

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

<i>Факультет</i> вычислительной техники	<i>Кафедра</i> МОиПЭВМ
Направление подготовки	09.03.02 «Информационные системы и технологии»
Направленность	Информационные системы и технологии

БАКАЛАВРСКАЯ РАБОТА
на тему
Приложение для автоматизации
образовательных процессов в онлайн школах
при подготовке к ЕГЭ

Студент	_____	_____Аникин Никита Андреевич_____
	(подпись, дата)	(ФИО полностью)
Руководитель	_____	_____Такташкин Д.В._____
	(подпись, дата)	(фамилия, инициалы)
Нормоконтролёр	_____	_____Такташкин Д.В._____
	(подпись, дата)	(фамилия, инициалы)
Работа допущена к защите (протокол заседания кафедры от _____ № _____)		
Заведующий кафедрой	_____	_____Козлов А.Ю._____
	(подпись)	(фамилия, инициалы)
Работа защищена с отметкой _____ (протокол заседания ГЭК от _____ № _____)		
Секретарь ГЭК	_____	_____Попова Н.А._____
	(подпись)	(фамилия, инициалы)

Пенза, 2024

Федеральное государственное бюджетное образовательное учреждение
высшего образования
ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет
Вычислительной техники

Кафедра
МО и ПЭВМ

«Утверждаю»
Заведующий кафедрой
А.Ю. Козлов
«25» декабря 2023 г

**ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
БАКАЛАВРА**

1. Студент Аникин Никита Андреевич гр. 20ВИ1 факультета ФВТ
<Фамилия, имя, отчество полностью>
2. Тема работы Приложение для автоматизации образовательных процессов в онлайн школах
при подготовке к ЕГЭ

Тема утверждена приказом ПГУ № _____ от " _____ " _____ 20 _____

2. Руководитель работы Такташкин Д.В.

3. Задание на работу (назначение разработки, исходные данные и т.п.)

Назначение: управление онлайн школой для подготовки к ЕГЭ

Функции:

1. Управление учебной группой

2. Управление пользователями системы

3. Контроль заработной платы

4. Создание занятий и ДЗ

5. Проверка ДЗ

6. Посещение занятий

7. Выполнение ДЗ

8. Оценка качества занятия

Технологии разработки: клиент-сервер, ООП, дизайн-система

Языки программирования: PHP, JavaScript

Среды разработки: JetBrains PhpStorm, JetBrains DataGrip, JetBrains WebStorm

4. Перечень подлежащих разработке вопросов

1. Анализ предметной области

2. Анализ требований на разработку

3. Проектирование системы

4. Реализация системы

5. Тестирование системы

5. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1	Анализ предметной области и требований к системе	10%	08.04.2024 – 13.04.2024	
2	Проектирование программных средств	20%	15.04.2024 – 25.04.2024	
3	Реализация программных средств	30%	26.04.2024 – 12.05.2024	
4	Тестирование программных средств	20%	13.05.2024 – 22.05.2024	
5	Оформление пояснительной записки	20%	23.05.2024 – 03.06.2024	

Дата выдачи задания «21» декабря 2023г.

Руководитель бакалаврской работы _____
(подпись, дата)

Задание к исполнению принял студент _____
(подпись, дата)

Работу к защите допустить
Декан ФВТ _____ Л.Р. Фионова
(подпись, дата)

Реферат

Пояснительная записка содержит 200 листов, 60 рисунков, 12 использованных источников, 29 таблиц, 1 листинг и 3 приложения

WEB-СЕРВИС, РЕПЕТИТОРСТВО, PHP 7.4., PHPSTORM, MYSQL, GIT, VUE.JS, BOOTSTRAP, NGINX, AXIOS, VUE ROUTER, ARCHITECTURE, КЛИЕНТ-СЕРВЕР.

Объектом исследования являются web-сервис, с помощью которого происходит взаимодействие модераторов, репетиторов и учениками.

Предмет исследования – технология разработки web-сервиса для автоматизации образовательных процессов в онлайн-школах.

Цель работы: реализовать web-сервис онлайн школы для подготовки к ЕГЭ.

Технологии разработки: PHP 7.4., MySQLt, Git, Vue.js, Bootstrap, Ngnix, Axios.

Результат работы: спроектирован и разработан web-сервис автоматизации образовательных процессов в онлайн-школах для подготовки к ЕГЭ.

					ПГУ 09.03.02 – 8ВИ201.01 ПЗ			
Изм.	Лист	№ докум.	Подпись	Дата	«Приложение для автоматизации образовательных процессов в онлайн школах при подготовке к ЕГЭ» Пояснительная записка	Лит.	Лист	Листов
Разраб.		Аникин Н.А.						
Провер.		Такташкин Д.В.					4	200
Реценз.						Группа 20ВИ1		
Н. Контр.		Такташкин Д.В						
Утверд.		Козлов А.Ю.						

Оглавление

Введение.....	7
1 Анализ предметной области и постановка задачи на разработку	8
1.1 Анализ предметной области.....	8
1.2 Анализ существующих решений	9
1.2.1 Сервис Soholms.....	9
1.2.2 Сервис Cloudtext.....	13
1.2.3 Сервис You teach me	16
1.2.4 Сервис Finch.....	18
1.3 Сравнительный анализ сервисов конкурентов	20
1.4 Постановка цели и задач.....	28
1.5 Постановка функциональных требований.....	28
1.6 Нефункциональные требования к клиентскому обеспечению.....	32
1.7 Нефункциональные требования к серверной части	34
1.8 Организация разработки и определение финансовых затрат.	35
2 Проектирование сервиса.....	39
2.1 Выбор и обоснование средств проектирования	39
2.2 Проектирование базы данных.....	40
2.3 Проектирование архитектуры.....	42
2.4 Проектирование интерфейса.....	46
3 Реализация сервиса.....	49
3.1 Обоснование средств разработки	49
3.2 Реализация архитектуры приложения.....	49
3.3 Реализация базы данных.....	50
3.4 Реализация интерфейса.....	52
3.4.1 Интерфейс для модератора школы.....	53
3.4.2 Интерфейс для репетитора школы	60
3.4.3 Интерфейс для ученика школы.....	69
4 Тестирование сервиса	75
4.1 Функциональное тестирование.....	75

4.2	Анализ метрик качества кода.....	76
	Заключение	81
	Список использованных источников	82
	Приложение А (справочное) Глоссарий	84
	Приложение Б (рекомендуемое) Исходной код клиентской части.....	86
	Приложение В (рекомендуемое) Исходной код серверной части	169
	Приложение Г (рекомендуемое) Тестирование приложения	195

Введение

В последние десятилетия цифровизация обучения стала одним из ключевых направлений развития в сфере образования. С появлением доступного интернета возможности обучения расширились, а стандарты учебных программ пересматриваются под влиянием цифровых технологий. Особенно актуальной становится эта тенденция в контексте подготовки к ЕГЭ (единому государственному экзамену) – важнейшему этапу в жизни школьников в современной России. Онлайн-школы, специализирующиеся на подготовке к ЕГЭ, играют ключевую роль в обеспечении качественной подготовки учеников. Автоматизация образовательных процессов в таких учебных заведениях становится неотъемлемым элементом, позволяющим эффективно адаптировать учебные программы под индивидуальные потребности каждого ученика и обеспечивать максимальную результативность обучения [1].

Однако создание своей учебной платформы это очень долгий и ресурсозатратный процесс, поэтому многие репетиторы и небольшие онлайн-школы прибегают к готовым продуктам, что есть на рынке. Зачастую такие решения не обеспечивают нужным инструментарием для раскрытия потенциала онлайн образования. В связи с этим возникает потребность создания веб-сервиса, который удовлетворяет потребности преподавателей и учеников, содействуя их развитию.

Целью данной выпускной квалификационной работы является разработка веб-приложения по автоматизации образовательных процессов в онлайн-школах. Сервис ответит на современные вызовы в образовании, и продемонстрирует практическое применение автоматизации для повышения качества образовательного процесса в онлайн-школах, подготавливающих к ЕГЭ.

1 Анализ предметной области и постановка задачи на разработку

1.1 Анализ предметной области

При анализе предметной области удалось выделить две основные группы пользователей, что используют сервисы для организации обучения, при подготовке к ЕГЭ.

Первые это репетиторы, что ведут множество групповых и индивидуальных занятий. Им нужна платформа, где они могли бы структурировать и организовывать информацию по различным параметрам: расписание занятий, выплаты учеников, отправка и проверка домашнего задания.

Вторые это небольшие онлайн-школы, где помимо организации работы с учениками, необходима организация работы репетиторов: распределение учеников, контроль зарплат, обработка обратной связи, отслеживание качества работы преподавателей.

Помимо разных групп пользователей, также есть разные подходы к обучению, они делятся на три типа.

Вебинарный. Несколько раз в месяц проводится общая трансляция, для большого количества учеников (от тридцати и более), после чего всем раздаются материалы и задания для закрепления знаний.

Групповой. Несколько раз в неделю, в маленькой группе людей (до пятнадцати человек) проводится онлайн занятие в сервисах для видео звонков (например Zoom), после занятий выдается домашнее задание, которое обычно необходимо успеть сделать до начала следующего урока.

Индивидуальный. Репетитор проводит занятия с учеником один на один, во всем остальном схоже с групповым обучением.

Исходя из всего вышеописанного, нужно реализовать сервис, который покроет потребности как и репетиторов, так и онлайн-школ, а также позволит реализовать все три подхода к обучению. Поэтому основной функционал продукт следующий:

- создание и курирование учебных групп / учеников;

- создание и проверка домашнего задания;
- создание и проведение занятий;
- управление пользователями;
- возможность отслеживать денежные средства;
- возможность отслеживать прогресс учеников.

1.2 Анализ существующих решений

Анализ продуктов конкурента помогает выявить основные потребности целевой аудитории, сравнить сильные и слабые стороны реализации того или иного аспекта сервиса, расставить приоритет разработки функционала продукта. В настоящее время на рынке лидерами автоматизации работы онлайн школ являются:

- Soholms;
- Finch;
- Cloudtext;
- You Teach me.

1.2.1 Сервис Soholms

SOHOLMS – это система управления обучением (Learning Management System), предназначенная для использования в малом и среднем бизнесе [2]. Она предоставляет инструменты для создания, управления и отслеживания обучающих курсов и материалов, а также для оценки прогресса обучения сотрудников или студентов.

Для создания занятия разработан гибкий конструктор, который позволяет создавать лонгрид – статьи, формата длинного текстового материала, с различным наполнением: начиная от обычного текста, заканчивая вставкой текстов, слайдеров, видео и прочего (рисунок 1).

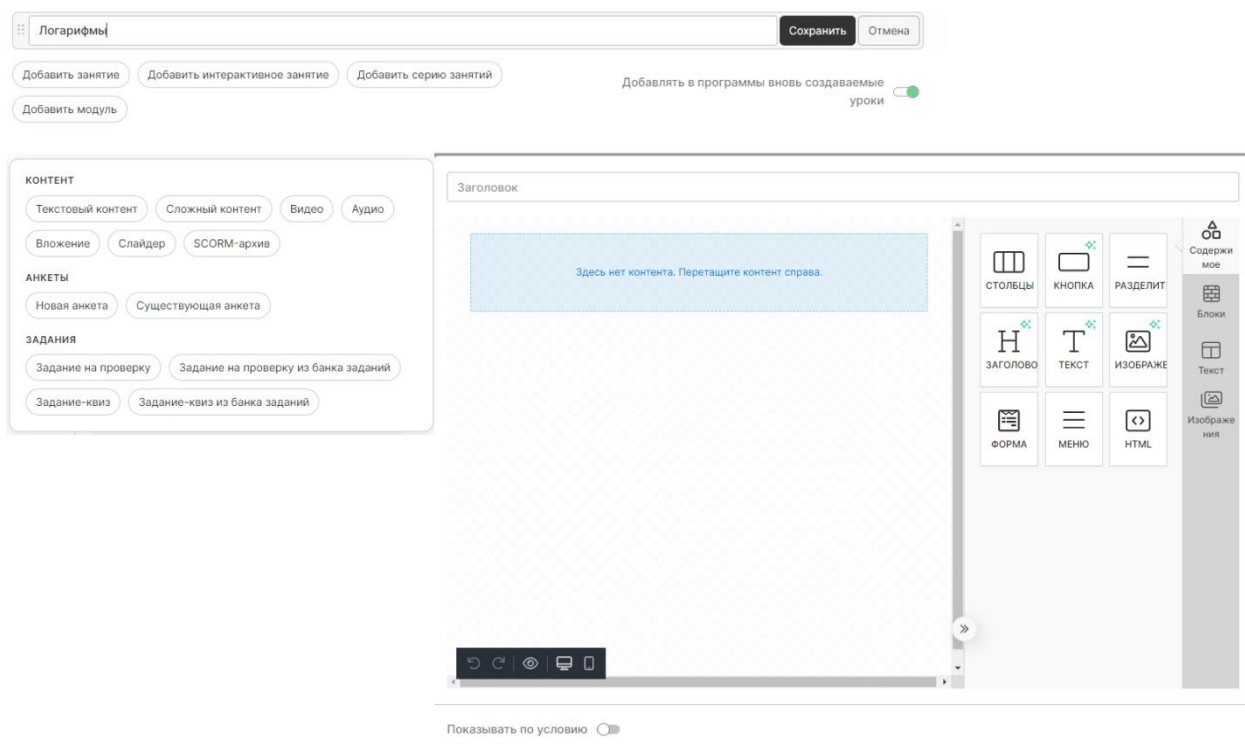


Рисунок 1 – Создание занятия

Платформа предоставляет разнообразные инструменты для оформления домашних заданий. Можно создавать ДЗ в формате длинной статьи с заданиями, с обширным выбором наполнения, а также использовать квизы – небольшие презентации со слайдами, содержащими тесты и информацию (рисунок 2).

