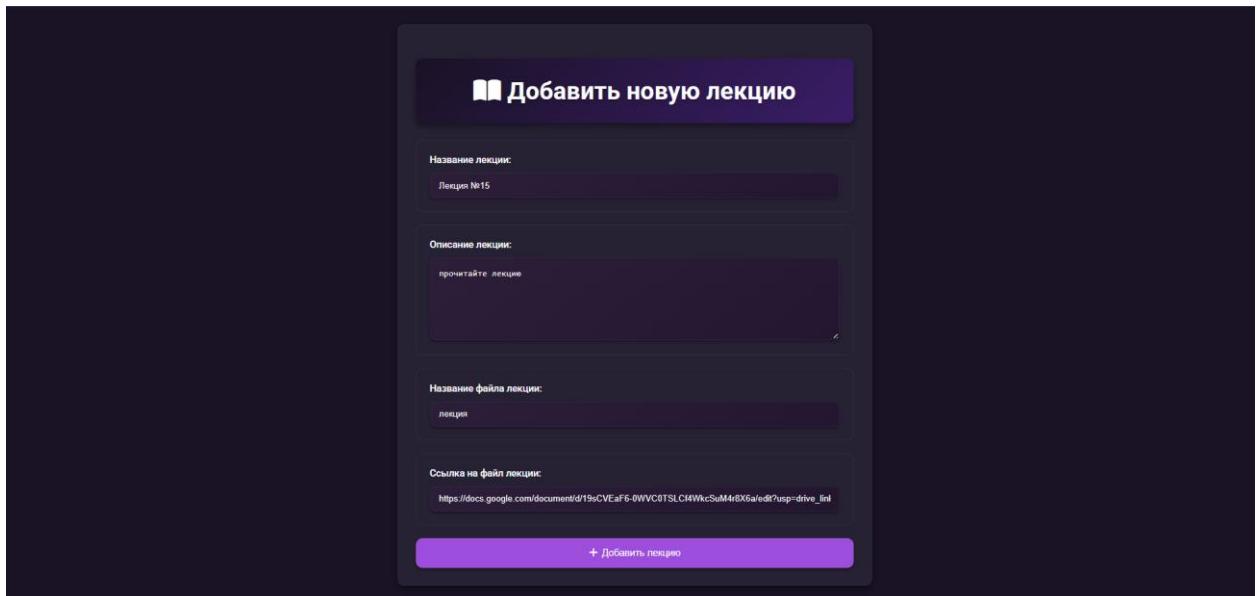


Рисунок 8 - Добавление лекции

По нажатию на лекцию на странице предметов осуществляется переход на страницу лекции где видна вся информация о лекции. Отсюда можно перейти к редактированию и удалению лекции.

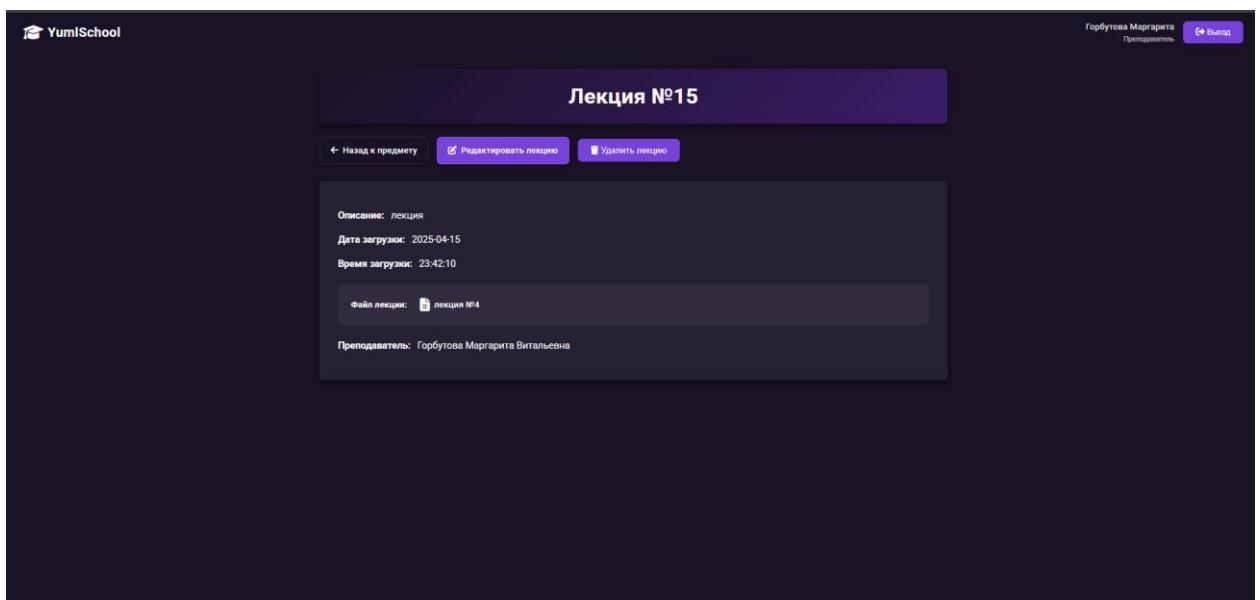


Рисунок 9 - Страница лекции

На странице редактирования лекции необходимо обновить данные о лекции а после нажать на кнопку «Сохранить изменения»

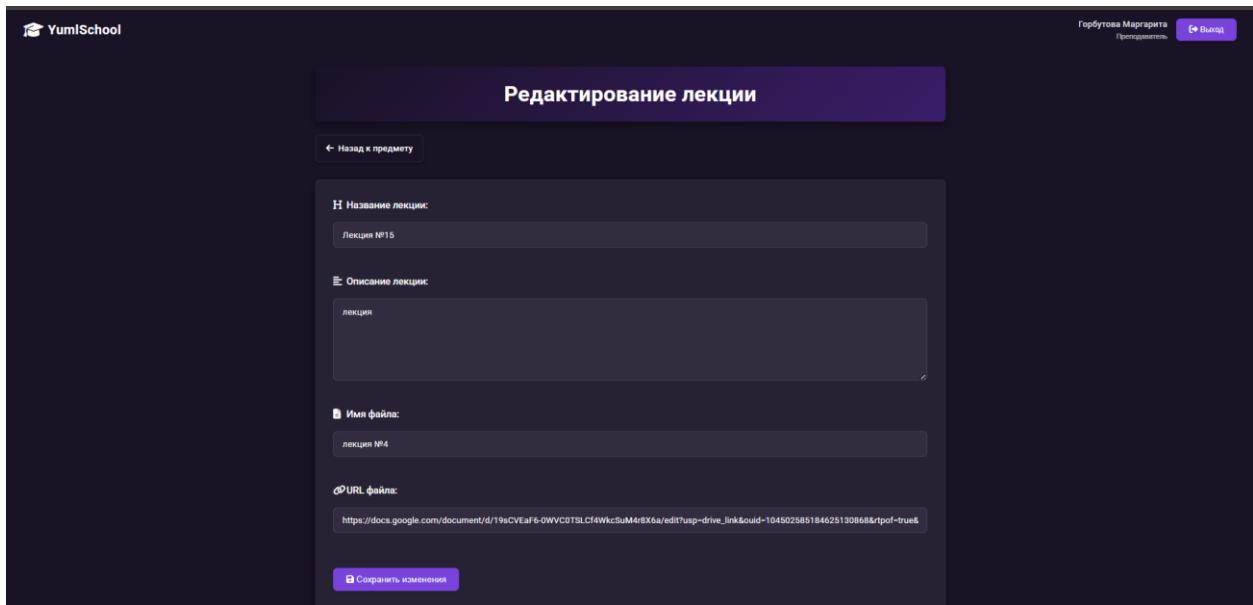


Рисунок 10 - Страница редактирования лекции

### 3.3.2.2. Функции элемента «Практическая работа»

Для работы с практическими работами преподавателю доступен блок «Лекции».

The screenshot shows the 'Лекции' (Lectures) section of the YumiSchool platform. It lists four lectures: 'Лекция №2', 'Лекция №15', 'Лекция №1', and 'Лекция №15'. Below this is a 'Практические работы' (Practical works) section with two entries: 'Практическая работа №1' and 'Практическая работа №1'. At the bottom is a 'Тесты' (Tests) section with three entries: 'Тест №1', 'Новый тест №1', and 'Новый тест №1 1234'. Each section has a purple '+ Добавить' (Add) button.

Рисунок 11 - Блок лекций

Для добавления работы необходимо нажать на кнопку «Добавить практическую работу». Далее, вписать данные в поля заполнения и нажать на кнопку «Добавить практическую работу».