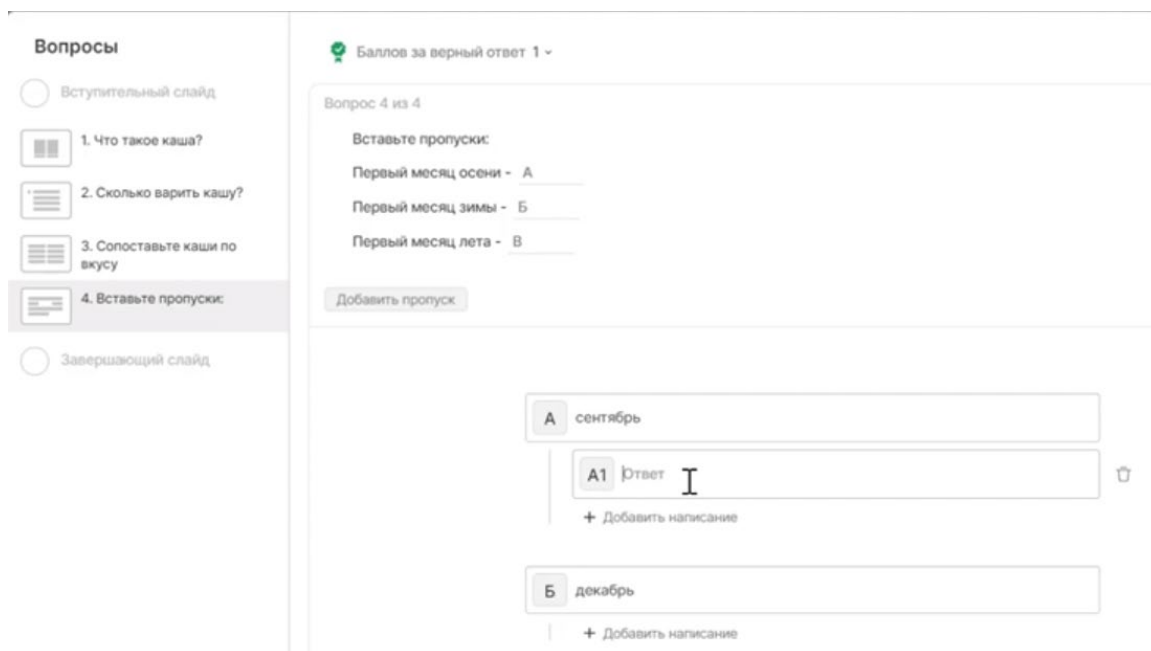


Рисунок 2 – Создание квиз теста

В сервисе все ученики распределяются по учебным группам, у каждой группы может быть свой преподаватель, проведение занятий можно контролировать с помощью расписания (рисунки 3, 4).

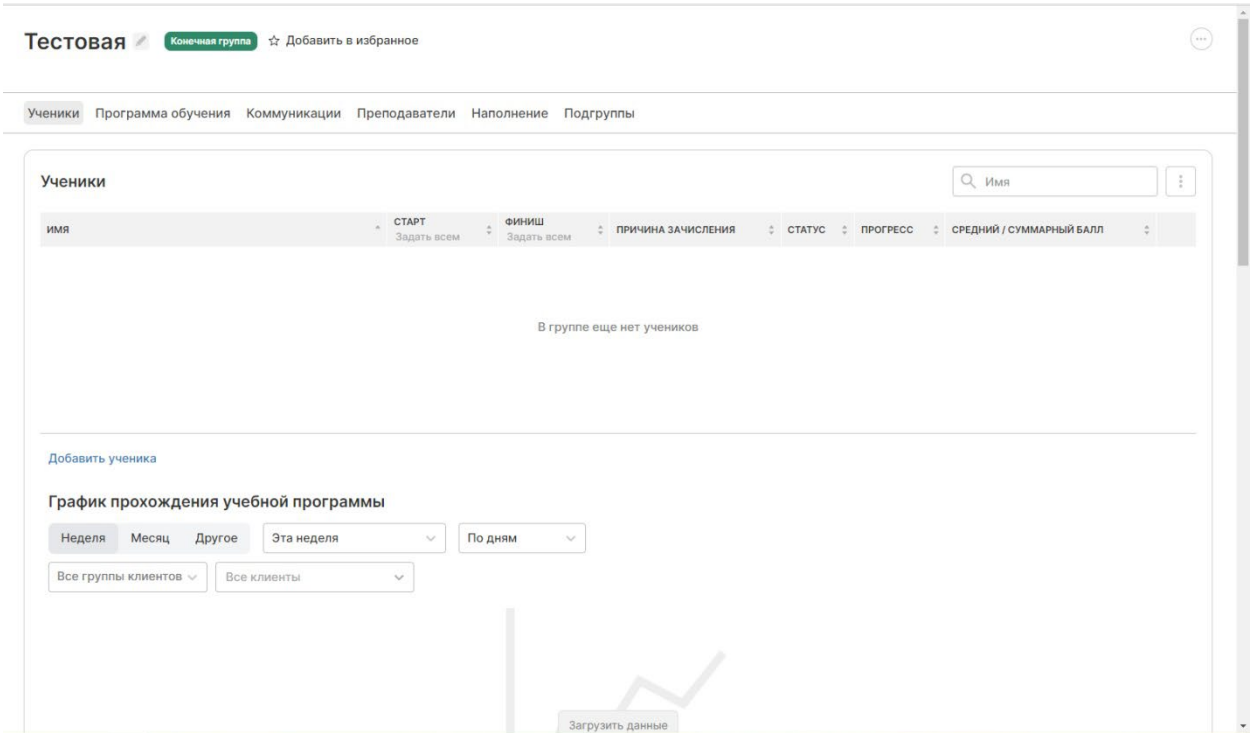


Рисунок 3 – Создание группы

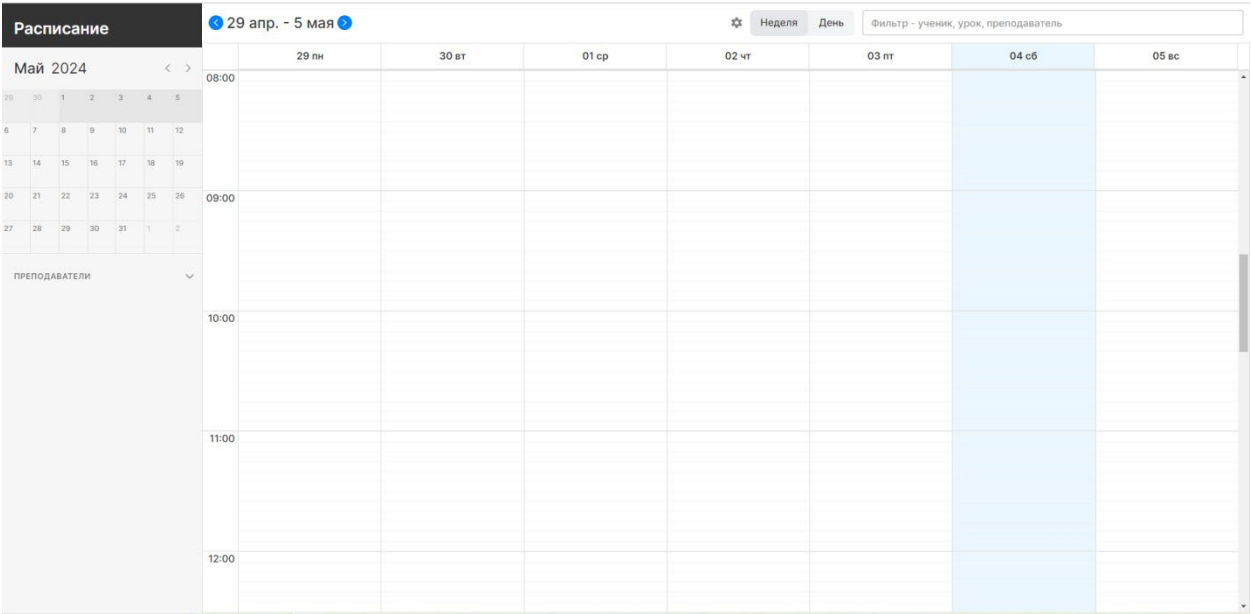


Рисунок 4 – Расписание занятий

Общаться с учениками можно посредством встроенного мессенджера (рисунок 5).

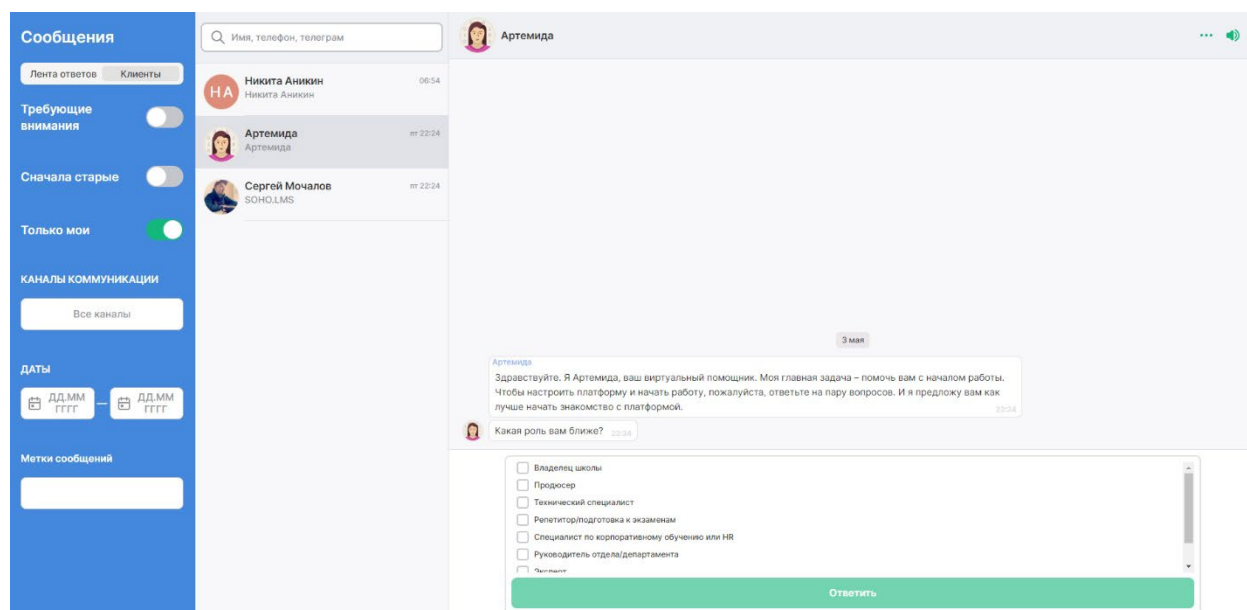


Рисунок 5 – Мессенджер, для общения с учениками

На платформе есть CRM система, которая отслеживает оплаты учеников за проведенные занятия, также с помощью систему можно контролировать зарплаты репетиторов (рисунки 6, 7).

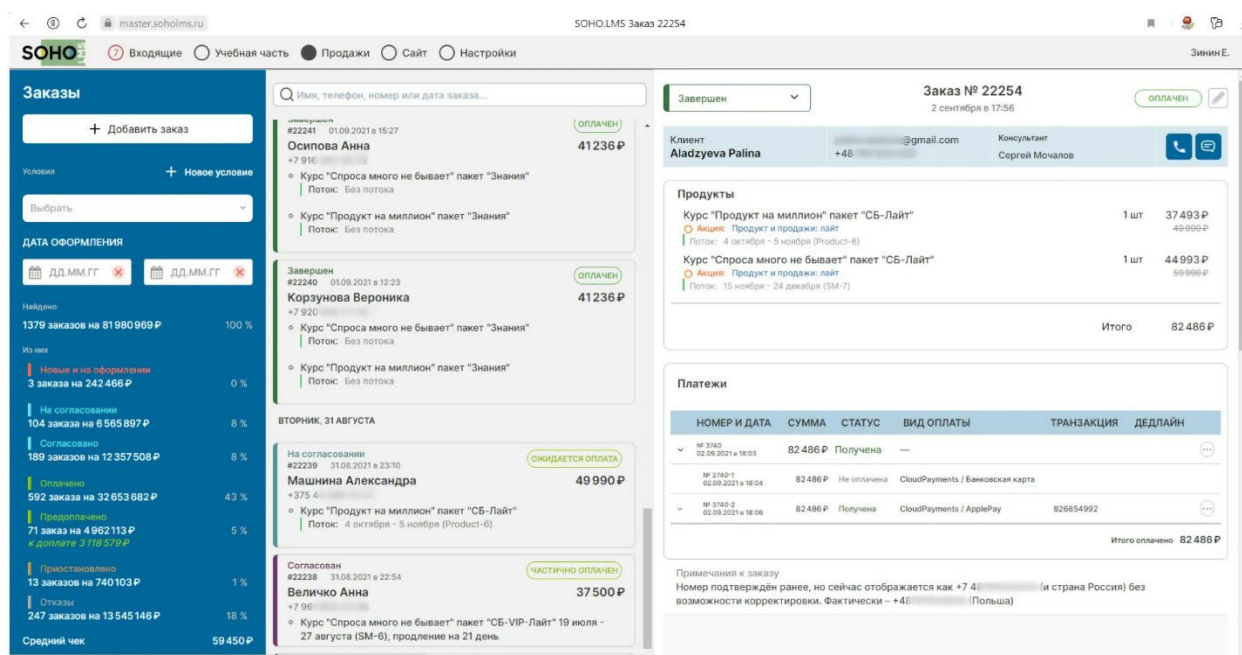


Рисунок 6 – Контроль оплат учеников

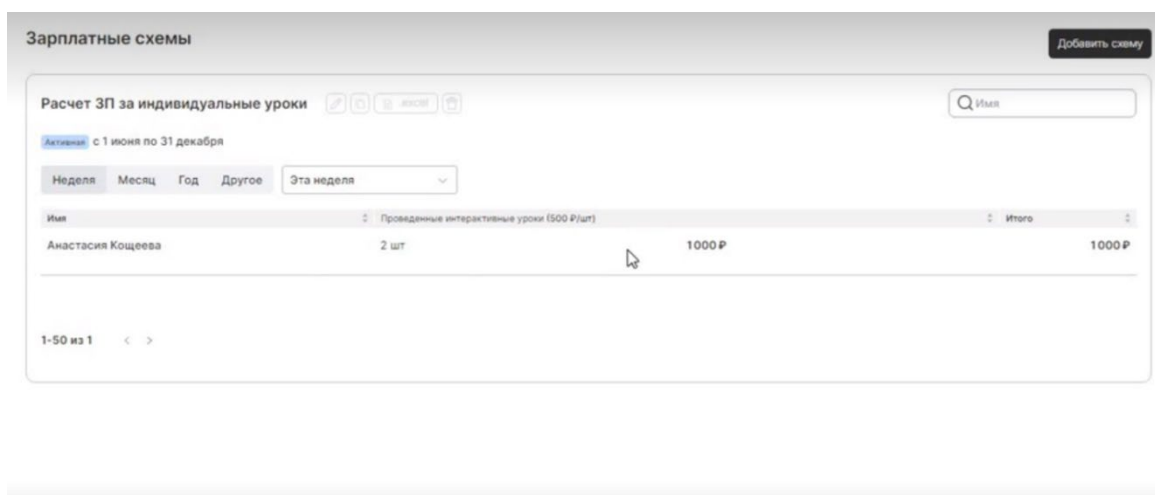


Рисунок 7 – Контроль зарплат

В заключении, можно сделать вывод, что сервис SOHOLMS успешно покрывает почти весь необходимый функционал для пользователя, однако реализация проверки домашнего задания оставляет желать лучшего, это просто выставление оценки за работу, без возможности указать ученику на его ошибки.