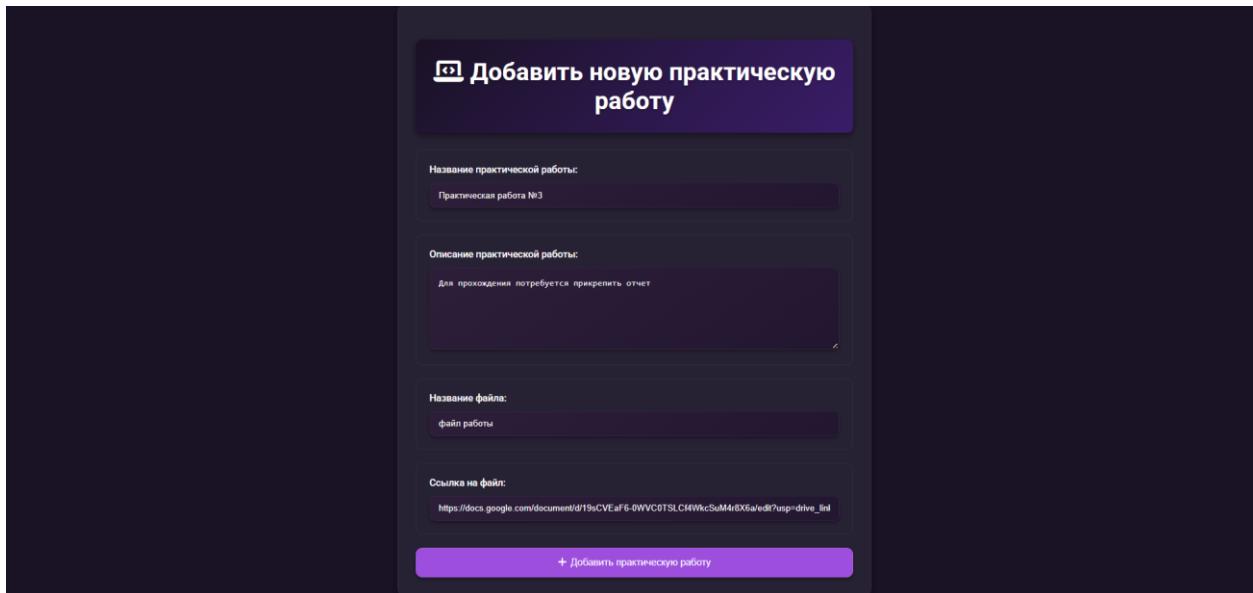


Рисунок 12 - Добавление практической работы

На странице практической работы можно удалить практическую работу, поставить оценку за сданные работы студентов, выгрузить информацию о практической работе и успеваемости студентов.

Студент	Последняя попытка	Статус	Оценка	Действия
Куртева Светлана Андреевна	2025-04-08 в 19:32:08	Задание сдано	5	<a href="#">Подробнее</a>

Рисунок 13 - Страница практической работы

Сданные работы можно проверить выбрав студента, раскрыв «Подробнее» и поставив оценку за работу

## Сданные работы студентов

Студент	Последняя попытка	Статус	Оценка	Действия
Куртева Светлана Андреевна	2025-04-08 в 19:32:08	Задание сдано	5	Скрыть

**Последняя работа студента**

Попытка 1 из 3 2025-04-08 в 19:32:08

Файл: отчет к работе 1

Статус: Зачтено (5)

Результат проверки:

Итоговая оценка: 5

Работа зачтена с оценкой 5

Рисунок 14 - Сданные работы студентов

Ниже находятся выгрузка данных об успеваемости и статистика в виде диаграммы о статусах работ

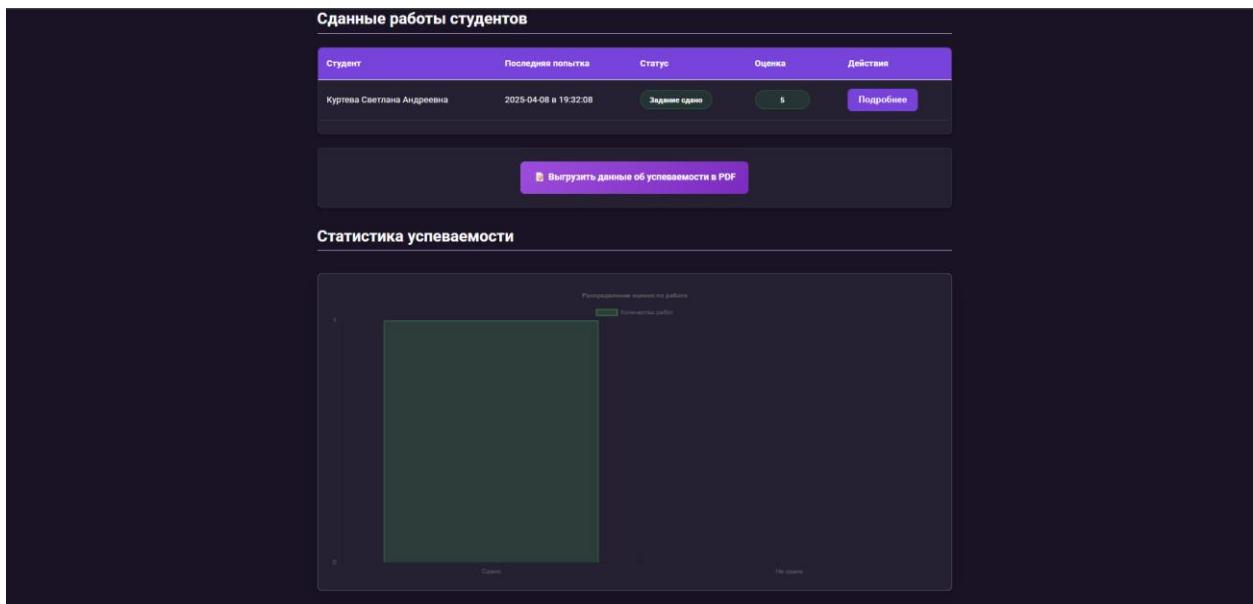


Рисунок 15 – Вывод диаграммы и файла PDF

### 3.3.2.3. Функции элемента «Тест»

На странице предмета размещается блок «Тесты».

Рисунок 16 - Страница с выводом тестов

На странице создания теста необходимо заполнить данные, ввести количество ограничения времени теста.

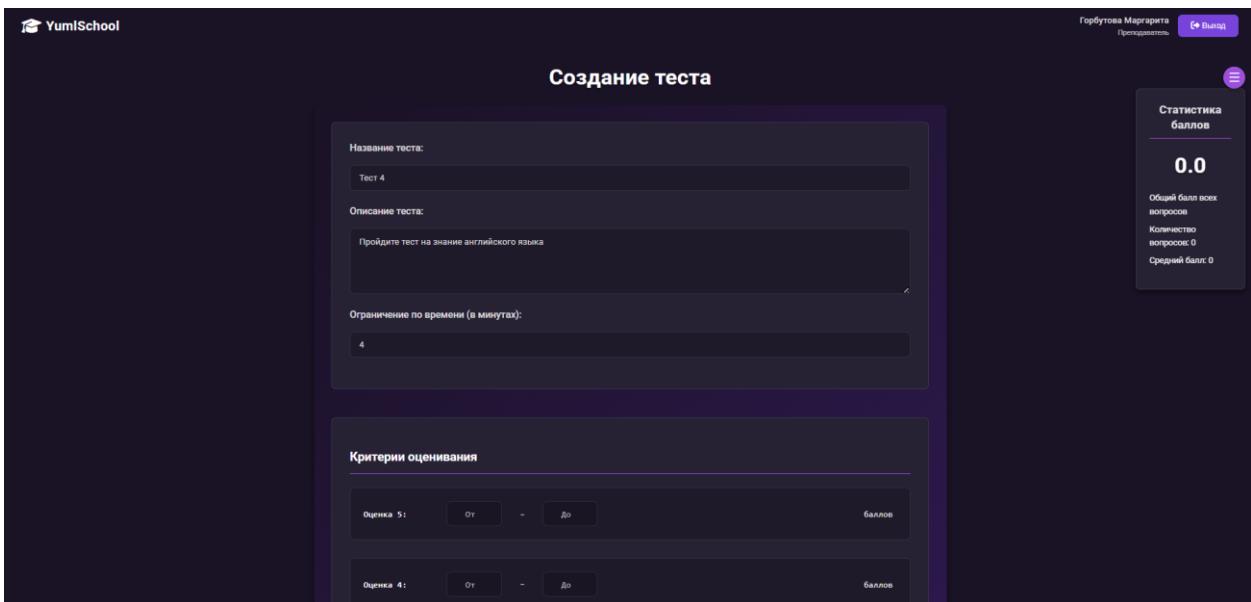


Рисунок 17 - Создание теста

Далее, необходимо проставить критерии теста – количество баллов соответствует оценке

Критерии оценивания

Оценка 5: 9 - 10 баллов

Оценка 4: 5 - 8 баллов

Оценка 3: 3 - 4 баллов

Рисунок 18 - Критерии оценивания

Вопросы делятся на пять типов:

- Один правильный ответ – один правильный ответ который надо отметить, нажав на галочку около ответа
- Несколько правильных ответов – Выбрать несколько правильных ответов, нажав на галочку около ответа
- Сопоставление – Ввести левую часть сходящуюся с правой
- Ввод ответа – Ввести правильный ответ
- Установление порядка – Установка порядка просто вписав ответы.

Данные уже установлены в порядке, у студента они перемешиваются

Скриншот интерфейса для создания вопроса. Виджет 'Вопрос #1' содержит следующие поля:

- Текст вопроса: как переводится what
- Тип вопроса: Один правильный ответ
- Баллы за вопрос: 2
- Вариант 1: что (radio button selected)
- Вариант 2: Как
- Вариант 3: где
- Вариант 4: Когда

Панель статистики включает:

- Статистика баллов: 2.0
- Общий балл всех вопросов
- Количество вопросов: 1
- Средний балл: 2.00

Рисунок 19 - Один правильный ответ

Скриншот интерфейса для создания вопроса. Виджет 'Вопрос #1' содержит следующие поля:

- Текст вопроса: как переводится what и where
- Тип вопроса: Несколько правильных ответов
- Баллы за вопрос: 2
- Вариант 1: что (checkbox)
- Вариант 2: когда (checkbox)
- Вариант 3: где (checkbox)
- Вариант 4: во сколько (checkbox)

Панель статистики включает:

- Статистика баллов: 2.0
- Общий балл всех вопросов
- Количество вопросов: 1
- Средний балл: 2.00

Рисунок 20 - Несколько правильных ответов

Скриншот интерфейса для создания вопроса. Виджет имеет темную тему с белыми элементами. В верхней части виджета есть заголовок "Вопрос #1" и красная кнопка "Удалить вопрос".

Поля ввода:

- Текст вопроса: "сопоставьте слова с их переводом"
- Тип вопроса: "Сопоставление"
- Баллы за вопрос: "2"
- Пара 1:
  - слово: "how"
  - перевод: "как"
- Пара 2:
  - слово: "where"
  - перевод: "где"
- Пара 3:
  - слово: "what"
  - перевод: "что"
- Пара 4:
  - слово: "which"
  - перевод: "который"

В правом верхнем углу виджета расположена статистика: "Статистика баллов" (2.0), "Общий балл всех вопросов" (2.0), "Количество вопросов" (1) и "Средний балл: 2.00".

Рисунок 21 - Сопоставление

Скриншот интерфейса для создания вопроса. Виджет имеет темную тему с белыми элементами. Виджет имеет темную тему с белыми элементами.

Поля ввода:

- Текст вопроса: "как переводится слово который"
- Тип вопроса: "Ввод ответа"
- Баллы за вопрос: "2"
- Правильный ответ: "which"

В нижней части виджета есть синяя кнопка "Добавить вопрос".

Рисунок 22 - Ввод ответа

The screenshot shows a dark-themed user interface for creating a test. On the left, there's a form for 'Вопрос #1' (Question #1). It includes fields for 'Текст вопроса:' (Text of the question) containing 'разместите буквы в конфигураторе', 'Тип вопроса:' (Type of question) set to 'Установление порядка' (Ordering), 'Баллы за вопрос:' (Points for the question) set to 2, and four sections labeled 'Элемент 1' through 'Элемент 4', each containing a letter ('a', 'b', 'c', 'd') and a small purple circular icon. At the top right is a red button labeled 'Удалить вопрос' (Delete question). On the far right, a sidebar displays 'Статистика баллов' (Score statistics) with a total score of 2.0, and a note that the total points for all questions are 2.00.

Рисунок 23 – Установление порядка

Внизу находятся кнопки для добавления вопросов и создания теста

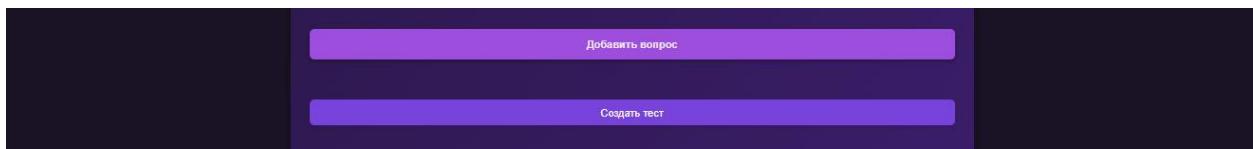


Рисунок 24 - Кнопки добавления вопроса и создания теста

После создания теста преподавателю доступна информация о тесте здесь:

- Информация – информация о тесте, доступно редактирование, результаты студентов, удаление.
- Вопросы – данные о вопросах
- Критерии оценивания – критерии оценивания теста
- Результаты – Переход на страницу результатов теста

The screenshot shows the YumiSchool platform interface for a test titled 'Тест №1'. At the top, there are user details: 'Горбутова Маргарита' (Prepared by), a 'Выход' (Logout) button, and a purple 'Информация' (Information) button. Below this, the test title 'Тест №1' is displayed, along with its description 'тест по теме востоковедение' (test on the topic of oriental studies), time limit '3 минут' (3 minutes), and subject 'Востоковедение' (Oriental studies). A navigation bar at the bottom includes 'Информация', 'Вопросы', 'Критерии оценивания', and 'Результаты'. The main content area is titled 'Информация о teste' (Information about the test) and contains fields for 'Название' (Name: Тест №1), 'Описание' (Description: тест по теме востоковедение), and 'Ограничение по времени' (Time limit: 3 минут). Below this are buttons for 'Редактировать тест' (Edit test), 'Результаты студентов' (Student results), 'Удалить тест' (Delete test), and 'Вернуться к предмету' (Return to subject).

Рисунок 25 – Информация

The screenshot shows the YumiSchool platform interface for the 'Вопросы теста' (Test questions) section of 'Тест №1'. At the top, there are user details: 'Горбутова Маргарита' (Prepared by), a 'Выход' (Logout) button, and a purple 'Вопросы' (Questions) button. Below this, the test title 'Тест №1' is displayed, along with its description 'тест по теме востоковедение' (test on the topic of oriental studies), time limit '3 минут' (3 minutes), and subject 'Востоковедение' (Oriental studies). A navigation bar at the bottom includes 'Информация', 'Вопросы', 'Критерии оценивания', and 'Результаты'. The main content area is titled 'Вопросы теста' (Test questions) and shows 'Вопрос 1' (Question 1) with the question 'что такое what?' and a note 'Один правильный ответ' (One correct answer). It lists four options: 'что', 'где', 'когда', and 'но сколько'. Below this is 'Вопрос 2' (Question 2) with a note 'Несколько правильных ответов' (Several correct answers).

Рисунок 26 – Вопросы

The screenshot shows the YumiSchool platform interface for the 'Критерии оценивания' (Grading criteria) section of 'Тест №1'. At the top, there are user details: 'Горбутова Маргарита' (Prepared by), a 'Выход' (Logout) button, and a purple 'Критерии оценивания' (Grading criteria) button. Below this, the test title 'Тест №1' is displayed, along with its description 'тест по теме востоковедение' (test on the topic of oriental studies), time limit '3 минут' (3 minutes), and subject 'Востоковедение' (Oriental studies). A navigation bar at the bottom includes 'Информация', 'Вопросы', 'Критерии оценивания', and 'Результаты'. The main content area is titled 'Критерии оценивания' (Grading criteria) and displays a table with four rows:

Оценка	Минимальные баллы	Максимальные баллы
5	9	10
4	5	8
3	3	4
2	Менее 3 баллов	

Рисунок 27 - Критерии оценивания

The screenshot shows a dark-themed interface for the YumiSchool platform. At the top left is the logo 'YumiSchool'. At the top right are user details: 'Горбутова Маргарита' and 'Преподаватель', with a 'Выход' (Logout) button. The main title 'Тест №1' is displayed above a purple header bar. Below the bar, it says 'ТЕСТ ПО ТЕМЕ ВОСТОКОВЕДЕНИЕ' and 'Время на выполнение: 3 минуты' (Time limit: 3 minutes). The subject is listed as 'Предмет: Востоковедение'. A navigation bar below the header includes tabs: 'Информация' (Information), 'Вопросы' (Questions), 'Критерии оценивания' (Evaluation criteria), and 'Результаты' (Results), with 'Критерии оценивания' being the active tab. A sub-section titled 'Результаты студентов' (Student results) is shown, containing a message: 'Для просмотра полной информации о результатах студентов перейдите на страницу результатов:' (To view full information about student results, go to the results page). A purple button labeled 'Просмотр результатов студентов' (View student results) is present.

Рисунок 28 - Результаты студентов

После перехода на результат тестирования преподавателю открывается страница со списком студентов проходивших тест и их результат

**Тест №1 - Результаты студентов**

тест по теме востоковедение

### Общая статистика

3 Всего студентов	1 Сдали тест	2 Не сдали тест
-------------------	--------------	-----------------

### Результаты по студентам

Студент	Попыток: 1	Лучший результат: 4 баллов	Оценка: 3.0
Светлана Куртева Андреевна	Попыток: 1	Лучший результат: 4 баллов	Оценка: 3.0
Иван Дремин Александрович	Попыток: 0	Нет попыток	
Максим Андреев Александрович	Попыток: 0	Нет попыток	

[Вернуться к тесту](#)

Рисунок 29 - Резульльтат тестирования

Из окна информации о teste можно нажать на «Редактировать teste» и в открывшемся окне поменять данные teste.

**YumiSchool**

Горбутова Маргарита  
Преподаватель [Выход](#)

### Редактирование teste

Название teste:  
Тест №1

Описание teste:  
тест по теме востоковедение

Ограничение по времени (в минутах):  
3

### Критерии оценивания

Оценка 5: 9 - 10 баллов

Оценка 4: 5 - 8 баллов

**Статистика баллов**

**10.0**

Общий балл всех вопросов  
Количество вопросов: 5  
Средний балл: 2.00

### 3.3.3. Функции студента

Студент может работать с данными предмета

The screenshot shows a dark-themed user interface for a student. At the top, there is a purple header bar with the title "Востоковедение". Below it, a white navigation bar contains the text "← Назад к предметам" on the left and "Группа: ПИ-14-09" and "Преподаватель: Горбутова Маргарита Витальевна" on the right. The main content area is divided into several sections: "Студенты группы" (Students of the group) listing three names: Куртева Светлана Андреевна, Дремин Иван Александрович, and Андреев Максим Александрович; "Лекции" (Lectures) showing four items: "Лекция №2", "Лекция №15", "Лекция №1", and "Лекция №15"; "Практические работы" (Practical works) showing two items: "Практическая работа №1" and "Практическая работа №1"; and "Тесты" (Tests) showing three items: "Тест №1", "Новый тест №1", and "Новый тест №1 1234".