1.2.2 Сервис Clouddtext

Clouddtext – это онлайн платформа, предназначенная для упрощения процесса проверки, отслеживания результатов выполнения и обратной связи по заданиям ЕГЭ [3].

На платформе не проводятся занятия, не регистрируются оплаты и не создается расписание, основной упор тут сделан на проверках заданий ЕГЭ. В сервисе есть большой банк заданий первой и второй части, а также продвинутая система проверки ДЗ, где можно комментировать не только отдельные участки текста в работе ученика, но и его фотографии. Банк заданий поделен на предметы ЕГЭ и регулярно пополняется новыми задачами. Также есть инструменты для контроля соблюдения сроков выполнения заданий. Примеры проверок показаны на рисунках 8, 9.

1

Ниж **пример сочинения** ЕГЭ по русскому языку от ученика. Попробуйте выделить фрагмент и дать комментарий.

Если вы ранее работали в Google Docs, то принцип подобный. Ошибки и баллы по критериям считаются автоматическими.

Если кликнуть на название критерия, то отобразится пояснение.

Слов: 288 Минимум: 150

Как быстро проходит жизнь?  Именно об этом размышляет Борис Петрович Екимов в предложенном для анализа тексте.

Сначала рассказчик вспоминает случай, который произошёл много лет назад, когда мальчик Петя опоздал к обеду: "Тогда наш Петя был маленьким. Учился в третьем, а пожилая женщина только начала собирать вещи: "Мать наша стоит, маленькая, виноватая, голову подняла и попросила робко: Не ругайте меня, пожалуйста". Казалось бы, ещё недавно Петя был маленьким и ходил в школу, но он уже повзрослел. Герои осознали, как быстро прошло время и постарела их мать, вспомнили о случае из прошлого и не стали упрекать её .

Эти примеры **сопоставлены друг другу** : сначала автор говорит о событиях, произошедших много лет назад, а затем описывает схожую ситуацию, которая возникла спустя годы. Это показывает, как на самом деле быстротечна наша жизнь и насколько трудно иногда бывает это принять.

Авторская позиция ясна. Писатель убеждён: жизнь проходит достаточно быстро, даже если нам кажется, что она тянется вечно.

Я полностью согласна с автором. Действительно, мы порой не замечаем, как быстро прошло время. Например, в произведении А.С. Пушкина "Евгений Онегин" главный герой

Логическая

Проблема сформулирована неверно!

Пунктуационная

Точка после кавычек не нужна

Речевая

автор подчёркивает, что к старости необходимо относиться со снисхождением

Речевая

... дополняют друг друга противопоставлены, а не "сопоставлены" друг друга

Рисунок 8 – Проверка текста

[illegible]

Рисунок 9 – Проверка текста

Учеников можно приглашать в сервис и объединять в учебные группы (рисунок 10).

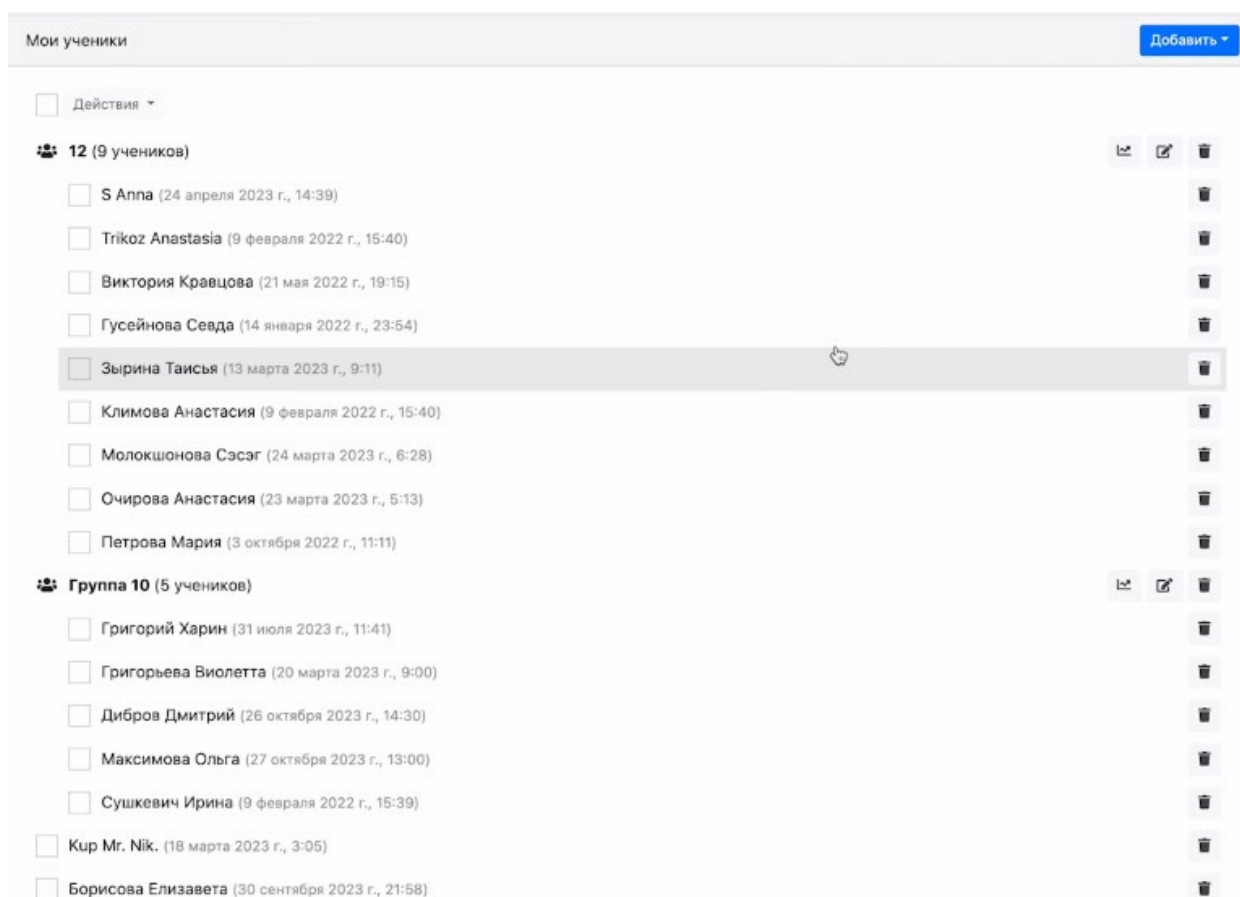


Рисунок 10 – Список учеников и групп

Вся статистика храниться в журнале выполнения заданий (рисунок 11).

Журнал Группа 10							
От	-	27 октября 2023 г.	Группа 10	Задание	Показать		
№	Имя	Кол-во выполненных ДЗ	Средний балл, %		Essay. Leisure activities in...	история	
1	Григорий Харин	0	0%				
2	Григорьева Виолетта	0	0%				
3	Дибров Дмитрий	50	81%	1 из 1 (100%)	13 из 16 (82%)		
4	Максимова Ольга	7	49%		Проверить	Проверить	
5	Сушкевич Ирина	5	84%				

Рисунок 11 – Журнал выполнения

По итогам анализа сервиса, можно сделать вывод, что основной упор при разработке продукта была направлена на создание и проверку ДЗ. Весь остальной функционал либо реализован слабо, либо его вообще нет.

1.2.3 Сервис You teach me

You teach me – это онлайн-платформа, созданная для упрощения процесса обучения и обмена знаниями [4]. С помощью этого сервиса преподаватели могут создавать и продавать свои курсы по различным темам.

В продукте занятия проводятся онлайн прямо на платформе, во время урока можно использовать интерактивную доску для взаимодействия с учеником, переписываться с ним в чате, прикладывая к уроку другой контент. Вся информация после проведения занятия сохраняется (рисунки 12, 13).

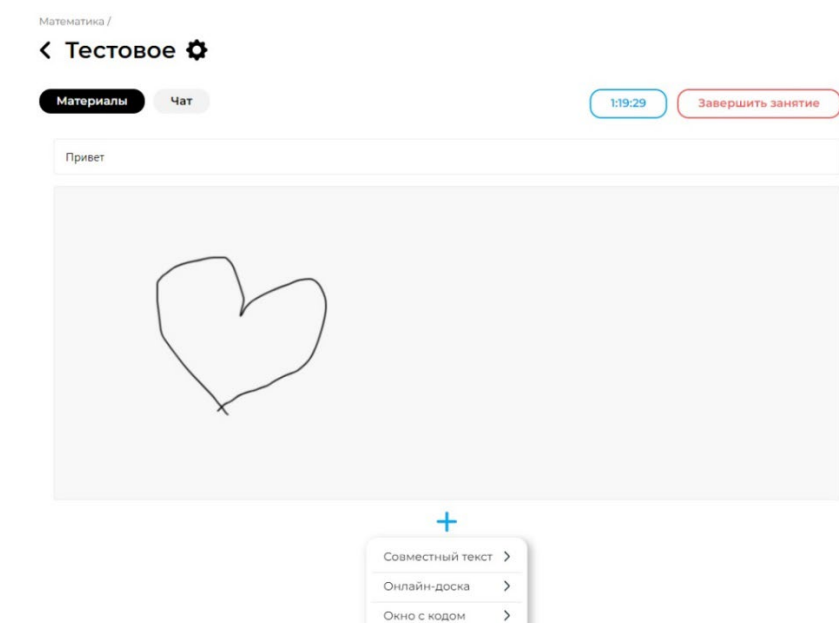


Рисунок 12 – Онлайн занятие

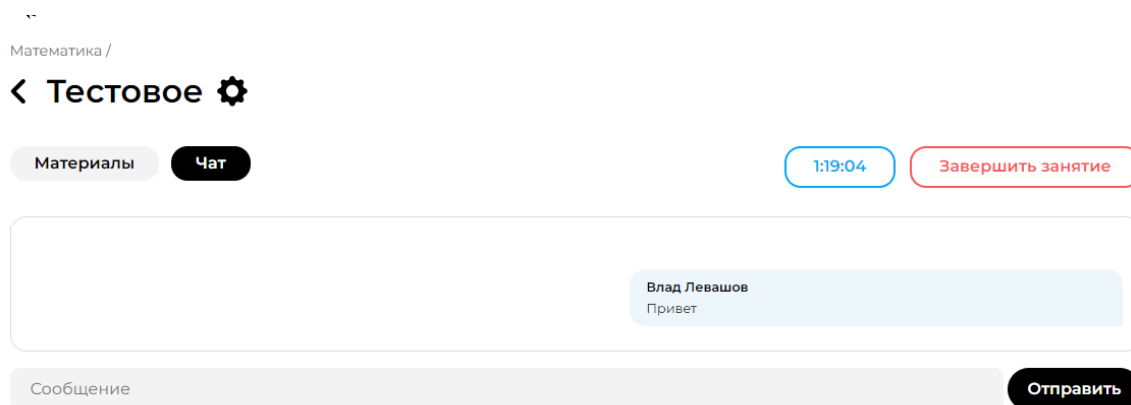


Рисунок 13 – Чат с учеником

В плане создания и проверки домашнего задания все достаточно примитивно: в домашнем задании можно лишь писать текст и отправлять задание ученику (рисунок 14).

< Логарифмы ⚙

Статус домашнего задания

Активно ▼

Здесь вы можете набирать текст!

Здесь вы можете набирать текст!

Здесь вы можете набирать текст!

+

Текстовый материал >

Из черновиков >

Рисунок 14 – Создание домашнего задания

Что облегчить менеджмент занятий, на платформе существует расписание уроков (рисунок 15).

Расписание

< С 29 апреля по 5 мая 2024 >

+ Создать занятие

	Пн 29.04	Вт 30.04	Ср 01.05	Чт 02.05	Пт 03.05	Сб 04.05	Вс 05.05
01:00							
02:00							
03:00							
04:00							
05:00							

Рисунок 15 – Расписание занятий

У продукта хорошо продумано взаимодействие ученика и учителя при проведении занятия, для всех остальных потребностей сервис предоставляет самый минимум функционала.

1.2.4 Сервис Finch

Finch – это сервис для организации образовательного процесса, а также регистрации денежных средств учеников [5]. Продукт не предназначен для проведения занятий или проверке ДЗ, сугубо контроль финансового вопроса.

Ученики добавляются на платформу вручную, их можно объединять в учебные группы, а также ввести их баланс, когда ученик оплатил, а когда почему-то не сделал этого (рисунок 16).

ГОВОРЛИВЫЙ КЕНГУРУ

Счет ученика

Баланс: **-3000 RUB**

Дата	Сумма
05.05.2024	-1500 RUB
03.05.2024	1500 RUB
03.05.2024	-1500 RUB
03.05.2024	-500 RUB
03.05.2024	-1000 RUB

[+ Пополнить](#)

Общая информация

Подготовка к музыкальному училищу, отделение вокала. Подготовка по 6-ти пунктам экзамена: специальность, коллоквиум, музыкальная грамота, сольфеджио, музыкальная литература, фортепиано. Провести тестовый экзамен, ученик очень волнуется и забывает при волнении изученное.

Имя: Говорливый Кенгуру

Основной контакт: kenguru@gmail.com

Статус: Зарегистрирован ✓

[+ Добавить занятие](#)

Занятия

Дата	Время	Статус	Стоимость	Оплата
05.05.2024	10:45-11:45	Проведено	1500 RUB	Долг
02.05.2024	10:45-11:45	Проведено	1500 RUB	Долг
01.05.2024	10:45-11:45	Проведено	500 RUB	Оплачено
30.04.2024	10:45-11:45	Проведено	1000 RUB	Оплачено

Рисунок 16 – Страница ученика

Помимо этого, на дашборде (рисунок 17) также представлена информация о текущем состоянии финансов компании, а также о долгах оплаты занятий учеников школы. Есть возможность перейти на конкретного ученика в школе, чтобы посмотреть какие занятия он не оплатил. Кроме того, доступ к детальной финансовой отчетности обеспечивает возможность проведения более глубокого анализа данных и принятия обоснованных решений.