Рисунок 30 - Предмет для студента

#### 3.3.3.1. Функции элемента «Лекция»

Для открытия лекций необходимо просто нажать по лекции. Здесь видна информация о ней и файл для просмотра

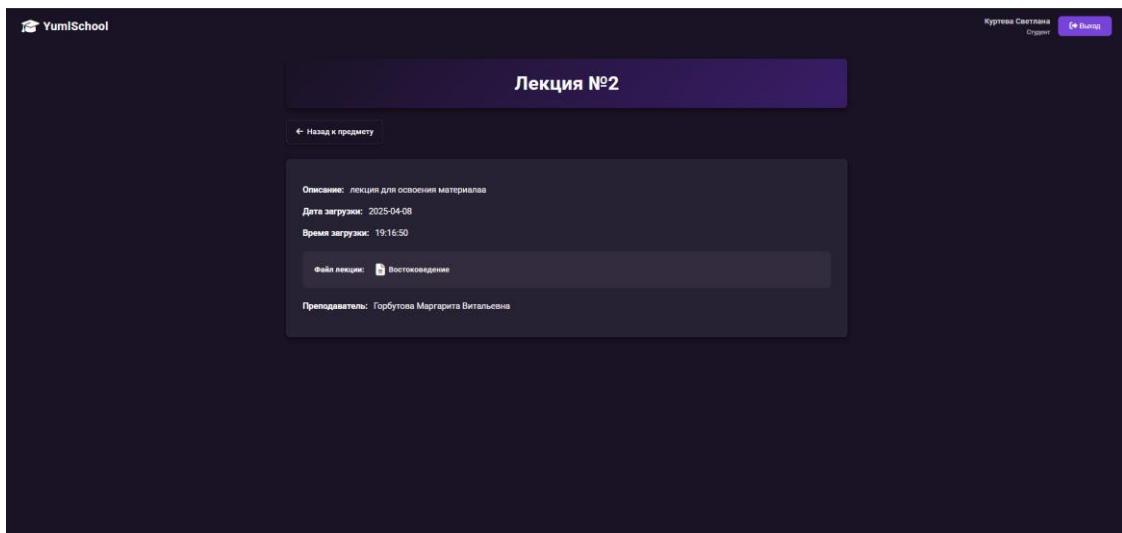


Рисунок 31 - Лекция у студента

### 3.3.3.2. Функции элемента «Практическая работа»

Для открытия практической работы студент должен нажать по ней. Студенту откроется информация о работе, к которой можно прикрепить файл отчета.

A screenshot of the YumiSchool platform interface. At the top, there's a dark header bar with the YumiSchool logo on the left and user information on the right. Below it is a white content area with a dark header titled 'Практическая работа: Практическая работа №1'. Underneath is a white content area containing a dark box with the following details:

- Описание: пройдите практическую работу
- Дата загрузки: 2025-04-23
- Время загрузки: 21:16:31
- Ссылка на файл: файл с текстом

A message box at the bottom left says: '⚠️ Вы еще не сдали работу по данной теме. Используйте кнопку ниже для добавления своей первой работы.' A blue button labeled 'Добавить работу' is visible. Below this is a section titled 'Статистика попыток' with the text 'Использовано попыток: 0 из 3' and three numbered circles (1, 2, 3). At the bottom, it says 'Номера использованных попыток:' followed by 'У вас осталось 3 попытки для сдачи этой работы.'

Рисунок 32 - Практическая работа

Далее, для добавления работы студенту необходимо заполнить данные и прикрепить работу нажав на «Добавить выполненную работу»

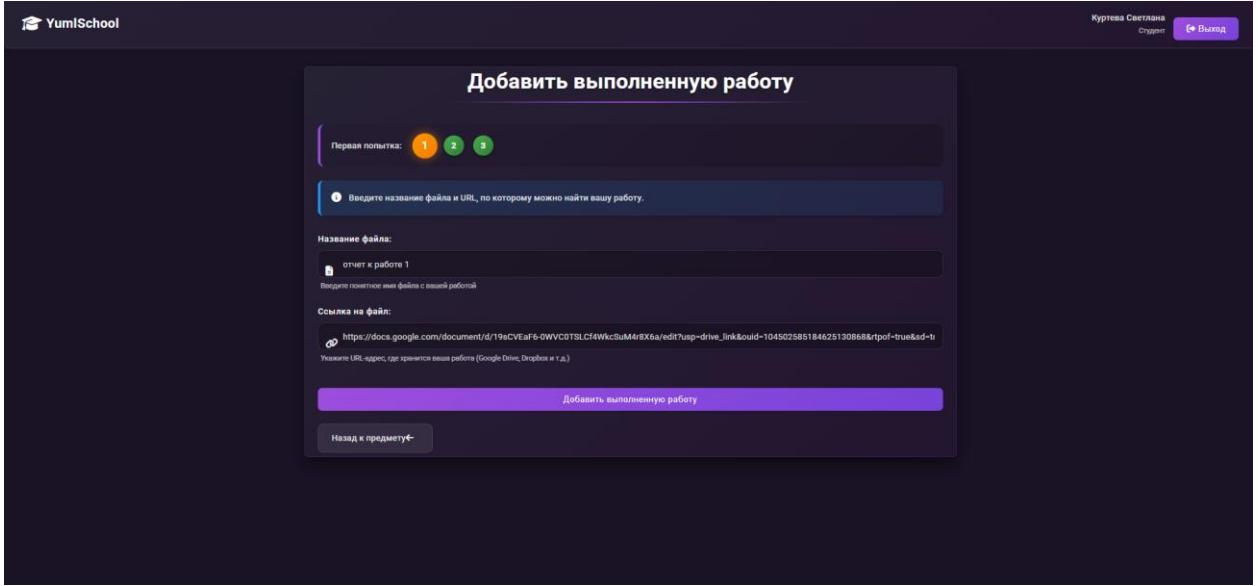


Рисунок 33 - Добавление работы

Далее студента оповещает система о сдаче работы.

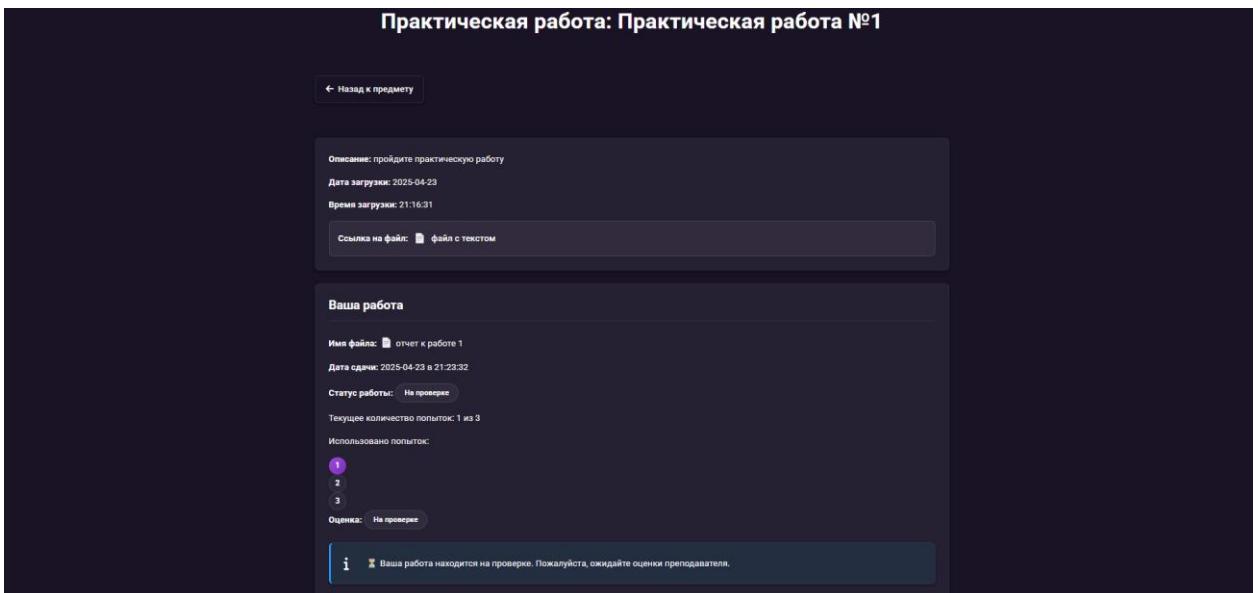


Рисунок 34 - Сдача практической работы

После проверки студента получает оценку за работу.