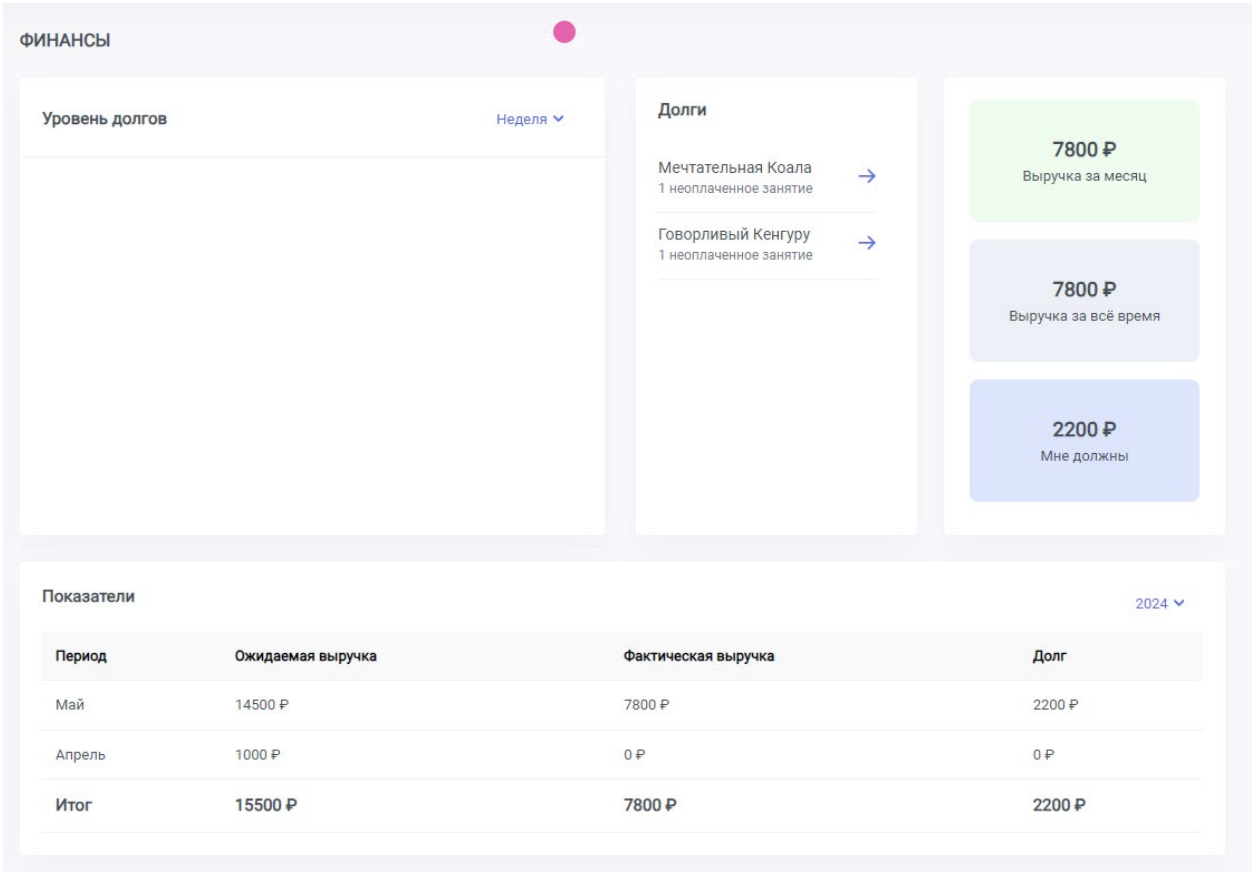


Рисунок 17 – Дашборд финансов школы

Также с учениками можно переписываться во внутреннем чате сервиса (рисунок 18).

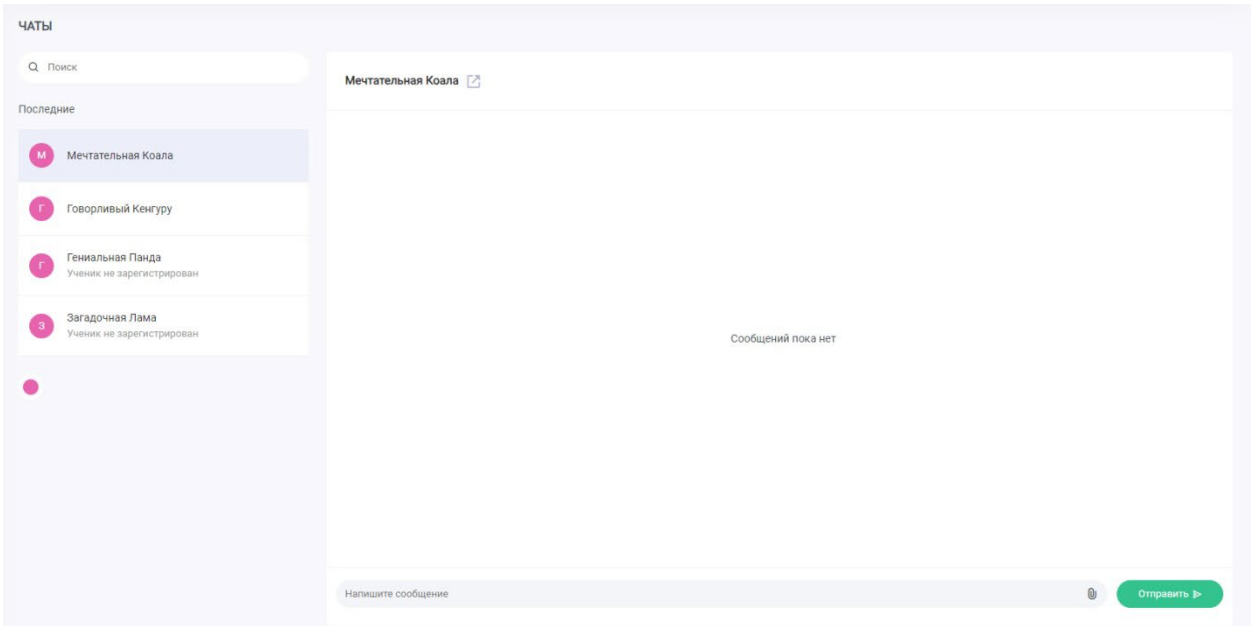


Рисунок 18 – Чат с учеником

Можно составлять расписание (рисунок 19).

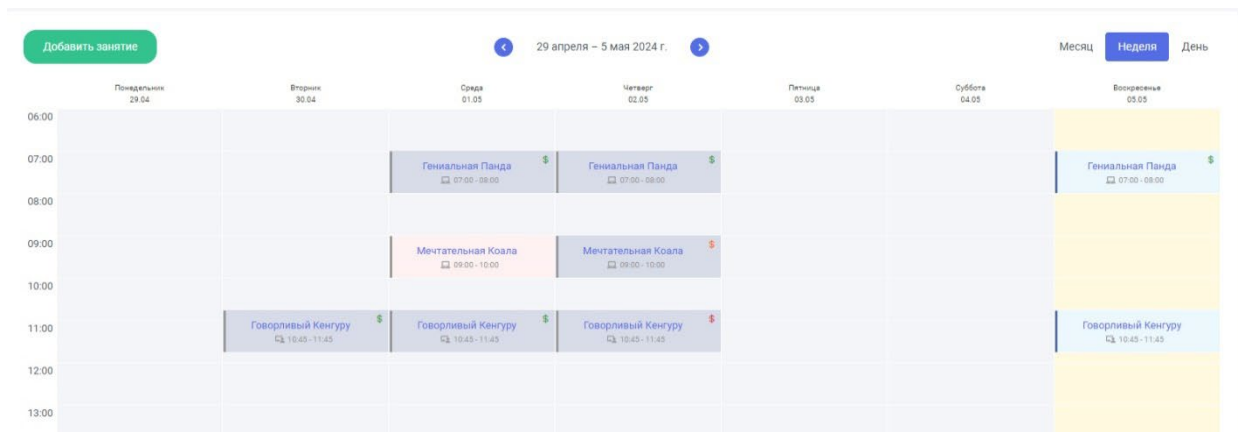


Рисунок 19 – Расписание занятий

По итогу сервис ориентирован только на управление финансами внутри онлайн-школы.

1.3 Сравнительный анализ сервисов конкурентов

Необходимо провести сравнение различных платформ с аналогичным назначением с использованием метода анализа иерархий. Метод иерархий (МАИ) – это инструмент многокритериального анализа, который помогает принимать решения в ситуациях, где есть неопределенность и множество факторов для учета. Он основан на сравнении различных вариантов по разным критериям и определении их относительной значимости [6]. Для этого сначала определяют критерии, которые важны при выборе платформы. Затем сравнивают эти платформы на основе каждого критерия и присвоим им относительные приоритеты. После этого анализируются приоритеты, чтобы сделать более информированный выбор.

Выбран арифметический метод анализа критериев вместо геометрического, так как он менее сложен в применении и не требует многочисленных математических вычислений. Кроме того, в анализе отсутствуют экстремальные значения, для обработки которых требуется геометрическое среднее. Изучив предметную область и проведя обзор сервисов, выделены следующие критерии для сравнения платформ-аналогов:

- проведение занятий;
- создание и проверка ДЗ;
- общение с учениками;
- ведение бухгалтерского учета;
- организация учебного процесса.

Необходимо создать матрицы попарных сравнений по качественной шкале, с последующим преобразованием в баллы. Обозначения баллов:

- безразлично (1 балл);
- немного лучше (3 балла);
- лучше (5 баллов);
- значительно лучше (7 баллов);
- принципиально лучше (9 баллов).

Также возможны промежуточные баллы – 2, 4, 6 и 8.

Создание матрицы a_{ij} – отношения критерия i к критерию j (таблица 1).

Таблица 1 – Матрица попарных сравнений критериев

	Проведение занятий	Создание и проверка ДЗ	Общение с учениками	Ведение бухгалтерского учета	Организация учебного процесса
Проведение занятий	1	1	9	5	3
Создание и проверка ДЗ	1	1	9	5	3
Общение с учениками	1/9	1/9	1	1/3	1/5
Ведение бухгалтерского учета	1/5	1/5	5	1	1/3
Организация учебного процесса	1/3	1/3	5	3	1

По качеству реализации критерия «Проведения занятий» сервисы можно расположить по следующим местам:

- 1) You teach me – репетитор имеет возможность менять и работать онлайн со страницей урока, прямо во время занятия с учеником, имеется интерактивное поле для взаимодействия;

2) Soholms – большой выбор контента для занятия, однако работать онлайн с учеником нельзя;

3) Cloudtext – не реализован функционал для проведения занятий;

4) Finch – не реализован функционал для проведения занятий.

Исходя из успешности реализации критерия, составлена таблица сравнения альтернатив критерия «Проведение занятий» (таблица 2).

Таблица 2 – Матрица сравнения альтернатив по критерию «Проведение занятий»

Проведение занятий	Soholms	Finch	Cloudtext	You Teach me
Soholms	1	9	9	1/3
Finch	1/3	1	9	9
Cloudtext	1/9	1/9	1	1
You Teach me	1/9	1/9	1	1

По качеству реализации критерия «Создание и проверка ДЗ» сервисы можно расположить по следующим местам:

1) Cloudtext – узконаправленный функционал для решения задач ЕГЭ, с банком заданий и продвинутой проверкой ДЗ;

2) Soholms – большой выбор контента для создания ДЗ, есть вариант создать тест квиз, но нет продуманного функционала проверки второй части;

3) You Teach me – скудные возможности для создания ДЗ;

4) Finch – не реализован функционал для создания и проверки ДЗ.

Исходя из успешности реализации критерия, составлена таблица сравнения альтернатив критерия «Создание и проверка ДЗ» (таблица 3).

Таблица 3 – Матрица сравнения альтернатив по критерию «Создание и проверка ДЗ»

Создание и проверка ДЗ	Soholms	Finch	Cloudtext	You Teach me
Soholms	1	9	1/3	4

Продолжение таблицы 3

Создание и проверка ДЗ	Soholms	Finch	Cloudtext	You Teach me
Finch	1/9	1	1/9	1/9
Cloudtext	3	9	1	5
You Teach me	1/5	9	1/5	1/4

По качеству реализации критерия «Общение с учениками» сервисы можно расположить по следующим местам:

- 1) Soholms – продвинутый мессенджер, с возможностью прикреплять к диалогу внутренние компоненты системы по типу прикреплять тесты, занятия, материалы из занятий и т. д.;
- 2) You Teach me – самый простой мессенджер с возможностью отправлять текстовые сообщения;
- 3) Finch – самый простой мессенджер с возможностью отправлять текстовые сообщения;
- 4) Cloudtext – не реализован функционал для общения с учениками.

Исходя из успешности реализации критерия, составлена таблица сравнения альтернатив критерия «Общение с учениками» (таблица 4).

Таблица 4 – Матрица сравнения альтернатив по критерию «Общение с учениками»

Общение с учениками	Soholms	Finch	Cloudtext	You Teach me
Soholms	1	5	9	5
Finch	1/5	1	9	1
Cloudtext	1/9	1/9	1	1/9
You Teach me	1/5	1	9	1

По качеству реализации критерия «Ведение бухгалтерского учета» сервисы можно расположить по следующим местам:

1) Soholms – автоматическая система с большим выбором платежных систем и страницей анализом доходов, помимо этого есть возможность регулировать зарплаты другим репетиторам;

2) Finch – система, где вручную нужно вносить информацию о выплатах учеников;

3) Cloudtext – не реализован функционал бухгалтерского учета;

4) You Teach me – не реализован функционал бухгалтерского учета;

Исходя из успешности реализации критерия, составлена таблица сравнения альтернатив критерия «Ведение бухгалтерского учета» (таблица 5).

Таблица 5 – Матрица сравнения альтернатив по критерию «Ведение бухгалтерского учета»

Ведение бухгалтерского учета	Soholms	Finch	Cloudtext	You Teach me
Soholms	1	3	9	9
Finch	1/3	1	9	9
Cloudtext	1/9	1/9	1	1
You Teach me	1/9	1/9	1	1

По качеству реализации критерия «Организация учебного процесса» сервисы можно расположить по следующим местам:

1) Soholms – можно организовывать учеников в группы, но также работать с учениками по-отдельности, прикреплять к группам преподавателей, а также формировать расписание;

2) You Teach me – можно организовывать учеников в группы, формировать расписание;

3) Finch – можно организовывать учеников в группы, формировать расписание;

4) Cloudtext – можно организовывать учеников в группы.