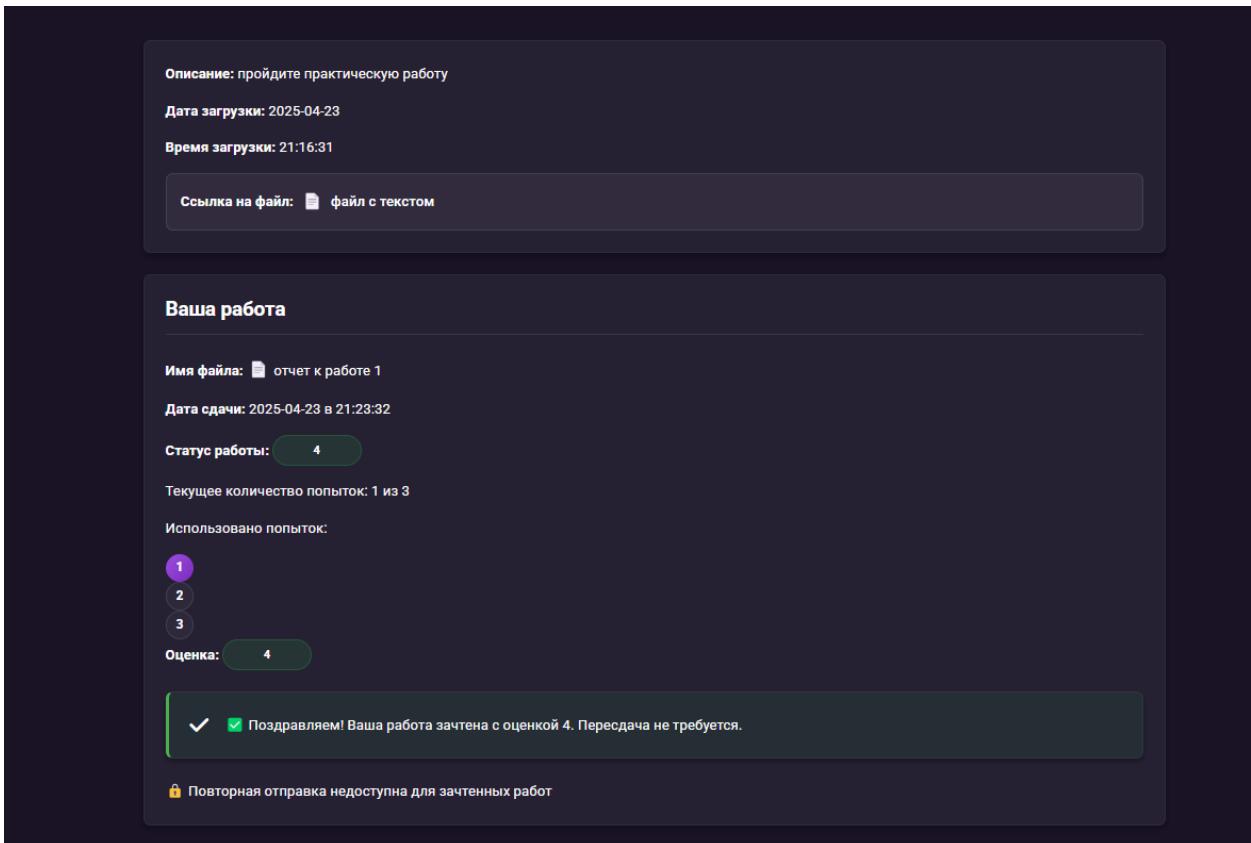


Рисунок 35 - Получение оценки

### 3.3.3.3. Функции элемента «Тест»

Для просмотра информации о тесте студенту необходимо нажать на тест в блоке тесты. Здесь видна информация о teste.

Рисунок 36 - Информация о teste

Во вкладке «Критерии оценивания» видна информация о баллах и соответствующих оценках

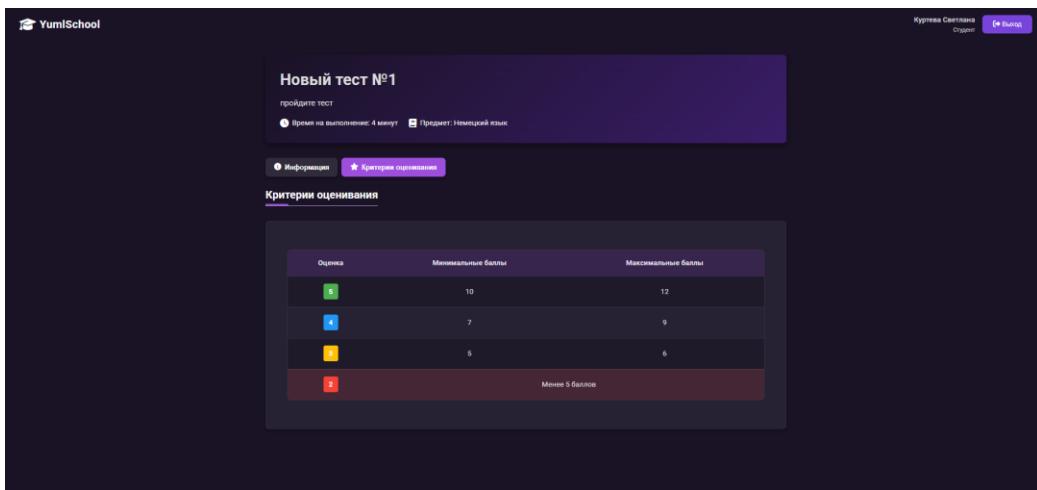


Рисунок 37 - Критерии оценивания

Для прохождения теста нажать на кнопку «Начать тест». При открытии теста начинается отчет времени до конца тестирования.

В вопросах типа «Ввод ответа» необходимо ввести правильный ответ.

В вопросах типа «Один вариант ответа» необходимо выбрать один вариант ответа.

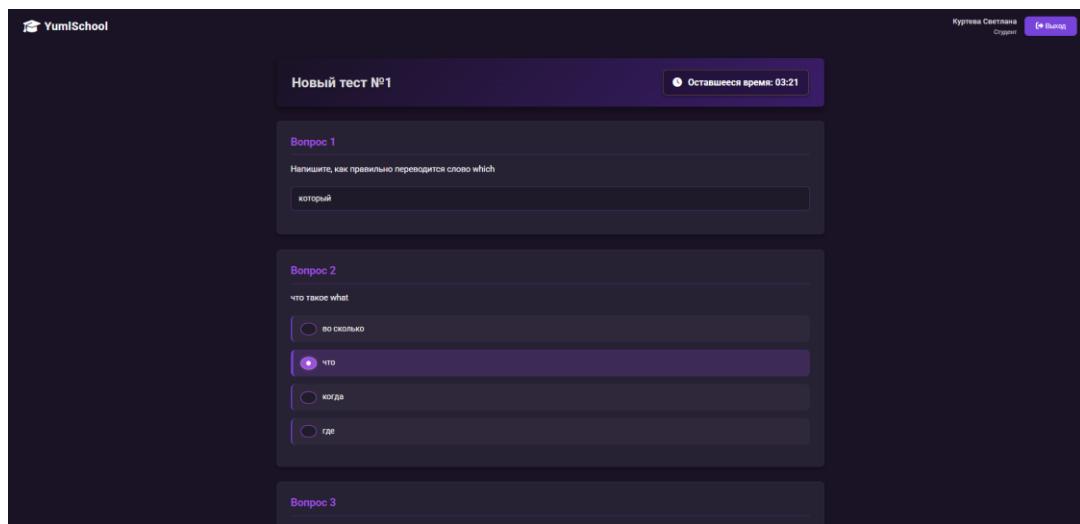


Рисунок 38 - Один правильный ответ

В вопросах типа «Несколько правильных ответов» необходимо выбрать несколько правильных ответов на вопрос.

В вопросах типа «Установление порядка» необходимо взять и перетянуть варианты ответа между собой.

The screenshot shows two questions from a digital test. Question 3 asks: "как переводится what и where". The options are: когда (checkbox), как (checkbox), где (checkbox, selected), and что (checkbox). Question 4 asks: "расставьте буквы в алфавитном порядке". It shows four letters: а, б, в, г, each with a corresponding number 1, 2, 3, 4 respectively. A text input field above the letters says: "Расположите элементы в правильном порядке, перетасовав их."

Рисунок 39 - Несколько правильных ответов и установление порядка  
В поросах типа «Сопоставление» необходимо совместить один вариант  
ответа с другим.

The screenshot shows a matching exercise titled "Вопрос 5" with the instruction: "сопоставьте правильно перевод слов". It lists four words on the left and four Russian words on the right, with green lines connecting them: "what" to "который", "where" to "что", "how" to "где", and "which" to "как". Below the list is a red button labeled "Собрать соединения" and a status message "4 из 4 соединений". At the bottom is a purple button labeled "Завершить тест".

Рисунок 40 – Сопоставление  
После прохождения теста доступны результаты

**Результаты теста "Новый тест №1"**

**Ваш последний результат**

★ Набрано баллов: 12

оценка: 5.0

Время завершения: 2025-04-23 22:24:03

Вы успешно сдали тест! Повторное прохождение невозможно.

**Критерии оценивания**

оценка	минимальные баллы	максимальные баллы
5	10	12
4	7	9
3	5	6
2	Менее 5 баллов	

Рисунок 41 - Результаты теста

Внизу расположена оценка за тест

Вы успешно сдали тест! Повторное прохождение невозможно.

**Критерии оценивания**

оценка	минимальные баллы	максимальные баллы
5	10	12
4	7	9
3	5	6
2	Менее 5 баллов	

**История попыток**

дата	время	баллы	оценка
2025-04-23	22:24:03	12	5.0

[← Вернуться к материалам теста](#)   [↻ Вернуться к предмету](#)

Рисунок 42 - Оценка за тест

### 3.3.4. Функции администратора БД

Администратору доступна выгрузка данных о группах, для этого необходимо воспользоваться кнопкой выгрузки, а для загрузки данных из JSON администратор нажимает на кнопку «Ваш файл» и выбрать файл для загрузки. После этого нажать на кнопку загрузки.

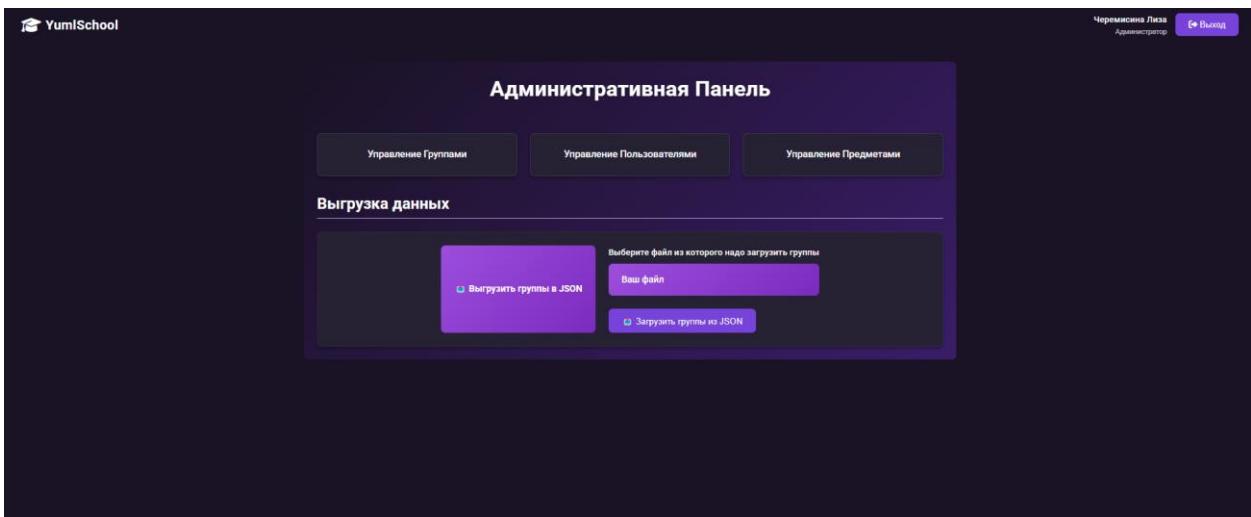


Рисунок 43 - Выгрузка данных

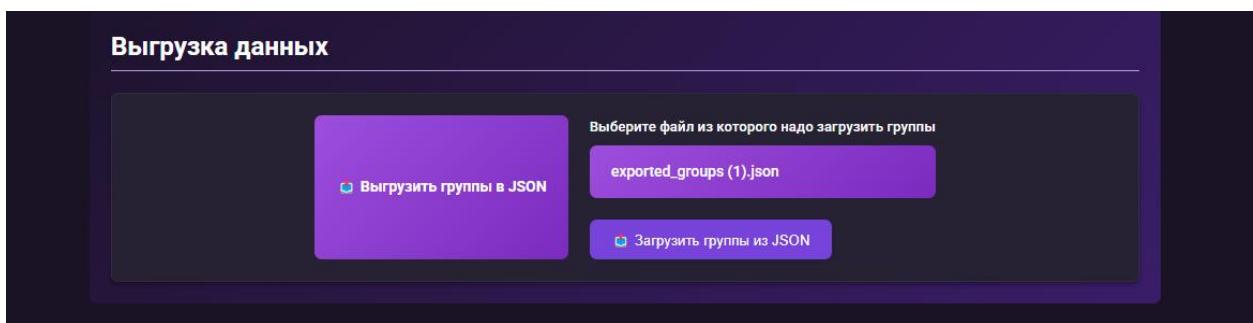


Рисунок 44 - Загрузка данных

По нажатию на кнопку «Управление группами» администратору доступны данные о группах. Здесь можно добавить, изменить, удалить данные о группах.

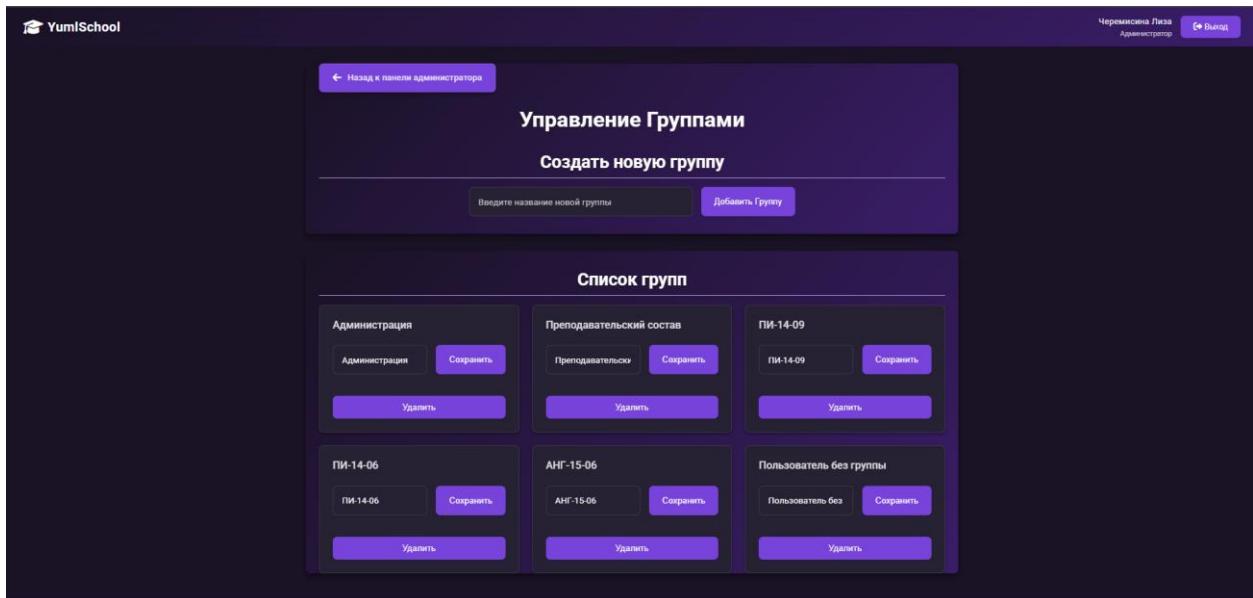


Рисунок 45 - Управление группами

По нажатию на кнопку «Управление пользователями» администратору доступны данные о пользователях. Здесь можно добавить, изменить, удалить данные о пользователях.

Ляхов Григорий Алексеевич

Уткин Дмитрий Олегович

Дремин Иван Александрович

Рисунок 46 - Управление пользователями

По нажатию на кнопку «Управление предметами» администратору доступны данные о предметах. Здесь можно добавить, изменить, удалить данные о предметах.

Название предмета:

Группа:

Преподаватель:

Добавить предмет

Список предметов

Востоковедение

Группа: ПИ-14-09  
Преподаватель: Горбутова Маргарита Витальевна

Изменить Удалить

Английский язык

Группа: ПИ-14-09  
Преподаватель: Горбутова Маргарита Витальевна

Изменить Удалить

Немецкий язык

Группа: ПИ-14-09  
Преподаватель: Горбутова Маргарита Витальевна

Изменить Удалить

Рисунок 47 - Управление предметами

Так выглядит редактирование предмета (Рисунок 48). Здесь необходимо изменить данные о предмете и сохранить изменения

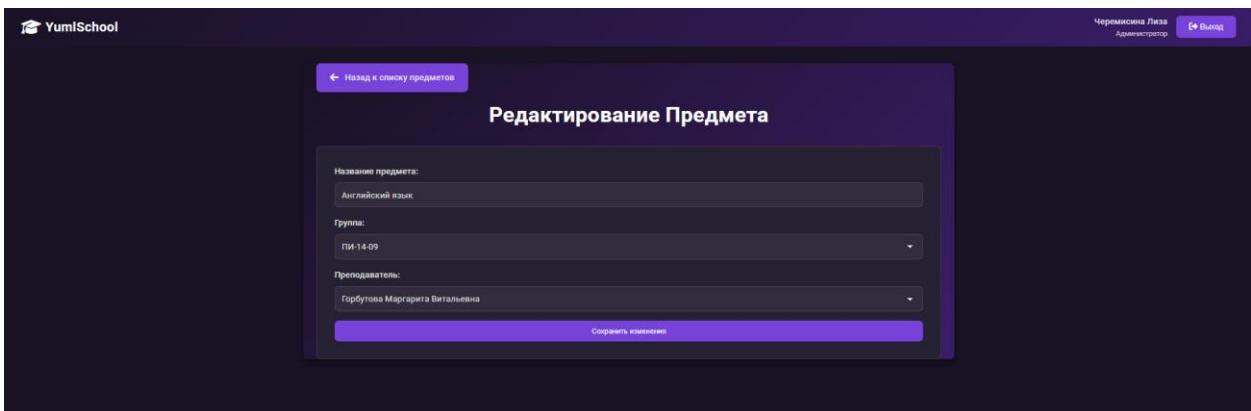


Рисунок 48 - Изменение предметов

#### 4. ВОЗМОЖНЫЕ НЕПОЛАДКИ И СПОСОБЫ УСТРАНЕНИЯ

№	Неполадки	Способ устранения
1	Не загружается страница. Ошибка «Нет соединения»	1. Проверить наличие подключения устройства к сети, повторить подключение к странице

## 5. РЕКОМЕНДАЦИИ ПО ОСВОЕНИЮ

Пример файла «.json» для загрузки данных о группе в базу данных



Рисунок 49 – Файл для загрузки данных

Пример файла об успеваемости в формате PDF из выгрузки данных практической работы

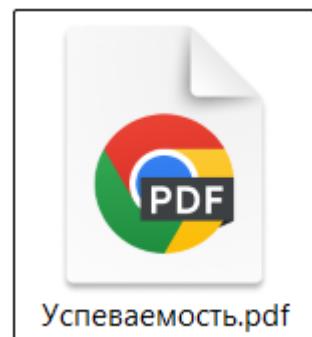


Рисунок 50 - Файл выгрузки pdf данных

## ПРИЛОЖЕНИЕ Д. Скрипт базы данных

### АННОТАЦИЯ

В данном разделе представлен скрипт базы данных для Web-приложения «YumlSchool»

#### 1. Скрипт базы данных:

- В данном разделе ожидается наименование скрипта, область применения скрипта и скрипт базы данных.