Исходя из успешности реализации критерия, составлена таблица сравнения альтернатив критерия «Организация учебного процесса» (таблица 6).

Таблица 6 – Матрица сравнения альтернатив по критерию «Организация учебного процесса»

Организация учебного процесса	Soholms	Finch	Cloudtext	You Teach me
Soholms	1	5	7	5
Finch	1/5	1	3	1
Cloudtext	1/7	1/3	1	1/3
You Teach me	1/5	1	3	1

Необходимо пронумеровать матрицу попарных сравнений критериев. Для этого сначала вычисляется сумма значений в каждом столбце и затем значения в таблице разделяется на полученную сумму. После этого вычисляется среднее значение для каждого столбца, чтобы определить весовой коэффициент. Полученные результаты представлены в таблице 7.

Таблица 7 – Весовые доли критериев

	Вес в долях	Вес в процентах
Проведение занятий	$\approx 0,363$	36,3 %
Создание и проверка ДЗ	$\approx 0,363$	36,3 %
Общение с учениками	$\approx 0,033$	3,3 %
Ведение бухгалтерского учета	$\approx 0,087$	8,7 %
Организация учебного процесса	$\approx 0,152$	15,2 %

По проведенным расчетам, становится ясно, что наиболее весомыми критериями является «Проведение занятий» и «Создание и проверка ДЗ», наименее важными является «Организация учебного процесса» и «Ведение бухгалтерского учета», а не требующий внимание критерий это «Общение с учениками». Далее, аналогичным образом необходимо получить векторы весовых коэффициентов сервисов-аналогов с точки зрения их соответствия отдельным критериям. Векторы коэффициентов сервисов-конкурентов приведены в таблицах 8 – 12.

Таблица 8 – Весовые доли по критерию «Проведение занятий»

Проведение занятий	Вес в долях	Вес в процентах
Soholms	$\approx 0,501$	50,1 %
Finch	$\approx 0,388$	38,8 %
Cloudtext	$\approx 0,054$	5,4 %
You Teach me	$\approx 0,054$	5,4 %

Таблица 9 – Весовые доли по критерию «Создание и проверка ДЗ»

Создание и проверка ДЗ	Вес в долях	Вес в процентах
Soholms	$\approx 0,295$	29,5 %
Finch	$\approx 0,035$	3,5 %
Cloudtext	$\approx 0,54$	54 %
You Teach me	$\approx 0,129$	12,9 %

Таблица 10 – Весовые доли по критерию «Общение с учениками»

Общение с учениками	Вес в долях	Вес в процентах
Soholms	$\approx 0,618$	61,8 %
Finch	$\approx 0,187$	18,7 %
Cloudtext	$\approx 0,035$	3,5 %
You Teach me	$\approx 0,158$	15,8 %

Таблица 11 – Весовые доли по критерию «Ведение бухгалтерского учета»

Ведение бухгалтерского учета	Вес в долях	Вес в процентах
Soholms	$\approx 0,564$	56,4 %
Finch	$\approx 0,337$	33,7 %
Cloudtext	$\approx 0,049$	4,9 %
You Teach me	$\approx 0,049$	4,9 %

Таблица 12 – Весовые доли по критерию «Организация учебного процесса»

Организация учебного процесса	Вес в долях	Вес в процентах
Soholms	$\approx 0,628$	62,8 %

Продолжение таблицы 12

Организация учебного процесса	Вес в долях	Вес в процентах
Finch	$\approx 0,154$	15,4 %
Cloudtext	$\approx 0,063$	6,3 %
You Teach me	$\approx 0,154$	15,4 %

Необходимо свести полученные результаты в одну таблицу для удобства восприятия (таблица 13).

Таблица 13 – Общая таблица весовых долей

	Проведение занятий	Создание и проверка ДЗ	Общение с учениками	Ведение бухгалтерского учета	Организация учебного процесса
Soholms	0,501	0,295	0,618	0,564	0,628
Finch	0,388	0,035	0,187	0,337	0,154
Cloudtext	0,054	0,54	0,035	0,049	0,063
You Teach me	0,054	0,129	0,158	0,049	0,154

Чтобы получить лучший вариант по всем критериям надо умножить матрицу из таблицы 13 на матрицу из таблицы 7. Результат умножения приведен в таблице 14.

Таблица 14 – Веса альтернативных приложений

	Вес в долях	Вес в процентах
Soholms	$\approx 0,452$	45,2 %
Finch	$\approx 0,21$	21 %
Cloudtext	$\approx 0,229$	22,9 %
You Teach me	$\approx 0,109$	10,9 %

Согласно сравнению, лучшим выбором согласно заданным критериям является платформа «Soholms», которая имеет весовую долю в 45,2 процента. Кроме того, этот сервис наиболее популярен и обладает самой большой пользовательской базой, что делает его наиболее подходящим для выбора в качестве прототипа.

1.4 Постановка цели и задач

Целью данной выпускной квалификационной работы является создание веб-сервиса, направленного на организацию онлайн школ по подготовке к ЕГЭ, а также учебной подготовки внутри них.

Для достижения поставленной цели требуются решить следующие задачи:

- проанализировать предметную область и сервисы конкурентов;
- описать архитектуру сервиса;
- спроектировать интерфейс приложения;
- реализовать архитектуру и интерфейс;
- протестировать получившийся продукт.

Пользователь в сервисы будут разделены на 3 основные группы: учеников, репетиторов и модераторов онлайн-школ.

Модератор должен иметь возможности добавлять и удалять репетиторов, формировать учебные группы из учеников, следить за финансовой ситуацией школы, расписывать время занятий.

Репетитор может создавать и проверять домашнее задание, создавать и проводить занятия, общаться с учениками, следить за своей зарплатой.

Модератор и репетитор может быть один и тот же человек.

Ученик должен проходить домашнее задание, просматривать занятия, общаться с репетитором, оценивать его работу.

1.5 Постановка функциональных требований

С учётом цели проекта и анализа сопоставимых программных продуктов, определены следующие функции, которые должны быть реализованы в сервисе:

- регистрация и авторизация учеников и репетиторов;
- создание и распределение в учебные группы;
- контроль и просмотр заработной платы репетитора;
- создание и проведение занятий;
- создание и проверка домашнего задания;

- возможность оценки качества преподавания;
- формирование расписания;
- оплата занятий.

На рисунке 20 показана диаграмма вариантов использования.

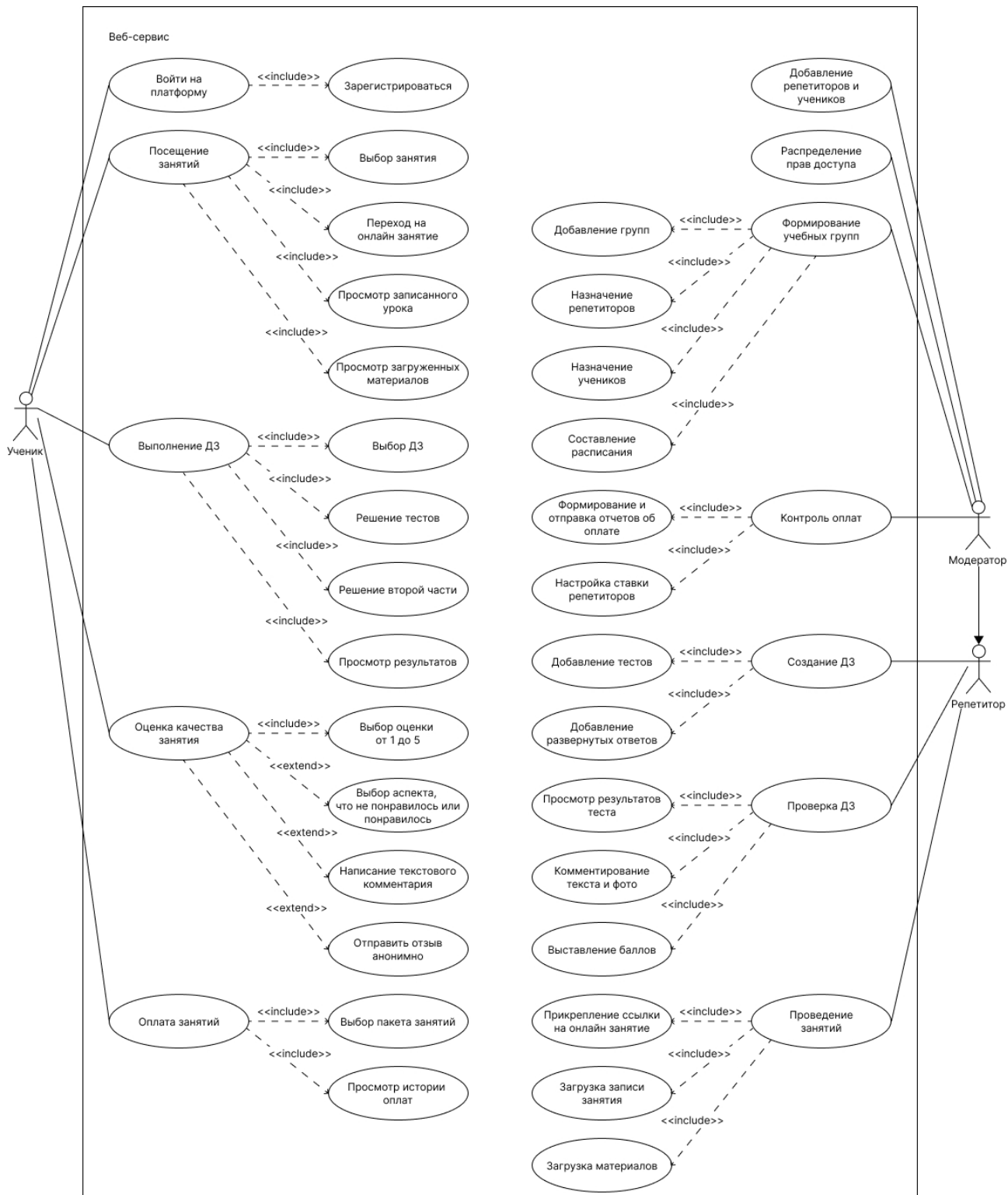


Рисунок 20 – Диаграмма вариантов использования

Процесс обработки данных и спецификация для варианта использования «Выполнение ДЗ» приведено в таблице 15.

Таблица 15 – Спецификация варианта использования «Выполнение ДЗ»

Прецедент: Выполнение ДЗ
ID: 1
Краткое описание: Ученик выполняет домашнее задание на платформе
Главные актеры: Ученик
Предусловия: <ol style="list-style-type: none"> 1. Пользователь авторизовался в сервисе 2. Репетитор опубликовал задание
Основной поток: <ol style="list-style-type: none"> 1. Пользователь заходит на страницу занятия, ДЗ после которого он должен выполнить 2. Нажимает на кнопку «Решить домашнее задание» 3. Последовательно решает тесты, пишет развернутые ответы на задания второй части 4. Ждет проверку от репетитора 5. Через время может посмотреть результаты выполненной работы

Процесс обработки данных и спецификация для варианта использования «Создание ДЗ» приведено в таблице 16.

Таблица 16 – Спецификация варианта использования «Создание ДЗ»

Прецедент: Создание ДЗ
ID: 2
Краткое описание: Репетитор создает домашнее задание и прикрепляет его к занятию
Главные актеры: Репетитор
Предусловия:

Продолжение таблицы 16

<ol style="list-style-type: none"> 1. Репетитор имеет доступ к группе 2. Занятие создано
<p>Основной поток:</p> <ol style="list-style-type: none"> 1. Репетитор открывает страницу занятия 2. Нажимает на кнопку «Создать ДЗ» 3. Попадает на страницу создания ДЗ, где может прикрепить заранее созданный тест, добавить блок с описанием или блок с развернутым ответом 4. Обозначает дедлайн сдачи ДЗ 5. Нажимает на кнопку «Добавить ДЗ»

Процесс обработки данных и спецификация для варианта использования «Проверка ДЗ» приведено в таблице 17.

Таблица 17 – Спецификация варианта использования «Проверка ДЗ»

Прецедент: Проверка ДЗ
ID: 3
Краткое описание: Репетитор проверяет домашнее задание у ученика
<p>Главные актеры:</p> <p>Репетитор</p> <p>Ученик</p>
<p>Предусловия:</p> <ol style="list-style-type: none"> 1. Ученик отправил домашнее задание на проверку
<p>Основной поток:</p> <ol style="list-style-type: none"> 1. Репетитор выбирает ученика, работу которого хочет проверить 2. Последовательно проверяет задания второй части, где может комментировать текстовый ответ ученика, а также прикрепленные картинки 3. Расставляет баллы за проверенные задания 2 части 4. Нажимает на кнопку «Оценить ответ» и оценка репетитора отправляются ученику

Процесс обработки данных и спецификация для варианта использования «Проведение занятий» приведено в таблице 18.

Таблица 18 – Спецификация варианта использования «Проведение занятий»

Прецедент: Проведение занятий
ID: 4
Краткое описание: Репетитор создает и проводит занятие в учебной группе
Главные актеры: Репетитор Ученик
Предусловия: 1. Учебная группа собрана
Основной поток: 1. Репетитор создает занятие, в котором прикрепляет ссылку на созвон в zoom / discord или других сервисов. 2. Пишет план для занятия, прикрепляет материалы, создает ДЗ 3. Публикует занятие для учеников 4. Проводит онлайн занятие, записывая его 5. После занятия загружает запись занятия 6. Прописывает таймкоды к записанному занятию 7. Сохраняет изменения.

В этом разделе были разработаны и описаны функциональные требования для сервиса.

1.6 Нефункциональные требования к клиентскому обеспечению

Верстка сервиса должна быть кроссбраузерной, сам сайт должен быть доступен для полнофункционального просмотра при помощи следующих браузеров:

- Microsoft Edge;
- Opera;

- Firefox;
- Google Chrome;
- Яндекс браузер;
- Safari.

Продукт должен быть оптимально отображаться при разрешении экрана от 375 на 812 и выше, без необходимости горизонтальной прокрутки. Элементы управления должны быть единообразно расположены на всех страницах. Графический интерфейс модулей должен соответствовать общему дизайну сайта и обеспечивать удобную навигацию и логическое перемещение между разделами. Использование однотипных процедур управления и навигационных элементов должно создавать интуитивно понятный интерфейс для пользователей. Важно также уделить внимание доступности продукта для пользователей с ограничениями или особыми потребностями, чтобы обеспечить равный доступ к функционалу и информации для всех групп пользователей.

Исходя из минимальных системных требований для работы браузеров, а также требованиям возможности работы сервиса на планшетных и мобильных устройствах, можно составить следующие требования (таблицы 19, 20).

Таблица 19 – Минимальные требования при работе на персональном компьютере

Категория требований	Минимальные системные требования
Операционная система	Windows 7 и выше или macOS 10.11 (El Capitan) и выше
Процессор	Процессора с тактовой частотой не ниже 1 ГГц
Объем оперативной памяти	не менее 2 ГБ
Место на жестком диске	не менее 100 МБ

Продолжение таблицы 19

Категория требований	Минимальные системные требования
Скорость интернет-соединения	не менее 5 – 10 Мбит/с

Таблица 20 – Минимальные требования при работе на мобильных и планшетных устройствах

Категория требований	Минимальные системные требования
Операционная система	Android 7.0 (Nougat) и выше или iOS 12 и выше
Процессор	процессор ARM или Apple A7
Объем оперативной памяти	не менее 1 ГБ
Место на жестком диске	не менее 100 МБ
Скорость интернет-соединения	не менее 5 – 10 Мбит/с

В этом разделе были установлены минимальные требования для работы с сервисом, как для персональных ПК, так и для мобильных устройств.

1.7 Нефункциональные требования к серверной части

Учитывая, что для разработки сервиса необходимо использовать следующие языки программирования (ЯП), базы данных, фреймворки и web-сервис:

- язык программирования Php 7.4;
- реляционная система управления базами данных база данных SQL;
- web-сервис Nginx;
- библиотека Bootstrap;
- фреймворк Vue.js.

Требования к серверному обеспечению приложения представлены в таблице 21.

Таблица 21 – Минимальные требования к серверу

Категория требований	Минимальные требования	Рекомендуемые требования
Операционная система	Linux, например, Ubuntu Server, CentOS или Debian	Linux, например, Ubuntu Server, CentOS или Debian
Процессор	Четырехъядерный процессор с тактовой частотой не менее 2.0 ГГц	Восьмиядерный процессор с тактовой частотой от 2.5 ГГц и выше
Объем оперативной памяти	не менее 8 ГБ	не менее 16 ГБ
Место на жестком диске	не менее 100 ГБ	не менее 250 ГБ
Скорость интернет-соединения	100 Мбит/с	Гигабитный Ethernet-адаптер

В этом разделе были установлены минимальные и рекомендуемые требования для серверной аппаратуры.

1.8 Организация разработки и определение финансовых затрат.

Перед тем как приступить к разработке программного продукта, важно детально спланировать процесс и оценить бюджетные затраты. Можно использовать методологию Agile, основанную на сотрудничестве, самоорганизации и быстрой обратной связи. Разработка ведется через короткие итерации, называемые спринтами. Каждый спринт начинается с планирования, в рамках которого команда выбирает задачи из бэклога, которые будут выполнены в течение спринта. Затем следует активная фаза выполнения задач, где команда работает согласно выбранным задачам, обмениваясь информацией и решая проблемы на ходу. Для этого составляется общий список задач, или бэклог, и разбивается на спринты с помощью трекера задач Jira (рисунок 21) [7].

☒	PEN-3 Проанализировать аналогичные сервисы	8
☒	PEN-4 Определить основные факторы сравнения сервисов	4
☒	PEN-5 Сравнить сервисы путем метода анализа иерархий	6
☒	PEN-6 Совершить постановку задачи, определив приоритетность реализации фич	8
☒	PEN-7 Сделать варфрейминг интерфейса	32
☒	PEN-8 Сделать UI дизайн	12
☒	PEN-9 Оформить UI кит	24

Рисунок 21 – Часть бэклога проекта

Каждая задача была подробно описана (рисунок 22).

Добавить Эпик / ☒ SCRUM-1

Проанализировать аналогичные сервисы

К выполнению

Действия

Описание

Просмотреть и зафиксировать основной функционал следующих сервисов:

1. Soholms (SOHO.LMS – платформа для онлайн-школ, курсов и репетиторов)
2. Finch (Увеличивайте доход с онлайн-платформой для репетиторов Finch)
3. Cloudtext (CloudText.ru — платформа для репетиторов и курсов ЕГЭ / ОГЭ)
4. You Teach me (YouTeachMe | Платформа для репетитора)

Активность

Закрепленные поля

Нажмите значок рядом с меткой поля, чтобы закрепить его.

Сведения

Исполнитель

Никита Аникин

Рисунок 22 – Пример задачи «Проанализировать аналогичные сервисы»

Все задачи из бэклога были разделены на спринты (рисунок 23).

Доска Спринт 1 Добавить даты (Задач: 5) 58 Запустить спринт

☒	PEN-3 Проанализировать аналогичные сервисы	8	К выполнению	
☒	PEN-4 Определить основные факторы сравнения сервисов	4	К выполнению	
☒	PEN-5 Сравнить сервисы путем метода анализа иерархий	6	К выполнению	
☒	PEN-6 Совершить постановку задачи, определив приоритетность реализации фич	8	К выполнению	
☒	PEN-7 Сделать варфрейминг интерфейса	32	К выполнению	

+ Создать задачу

Рисунок 23 – Пример двухнедельного спринта

После составления всех спринтов, трекер задач автоматически формирует диаграмму Ганта, чтобы наглядно отобразить рабочий процесс (рисунок 24).

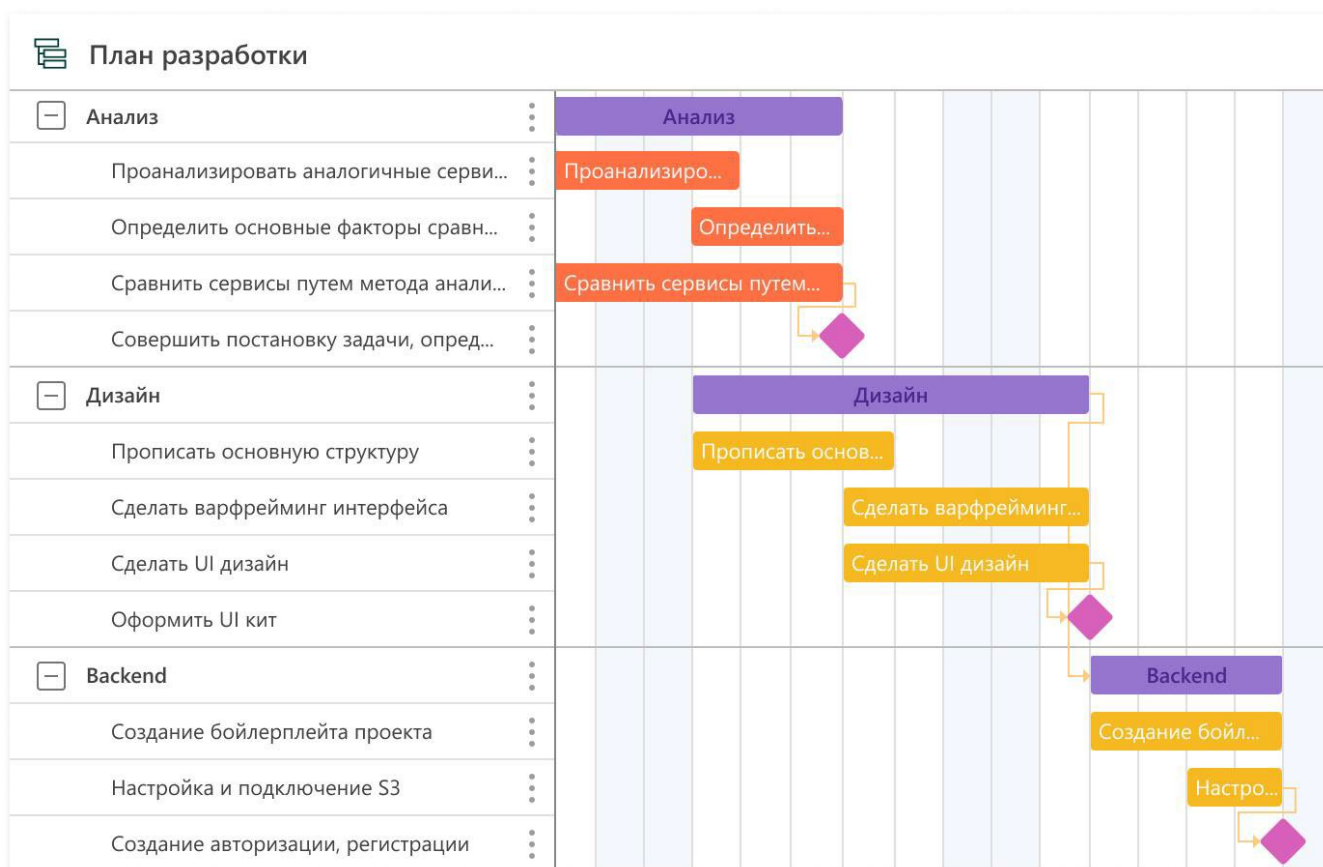


Рисунок 24 – Диаграмма Ганта

Для оценки ресурсов все задачи были распределены между тремя основными специалистами, необходимыми для выполнения проекта: UI/UX дизайнером, Frontend разработчиком и Backend разработчиком. Затем было проведено исследование с использованием популярного сайта для найма специалистов hh.ru, чтобы определить среднюю стоимость каждого специалиста, необходимого для разработки сервиса (таблица 22) [8].

Таблица 22 – Средняя зарплата специалистов

Должность	Средняя зарплата за 180 часов	Средняя зарплата за час
UI/UX дизайнер	82031.7 Р	455
Frontend разработчик	84213.3 Р	467
Backend разработчик	151 465 Р	841

Исходя из затрат времени на каждую задачу из бэклога, стоимости работы специалиста за час, была рассчитана стоимость проекта (таблица 23).

Таблица 23 – Итоговая стоимость проекта

Область работы	Стоимость работы за час	Количество часов	Итого
Дизайн и анализ	400	120	48000 Р
Backend	841	200	168000 Р
Frontend	420	180	75600 Р
Итого	2913600 Р		

Теперь, когда стоимость рассчитана и задачи определены, можно приступить к проектированию сервиса.

2 Проектирование сервиса

2.1 Выбор и обоснование средств проектирования

Для эффективного проектирования сервиса были выбраны инструменты, обладающие качествами, способствующими удобству работы и ясности представления концепций. Среди них ключевыми являются Figma и диаграммы UML.

Figma была выбрана в качестве основного инструмента для создания макетов и прототипов сервиса, из-за того, что она имеет ряд преимуществ.

1. Многофункциональность: этот инструмент предоставляет широкий спектр возможностей, включая создание интерактивных прототипов, работу с векторной графикой, а также интеграцию с другими инструментами и сервисами.

2. Систематизация: Figma предоставляет широкие возможности для создания и управления компонентами интерфейса. Создав компонент кнопки один раз, можно легко использовать его на множестве страниц. При необходимости внесения изменений достаточно отредактировать родительский компонент, что автоматически приведет к обновлению всех экземпляров кнопок на всем интерфейсе.

3. Dev Mode: в приложении есть второй режим работы, специально для разработчиков, все атрибуты компонентов удобно отображаются в виде HTML и CSS кода, что облегчает работу с дизайном при верстке.

Диаграммы UML являются неотъемлемой частью проектирования программных систем, благодаря гибкости проектирования.

1. Стандартизация: UML представляет собой универсальный язык моделирования, который широко принят в индустрии разработки программного обеспечения. Это обеспечивает единый и понятный способ визуализации системы для всей команды.

2. Разнообразие диаграмм: UML включает в себя различные типы диаграмм, такие как диаграммы классов, последовательности, состояний и

многие другие, что позволяет моделировать различные аспекты системы с учётом их взаимосвязей [9].

Выбор данных инструментов обусловлен их удобством использования, распространённостью и возможностью эффективного взаимодействия.

2.2 Проектирование базы данных

Построена логическая модель базы данных, которая абстрагирует данные от конкретной системы управления базами данных (СУБД). Эта модель основана на анализе предметной области и представлена на рисунке 25.

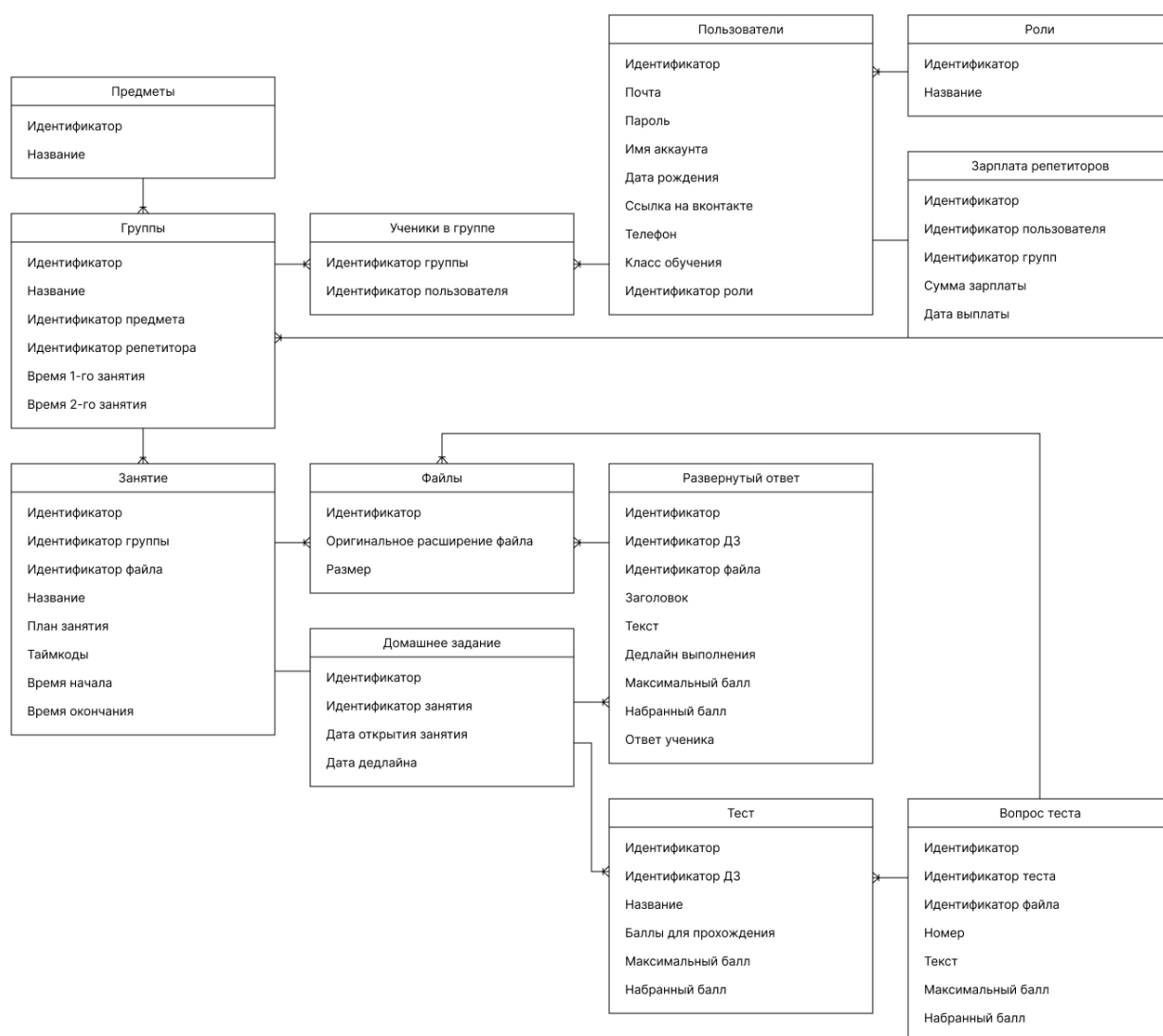


Рисунок 25 – Логическая схема БД

Спроектированная схема даст возможность хранить и осуществлять доступ к данным. Краткое описание сущностей представлено в таблице 24.