## СОДЕРЖАНИЕ

1.1. Наименование скрипта .....	3
1.2. Область применения скрипта .....	3
1.3. Словарь данных базы данных .....	3
1.4. Скрипт .....	8

# 1. СКРИПТ БАЗЫ ДАННЫХ

## 1.1. Наименование скрипта

Наименование – «yumlsschooldb.sql».

## 1.2. Область применения скрипта

Скрипт предназначен для программы «yumlsschool».

## 1.3. Словарь данных базы данных

Таблица 1. Словарь

Ключ	Поле	Тип данных	Обязательность заполнения	Описание
1	2	3	4	5
PK	ID_TestAttempt	Int	Not Null	Идентификатор попытки теста
FK	Test_ID	int	Not Null	Код теста
FK	User_ID	int	Not Null	Код пользователя
	Start_Time	datetime	Not Null	Начало времени
	End_Time	datetime	Не обязательно	Конец времени
	Status	varchar(20)	Not Null	Статус
	Score	int	Not Null	Выполненная работа
	Grade	decimal(3, 1)	Not Null	Оценка за тест
PK	ID_Criteria	int	Not Null	Идентификатор критерия теста
FK	ID_Test	int	Not Null	Код теста
	Grade	int	Not Null	Оценка критерия

	Min_Points	int	Not Null	Минимальное количество баллов
	Max_Points	int	Not Null	Максимальное количество баллов
PK	ID_Test	int	Not Null	Идентификатор теста
	Name_Test	varchar(255)	Not Null	Название
	Description_Test	varchar(100)	Не обязательно	Описание
	DateUpload_Test	date	Не обязательно	Дата выдачи
	TimeUpload_Test	time(7)	Не обязательно	Время выдачи
	TimeLimit	int	Не обязательно	Лимит теста
FK	Subject_ID	int	Не обязательно	Код предмета
FK	User_ID	int	Не обязательно	Код пользователя
PK	ID_LectionMaterial	int	Not Null	Идентификатор лекции
	Name_LectionMaterial	varchar(64)	Not Null	Название лекции
	Description_LectionMaterial	varchar(1024)	Not Null	Описание
	DateUpload_LectionMaterial	date	Не обязательно	Дата выдачи

	TimeUpload_LectureMaterial	time(7)	Не обязательно	Время выдачи
	Name_FileLecture	varchar(64)	Not Null	Название для файла лекции
	URL_FileLecture	varchar(1024)	Not Null	URL лекции
FK	User_ID	int	Не обязательно	Код пользователя
FK	Subject_ID	int	Не обязательно	Код предмета
PK	ID_Subject	int	Not Null	Идентификатор предмета
	Name_Subject	varchar(64)	Not Null	Название предмета
FK	ID_Group	int	Not Null	Код группы
FK	User_ID	int	Not Null	Код пользователя
PK	ID_Group	int	Not Null	Идентификатор группы
	Name_Group	varchar(50)	Not Null	Название группы
PK	ID_Question	int	Not Null	Идентификатор вопроса в тесте
	Question_Text	varchar(1000)	Not Null	Текст вопроса
	QuestionType	int	Not Null	Тип вопроса
	AnswerVariant	varchar(MAX)	Not Null	Вариант ответа

	CorrectAnswer	varchar(MA X)	Not Null	Правильный ответ
	Points	int	Не обязательно	Баллы за тест
FK	ID_Test	int	Не обязательно	Код теста
PK	ID_User	int	Not Null	Идентификатор пользователя
	FirstName	varchar(64)	Not Null	Имя
	SecondName	varchar(64)	Not Null	Фамилия
	MiddleName	varchar(64)	Не обязательно	Отчество
	Email	varchar(255)	Not Null	Почта
FK	ID_Role	int	Not Null	Код роли
FK	ID_Group	int	Not Null	Код группы
	Login	varchar(150)	Not Null	Логин
	Password	varchar(150)	Not Null	Пароль
PK	ID_Group	int	Not Null	Идентификатор группы
	Name_Group	varchar(50)	Not Null	Название группы
PK	ID_Role	int	Not Null	Идентификатор роли
	Role_Name	varchar(30)	Not Null	Название роли
PK	ID_Status	int	Not Null	Идентификатор статуса
	Name_Status	varchar(64)	Not Null	Название статуса

PK	ID_CompletedWork	int	Not Null	Идентификатор выполненной работы
	DateUpload_CompletedWork	date	Не обязательно	Дата сдачи
	TimeUpload_CompletedWork	time(7)	Не обязательно	Время сдачи
	Name_FileCompletedWork	varchar(64)	Not Null	Название файла
	URL_FileCompletedWork	varchar(300)	Not Null	URL файла
	Grade	varchar(64)	Not Null	Оценка
FK	User_ID	int	Не обязательно	Код пользователя
FK	PracticalWork_ID	int	Не обязательно	Код практической работы
FK	Status_ID	int	Не обязательно	Статус работы
	AttemptCount	int	Не обязательно	Количество попыток
PK	ID_PracticalWork	int	Not Null	Идентификатор практической работы
	Name_PracticalWork	varchar(64)	Not Null	Название практической
	Description_PracticalWork	varchar(300)	Not Null	Описание

	DateUpload_PracticalWork	date	Не обязательно	Дата выдачи
	TimeUpload_PracticalWork	time(7)	Не обязательно	Время выдачи
	Name_FilePracticalWork	varchar(64)	Not Null	Название файла
	URL_FilePracticalWork	varchar(3000)	Not Null	URL файла
FK	User_ID	int	Не обязательно	Код пользователя
FK	Subject_ID	int	Не обязательно	Код предмета

#### 1.4. Скрипт

```

CREATE DATABASE yumlsschooldb
USE [yumlsschooldb]
GO
***** Object: Table [dbo].[CompletedWork]  Script Date: 20.03.2025 19:12:47 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[CompletedWork](
    [ID_CompletedWork] [int] IDENTITY(1,1) NOT NULL,
    [DateUpload_CompletedWork] [date] NULL,
    [TimeUpload_CompletedWork] [time](7) NULL,
    [Name_FileCompletedWork] [varchar](64) NOT NULL,
    [URL_FileCompletedWork] [varchar](3000) NOT NULL,
    [Grade] [varchar](64) NOT NULL,
    [User_ID] [int] NULL,
    [PracticalWork_ID] [int] NULL,
    [Status_ID] [int] NULL,
    [AttemptCount] [int] NULL,
PRIMARY KEY CLUSTERED
(
    [ID_CompletedWork] ASC
)

```

```

)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF)
ON [PRIMARY]
) ON [PRIMARY]
GO
***** Object: Table [dbo].[GradingCriteria]  Script Date: 20.03.2025 19:12:47 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[GradingCriteria](
[ID_Criteria] [int] IDENTITY(1,1) NOT NULL,
[ID_Test] [int] NOT NULL,
[Grade] [int] NOT NULL,
[Min_Points] [int] NOT NULL,
[Max_Points] [int] NOT NULL,
PRIMARY KEY CLUSTERED
(
    [ID_Criteria] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF)
ON [PRIMARY]
) ON [PRIMARY]
GO
***** Object: Table [dbo].[Group]  Script Date: 20.03.2025 19:12:47 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Group](
[ID_Group] [int] IDENTITY(1,1) NOT NULL,
[Name_Group] [varchar](50) NOT NULL,
PRIMARY KEY CLUSTERED
(
    [ID_Group] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF)
ON [PRIMARY]
) ON [PRIMARY]
GO
***** Object: Table [dbo].[LectureMaterial]  Script Date: 20.03.2025 19:12:47 *****/
SET ANSI_NULLS ON

```

```

GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[LectionMaterial](
    [ID_LectionMaterial] [int] IDENTITY(1,1) NOT NULL,
    [Name_LectionMaterial] [varchar](64) NOT NULL,
    [Description_LectionMaterial] [varchar](1024) NOT NULL,
    [DateUpload_LectionMaterial] [date] NULL,
    [TimeUpload_LectionMaterial] [time](7) NULL,
    [Name_FileLecture] [varchar](64) NOT NULL,
    [URL_FileLecture] [varchar](1024) NOT NULL,
    [User_ID] [int] NULL,
    [Subject_ID] [int] NULL,
PRIMARY KEY CLUSTERED
(
    [ID_LectionMaterial] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF)
ON [PRIMARY]
) ON [PRIMARY]
GO
/***** Object: Table [dbo].[PracticalWork]  Script Date: 20.03.2025 19:12:47 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[PracticalWork](
    [ID_PracticalWork] [int] IDENTITY(1,1) NOT NULL,
    [Name_PracticalWork] [varchar](64) NOT NULL,
    [Description_PracticalWork] [varchar](3000) NOT NULL,
    [DateUpload_PracticalWork] [date] NULL,
    [TimeUpload_PracticalWork] [time](7) NULL,
    [Name_FilePracticalWork] [varchar](64) NOT NULL,
    [URL_FilePracticalWork] [varchar](3000) NOT NULL,
    [User_ID] [int] NULL,
    [Subject_ID] [int] NULL,
PRIMARY KEY CLUSTERED
(
    [ID_PracticalWork] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF)
ON [PRIMARY]