Таблица 24 – Описание сущностей

Название сущности	Описание
Пользователи	Хранится информация о пользователе, его данные для авторизации, контактная информация
Роли	Хранится информация о ролях пользователя в сервисах
Зарплата репетитора	Хранится информация о месячной зарплате репетитора в разрезе групп
Ученики в группе	Хранится информация списка учеников в определенной группе
Группы	Хранится информация о учебных группах
Предметы	Хранится информация о предметах, преподаваемых в школе
Занятия	Хранится информация о занятиях в учебной группе
Файлы	Хранится информация о файлах внутри системы
Домашнее задание	Хранится информация о ДЗ внутри занятия
Развернутый ответ	Хранится информация о вопросах второй части внутри ДЗ
Тест	Хранится информация о созданных тестах
Вопрос теста	Хранится информация о вопросах внутри теста
Статистика	Хранится информация о статистике выполнения заданий учениками

Данная структура позволит хранить информацию о всех работах и взаимодействиях пользователей.

2.3 Проектирование архитектуры

В процессе проектирования программных средств часто применяется диаграмма классов, которая является важным инструментом в области разработки программного обеспечения, который представляет собой графическое представление структуры системы в виде классов и их взаимосвязей. Каждый класс в этой диаграмме представляет абстракцию конкретной сущности или компонента системы, описывая его атрибуты и методы.

Этот инструмент обеспечивает ценную информацию о том, как компоненты системы взаимодействуют друг с другом, какие операции они выполняют и какие данные обрабатывают. Помимо этого, диаграмма классов помогает выделить основные компоненты системы, выделить интерфейсы, а также определить иерархию классов и зависимости между ними.

Основная цель диаграммы классов – обеспечить понимание структуры системы разработчикам и другим участникам проекта. Она служит важным инструментом для архитектурного проектирования, позволяя разработчикам лучше организовать код и обеспечить его масштабируемость и гибкость. Кроме того, диаграмма классов часто используется в качестве основы для последующей реализации системы, что делает ее неотъемлемой частью жизненного цикла разработки программного обеспечения. Кроме того, диаграмма классов способствует стандартизации структуры кода, упрощает коммуникацию между членами команды разработчиков и повышает его читаемость. Её использование также облегчает процесс обучения новых участников проекта, позволяя им быстрее вникнуть в архитектуру и логику работы системы [10].

Для клиентской части приложения, связанного с учебным процессом, разработана диаграмма классов, представленная на рисунке 26.

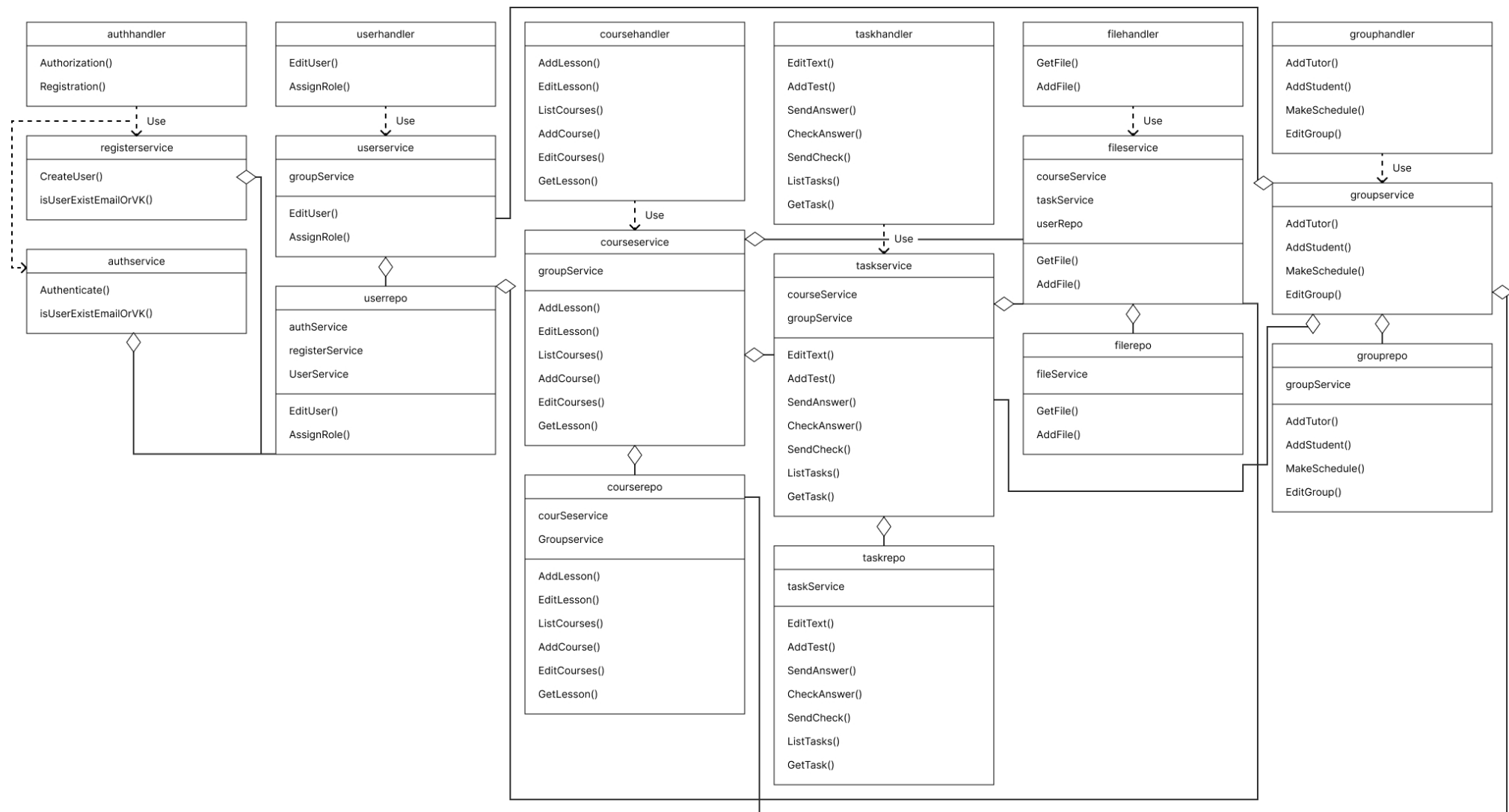


Рисунок 26 – Диаграмма классов

Краткое описание классов представлено в таблице 25.

Таблица 25 – Краткое описание классов

Название класса	Описание
authhandler	Пакет, содержащий функции обработчики для входящих запросов, связанных с авторизацией и регистрацией
userhandler	Пакет, содержащий функции обработчики для входящих запросов, связанных с изменением данных пользователей
coursehandler	Пакет, содержащий функции обработчики для входящих запросов, связанных с курсами
taskhandler	Пакет, содержащий функции обработчики для входящих запросов, связанных с заданиями
filehandler	Пакет, содержащий функции обработчики для входящих запросов, связанных с файлами
grouphandler	Пакет, содержащий функции обработчики для входящих запросов, связанных с учебными группами
registerservice	Сервис реализует бизнес-логику работы с регистрацией
authservice	Сервис реализует бизнес-логику работы с авторизацией
userservice	Сервис реализует бизнес-логику работы с пользователями

Продолжение таблицы 25

Название класса	Описание
courseservice	Сервис реализует бизнес-логику работы с курсами
taskservice	Сервис реализует бизнес-логику работы с заданиями
fileservice	Сервис реализует бизнес-логику работы с файлами
groupservice	Сервис реализует бизнес-логику работы с группами
userrepo	Репозиторий реализующий основные операции для работы с БД, связанных с пользователями
courserepo	Репозиторий реализующий основные операции для работы с БД, связанных с курсами
taskrepo	Репозиторий реализующий основные операции для работы с БД, связанных с заданиями
filerepo	Репозиторий реализующий основные операции для работы с БД, связанных с файлами
grouprepo	Репозиторий реализующий основные операции для работы с БД, связанных с группами

Инициализация всех объектов классов и разрешение зависимостей между ними выполняются вручную при запуске приложения с помощью конструкторов.

2.4 Проектирование интерфейса

Перед реализацией интерфейса в виде полноценного дизайна продукта необходимо спроектировать прототип в графическом редакторе Figma. Для начала необходимо спроектировать карту продукта – схему работы сервиса, где показано, какой функционал и на какой странице должен располагаться, как связаны страницы в приложении между собой. Часть карты приложения представлена на рисунке 27.



Рисунок 27 – Часть карты сервиса

После проектирования карты сервиса, следует приступить к созданию wireframe.

Wireframe – это образ дизайна низкой точности. Он должен четко показывать:

- основую группу контента;
- структуру информации;
- описание и базовую визуализацию взаимодействия между интерфейсом и пользователем.

Wireframe не бессмысленный набор серых блоков, хотя он выглядит примерно так. Можно рассматривать его как скелет дизайна, который должен

изображать каждую деталь финального продукта [11]. Пример прототипа одного экрана показан на рисунке 28.

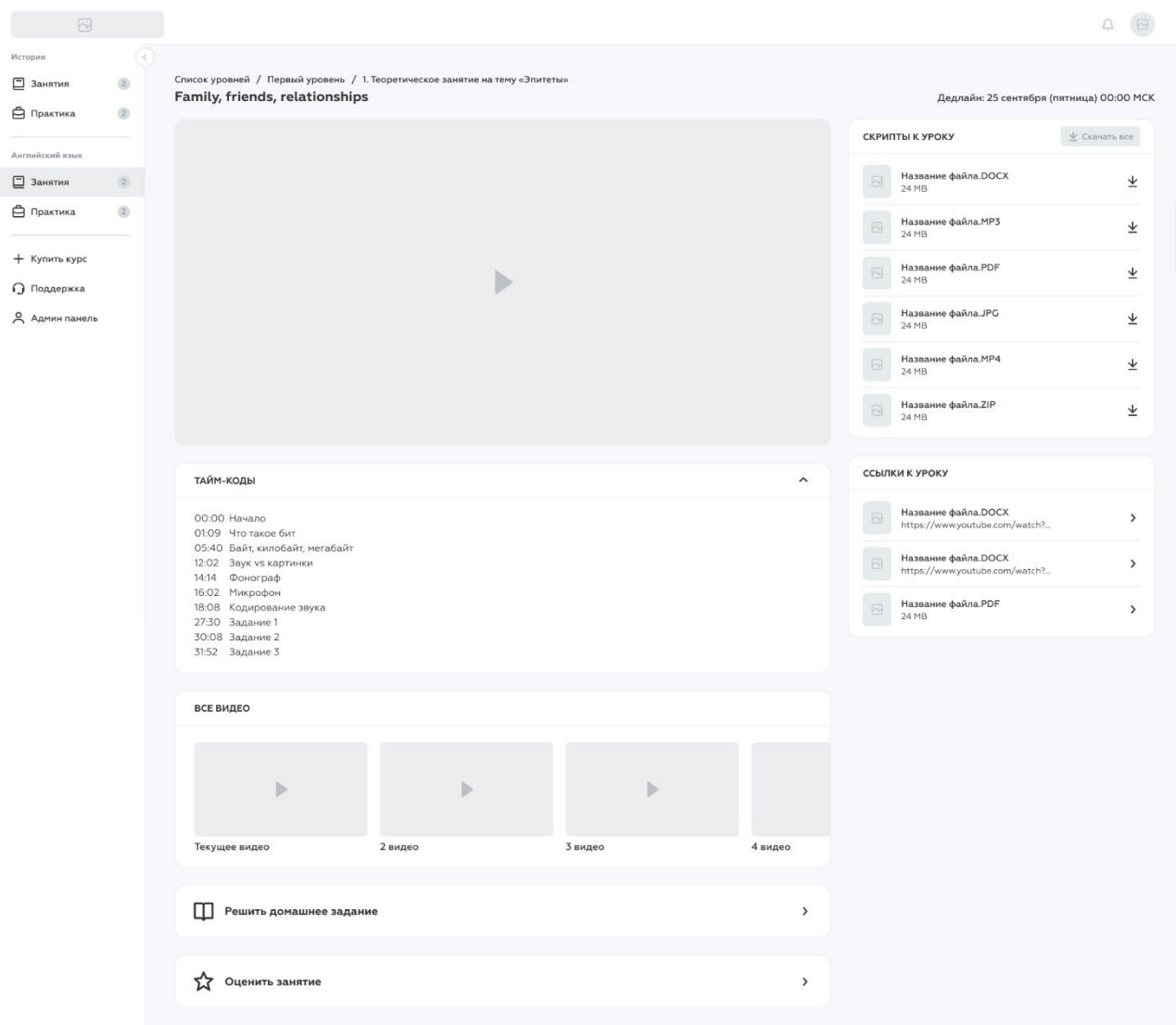


Рисунок 28 – Часть карты сервиса

Также после проектирования основных экранов создается карта взаимодействия, которая уже подробно показывает пути переходов от одного экрана к другому.

Кусок карты взаимодействия приложения показан на рисунке 29.