```

```

) ON [PRIMARY]
GO
/******** Object: Table [dbo].[QuestionsTests]  Script Date: 20.03.2025 19:12:47 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[QuestionsTests](
    [ID_Question] [int] IDENTITY(1,1) NOT NULL,
    [Question_Text] [varchar](1000) NOT NULL,
    [QuestionType] [int] NOT NULL,
    [AnswerVariants] [varchar](max) NOT NULL,
    [CorrectAnswer] [varchar](max) NOT NULL,
    [Points] [int] NULL,
    [ID_Test] [int] NULL,
PRIMARY KEY CLUSTERED
(
    [ID_Question] ASC
) WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF)
ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
GO
/******** Object: Table [dbo].[Role]  Script Date: 20.03.2025 19:12:47 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Role](
    [ID_Role] [int] IDENTITY(1,1) NOT NULL,
    [Role_Name] [varchar](30) NOT NULL,
PRIMARY KEY CLUSTERED
(
    [ID_Role] ASC
) WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF)
ON [PRIMARY]
) ON [PRIMARY]
GO
/******** Object: Table [dbo].[Status]  Script Date: 20.03.2025 19:12:47 *****/
SET ANSI_NULLS ON

```

```

GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Status](
    [ID_Status] [int] IDENTITY(1,1) NOT NULL,
    [Name_Status] [varchar](64) NOT NULL,
PRIMARY KEY CLUSTERED
(
    [ID_Status] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF)
ON [PRIMARY]
) ON [PRIMARY]
GO
/******** Object: Table [dbo].[Subject]  Script Date: 20.03.2025 19:12:47 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Subject](
    [ID_Subject] [int] IDENTITY(1,1) NOT NULL,
    [Name_Subject] [varchar](64) NOT NULL,
    [ID_Group] [int] NOT NULL,
    [User_ID] [int] NOT NULL,
PRIMARY KEY CLUSTERED
(
    [ID_Subject] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF)
ON [PRIMARY]
) ON [PRIMARY]
GO
/******** Object: Table [dbo].[TestAttempt]  Script Date: 20.03.2025 19:12:47 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[TestAttempt](
    [ID_TestAttempt] [int] IDENTITY(1,1) NOT NULL,
    [Test_ID] [int] NOT NULL,
    [User_ID] [int] NOT NULL,
    [Start_Time] [datetime] NOT NULL,

```

```

[End_Time] [datetime] NULL,
[Status] [varchar](20) NOT NULL,
[Score] [int] NOT NULL,
[Grade] [decimal](3, 1) NOT NULL,
PRIMARY KEY CLUSTERED
(
    [ID_TestAttempt] ASC
) WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF)
ON [PRIMARY]
) ON [PRIMARY]
GO
/******** Object: Table [dbo].[Tests]  Script Date: 20.03.2025 19:12:47 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Tests](
    [ID_Test] [int] IDENTITY(1,1) NOT NULL,
    [Name_Test] [varchar](255) NOT NULL,
    [Description_Test] [varchar](1000) NULL,
    [DateUpload_Test] [date] NULL,
    [TimeUpload_Test] [time](7) NULL,
    [TimeLimit] [int] NULL,
    [Subject_ID] [int] NULL,
    [User_ID] [int] NULL,
PRIMARY KEY CLUSTERED
(
    [ID_Test] ASC
) WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF)
ON [PRIMARY]
) ON [PRIMARY]
GO
/******** Object: Table [dbo].[User]  Script Date: 20.03.2025 19:12:47 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[User](
    [ID_User] [int] IDENTITY(1,1) NOT NULL,
    [FirstName] [varchar](64) NOT NULL,

```

```

[SecondName] [varchar](64) NOT NULL,
[MiddleName] [varchar](64) NULL,
[Email] [varchar](255) NOT NULL,
[ID_Role] [int] NOT NULL,
[ID_Group] [int] NOT NULL,
[Login] [varchar](150) NOT NULL,
[Password] [varchar](150) NOT NULL,
PRIMARY KEY CLUSTERED
(
    [ID_User] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF)
ON [PRIMARY],
UNIQUE NONCLUSTERED
(
    [Email] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF)
ON [PRIMARY]
) ON [PRIMARY]
GO

GO

ALTER TABLE [dbo].[CompletedWork] ADD DEFAULT (CONVERT([time],getdate())) FOR
[TimeUpload_CompletedWork]
GO
ALTER TABLE [dbo].[LectionMaterial] ADD DEFAULT (CONVERT([date],getdate())) FOR
[DateUpload_LectionMaterial]
GO
ALTER TABLE [dbo].[LectionMaterial] ADD DEFAULT (CONVERT([time],getdate())) FOR
[TimeUpload_LectionMaterial]
GO
ALTER TABLE [dbo].[PracticalWork] ADD DEFAULT (CONVERT([date],getdate())) FOR
[DateUpload_PracticalWork]
GO
ALTER TABLE [dbo].[PracticalWork] ADD DEFAULT (CONVERT([time],getdate())) FOR
[TimeUpload_PracticalWork]
GO
ALTER TABLE [dbo].[QuestionsTests] ADD DEFAULT ((1)) FOR [Points]
GO
ALTER TABLE [dbo].[TestAttempt] ADD DEFAULT (getdate()) FOR [Start_Time]

```

```

GO
ALTER TABLE [dbo].[TestAttempt] ADD DEFAULT ((0)) FOR [Score]
GO
ALTER TABLE [dbo].[TestAttempt] ADD DEFAULT ((0)) FOR [Grade]
GO
ALTER TABLE [dbo].[Tests] ADD DEFAULT (CONVERT([date],getdate())) FOR [DateUpload_Test]
GO
ALTER TABLE [dbo].[Tests] ADD DEFAULT (CONVERT([time],getdate())) FOR [TimeUpload_Test]
GO
ALTER TABLE [dbo].[User] ADD DEFAULT ('Otcytctbyer') FOR [MiddleName]
GO
ALTER TABLE [dbo].[CompletedWork] WITH CHECK ADD FOREIGN KEY([PracticalWork_ID])
REFERENCES [dbo].[PracticalWork] ([ID_PracticalWork])
GO
ALTER TABLE [dbo].[CompletedWork] WITH CHECK ADD FOREIGN KEY([Status_ID])
REFERENCES [dbo].[Status] ([ID_Status])
GO
ALTER TABLE [dbo].[CompletedWork] WITH CHECK ADD FOREIGN KEY([User_ID])
REFERENCES [dbo].[User] ([ID_User])
GO
ALTER TABLE [dbo].[GradingCriteria] WITH CHECK ADD CONSTRAINT [FK_GradingCriteria_Tests]
FOREIGN KEY([ID_Test])
REFERENCES [dbo].[Tests] ([ID_Test])
ON DELETE CASCADE
GO
ALTER TABLE [dbo].[GradingCriteria] CHECK CONSTRAINT [FK_GradingCriteria_Tests]
GO
ALTER TABLE [dbo].[LectureMaterial] WITH CHECK ADD FOREIGN KEY([Subject_ID])
REFERENCES [dbo].[Subject] ([ID_Subject])
GO
ALTER TABLE [dbo].[LectureMaterial] WITH CHECK ADD FOREIGN KEY([User_ID])
REFERENCES [dbo].[User] ([ID_User])
GO
ALTER TABLE [dbo].[PracticalWork] WITH CHECK ADD FOREIGN KEY([Subject_ID])
REFERENCES [dbo].[Subject] ([ID_Subject])
GO
ALTER TABLE [dbo].[PracticalWork] WITH CHECK ADD FOREIGN KEY([User_ID])
REFERENCES [dbo].[User] ([ID_User])
GO
ALTER TABLE [dbo].[QuestionsTests] WITH CHECK ADD FOREIGN KEY([ID_Test])
REFERENCES [dbo].[Tests] ([ID_Test])
GO

```

```

ALTER TABLE [dbo].[Subject] WITH CHECK ADD FOREIGN KEY([ID_Group])
REFERENCES [dbo].[Group] ([ID_Group])
GO
ALTER TABLE [dbo].[Subject] WITH CHECK ADD FOREIGN KEY([User_ID])
REFERENCES [dbo].[User] ([ID_User])
GO
ALTER TABLE [dbo].[TestAttempt] WITH CHECK ADD FOREIGN KEY([Test_ID])
REFERENCES [dbo].[Tests] ([ID_Test])
GO
ALTER TABLE [dbo].[TestAttempt] WITH CHECK ADD FOREIGN KEY([User_ID])
REFERENCES [dbo].[User] ([ID_User])
GO
ALTER TABLE [dbo].[Tests] WITH CHECK ADD FOREIGN KEY([Subject_ID])
REFERENCES [dbo].[Subject] ([ID_Subject])
GO
ALTER TABLE [dbo].[Tests] WITH CHECK ADD FOREIGN KEY([User_ID])
REFERENCES [dbo].[User] ([ID_User])
GO
ALTER TABLE [dbo].[User] WITH CHECK ADD FOREIGN KEY([ID_Group])
REFERENCES [dbo].[Group] ([ID_Group])
GO
ALTER TABLE [dbo].[User] WITH CHECK ADD FOREIGN KEY([ID_Role])
REFERENCES [dbo].[Role] ([ID_Role])
GO
ALTER TABLE [dbo].[GradingCriteria] WITH CHECK ADD CONSTRAINT [CK_Grade] CHECK
(([Grade]=(5) OR [Grade]=(4) OR [Grade]=(3)))
GO
ALTER TABLE [dbo].[GradingCriteria] CHECK CONSTRAINT [CK_Grade]
GO
ALTER TABLE [dbo].[GradingCriteria] WITH CHECK ADD CONSTRAINT [CK_Points] CHECK
(([Min_Points]<=[Max_Points]))
GO
ALTER TABLE [dbo].[GradingCriteria] CHECK CONSTRAINT [CK_Points]
GO
ALTER TABLE [dbo].[TestAttempt] WITH CHECK ADD CHECK (([Status]='completed' OR
[Status]='in_progress'))
GO
ALTER TABLE [dbo].[User] WITH CHECK ADD CHECK (([Email] like '%@%.%'))
GO

```