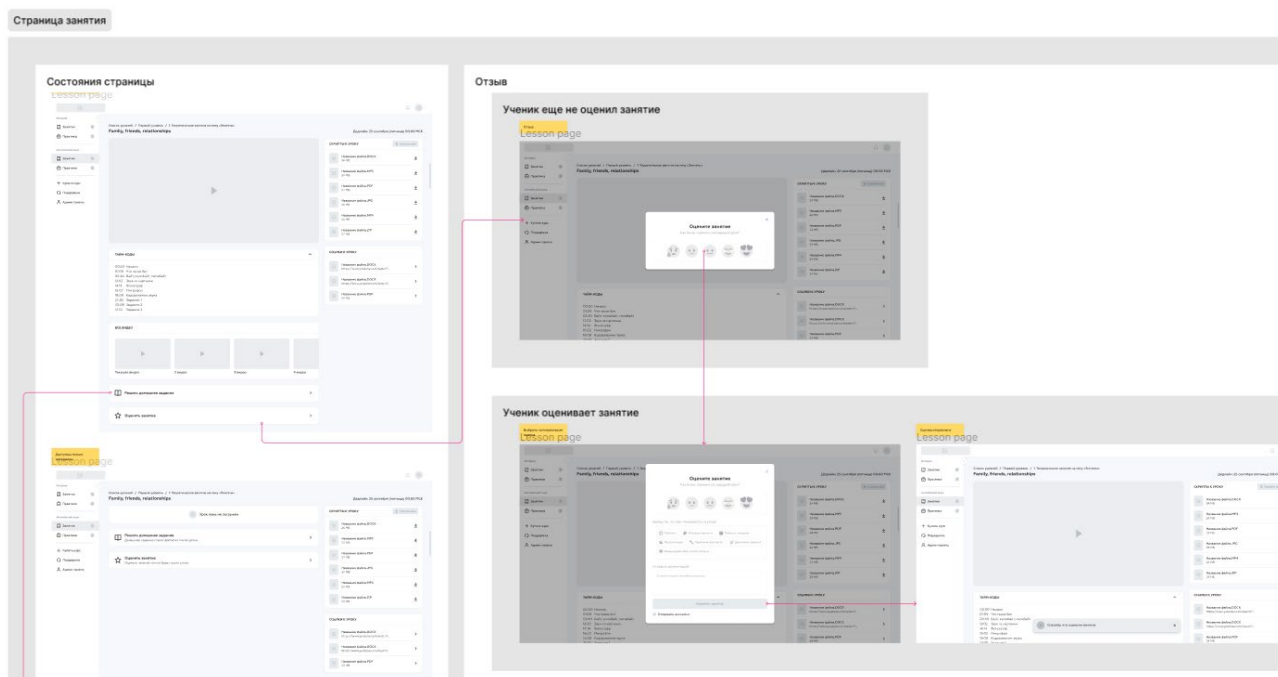


Рисунок 29 – Пример карты взаимодействия

В итоге Wireframe с картой взаимодействия созданы, далее можно переходить к созданию дизайна сервиса.

3 Реализация сервиса

3.1 Обоснование средств разработки

Для разработки бэкенд части приложения выбраны средства, обеспечивающие полную поддержку языка программирования PHP 7.4. PhpStorm предоставляет удобное окружение для написания кода на PHP, а также интеграцию с базой данных MySQL и системой контроля версий Git. Для взаимодействия с базой данных MySQL используется библиотека digitalstars/DataBase, обеспечивающая безопасность и эффективность работы с запросами.

Для фронтенд разработки выбраны средства, обеспечивающие удобство и гибкость в создании интерфейса пользовательской части приложения. Фреймворк Vue.js был выбран за его простоту использования и мощные возможности для создания реактивных пользовательских интерфейсов. Библиотеки Bootstrap и bootstrap-vue добавляют дополнительные возможности для ускорения разработки и создания стильного дизайна приложения.

Каждый выбранный инструмент для бэкенд и фронтенд разработки был отобран на основе его популярности, гибкости, документации и совместимости с другими компонентами системы, что обеспечивает эффективную и гармоничную разработку приложения.

3.2 Реализация архитектуры приложения

Диаграмма развёртывания системы показывает, как компоненты приложения будут размещены на физических устройствах и как они будут взаимодействовать между собой. В данной системе присутствуют три основных компонента: компьютер пользователя, сервер, написанный на языке программирования Php, и сервер базы данных MySQL. Пользователь взаимодействует с системой через веб-браузер на своем компьютере. Сервер принимает запросы пользователя, отправляет REST-запросы на сервер базы

данных, получает ответы и передаёт их обратно пользователю. Сервер базы данных MySQL обрабатывает запросы от сервера и возвращает результаты.

Диаграмма развёртывания приложения организации работы онлайн школ по подготовке к ЕГЭ представлена на рисунке 30.

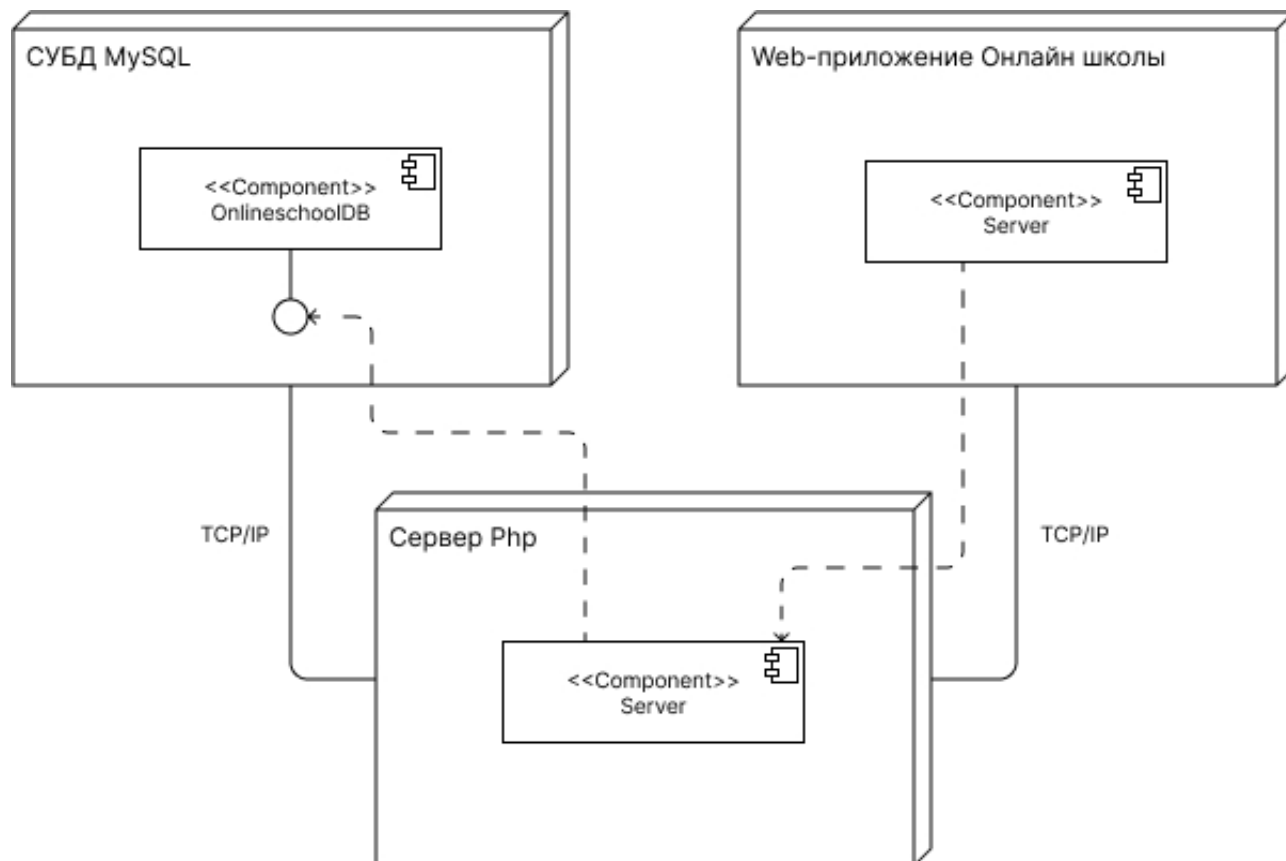


Рисунок 30 – Диаграмма развёртывания

Взаимодействие СУБД и сервера происходит через интерфейс БД OnlineschoolDB. Взаимодействие приложения в веб-браузере пользователя с сервером и с сервером БД происходит через TCP/IP соединение.

3.3 Реализация базы данных

Физическая модель базы данных «OnlineschoolDB» была создана на основе логической модели, с помощью инструментов MySQL. Эта модель представлена на рисунке 31.

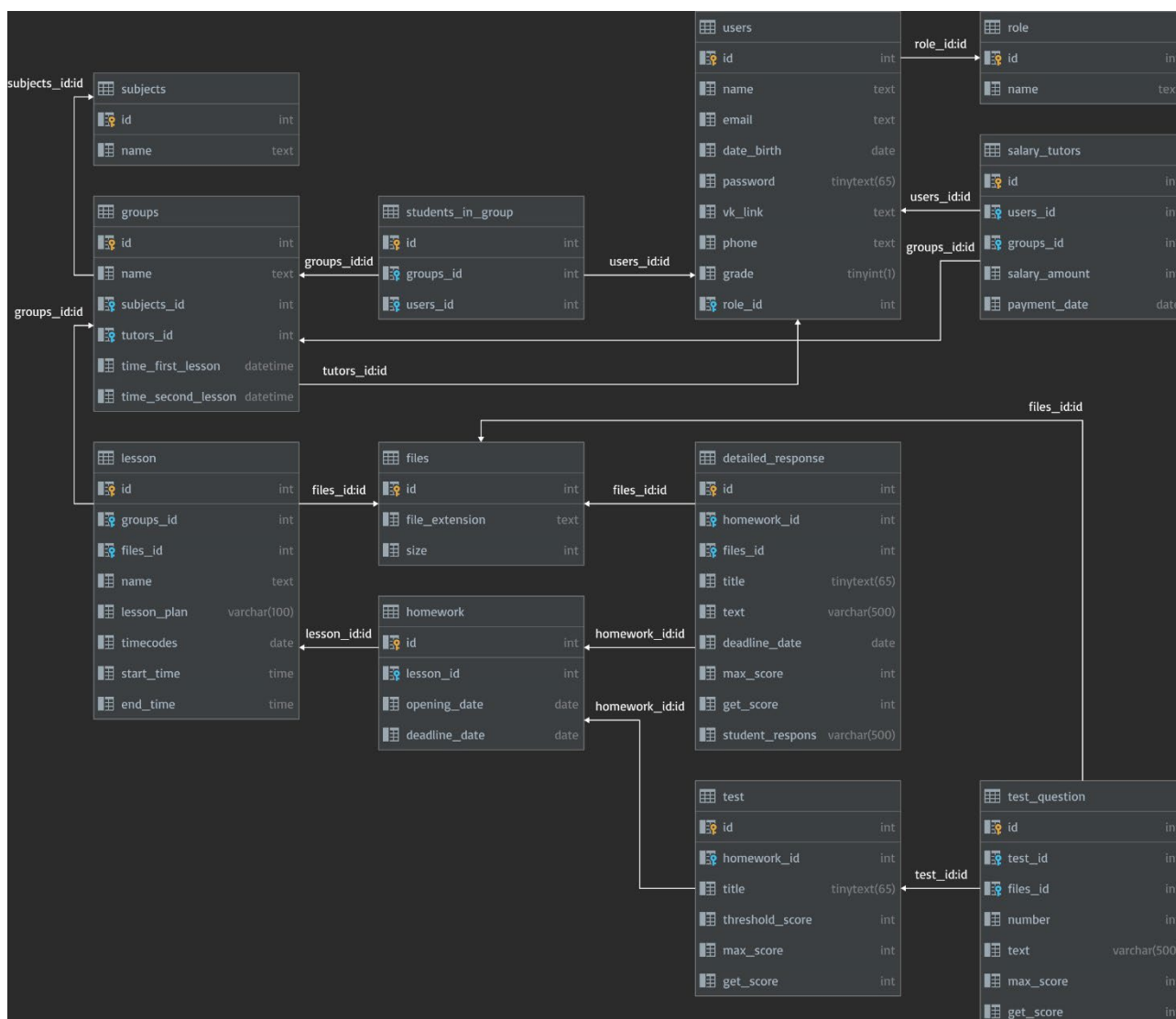


Рисунок 31 – Физическая схема БД

В физической модели базы данных для реализации связи многие-ко-многим были использованы промежуточные таблицы. Все связи между сущностями представлены в таблице 26.

Таблица 26 – Связи между сущностями

Таблица 1	Таблица 2	Связь
role	users	Один-ко-многим
salary_tutors	users	Один-к-одному
salary_tutors	groups	Один-ко-многим
users	students_in_group	Один-ко-многим
groups	students_in_group	Один-ко-многим

Продолжение таблицы 26

Таблица 1	Таблица 2	Связь
subjects	groups	Один-ко-многим
groups	lesson	Один-ко-многим
homework	lesson	Один-к-одному
homework	detailed_response	Один-ко-многим
homework	test	Один-ко-многим
test_question	test	Один-ко-многим
lesson	files	Один-ко-многим
detailed_response	files	Один-ко-многим
test_question	files	Один-ко-многим

Эта база данных была разработана таким образом, чтобы обеспечить эффективное хранение, извлечение и изменение данных. Для повышения производительности операций на основных полях были созданы индексы.

3.4 Реализация интерфейса

Интерфейс веб-приложения должен стремиться к максимальной простоте, удобству и понятности. Основная часть интерфейса будет собрана из готовых компонентов Bootstrap, что делает его знакомым для многих пользователей, учитывая широкое использование этой библиотеки на других сайтах. Вход в систему представлен общим для всех пользователей интерфейсом авторизации (рисунок 32), который предоставляет возможность входа как посредством логина и пароля, так и через аккаунт в социальной сети «ВКонтакте» с использованием её API.

Этот подход к интерфейсу обеспечивает не только удобство использования для новых пользователей, но и возможность быстрого освоения для тех, кто уже знаком с аналогичными веб-ресурсами. Кроме того, интеграция социальной авторизации через «ВКонтакте» расширяет спектр вариантов входа в систему, что делает процесс авторизации более гибким и удобным. Такой подход к дизайну интерфейса позволяет сосредоточить внимание пользователей на

основных функциях приложения, минимизируя время, необходимое для ознакомления с его возможностями. В результате, пользователи получают интуитивно понятный и эффективный инструмент для выполнения своих задач.

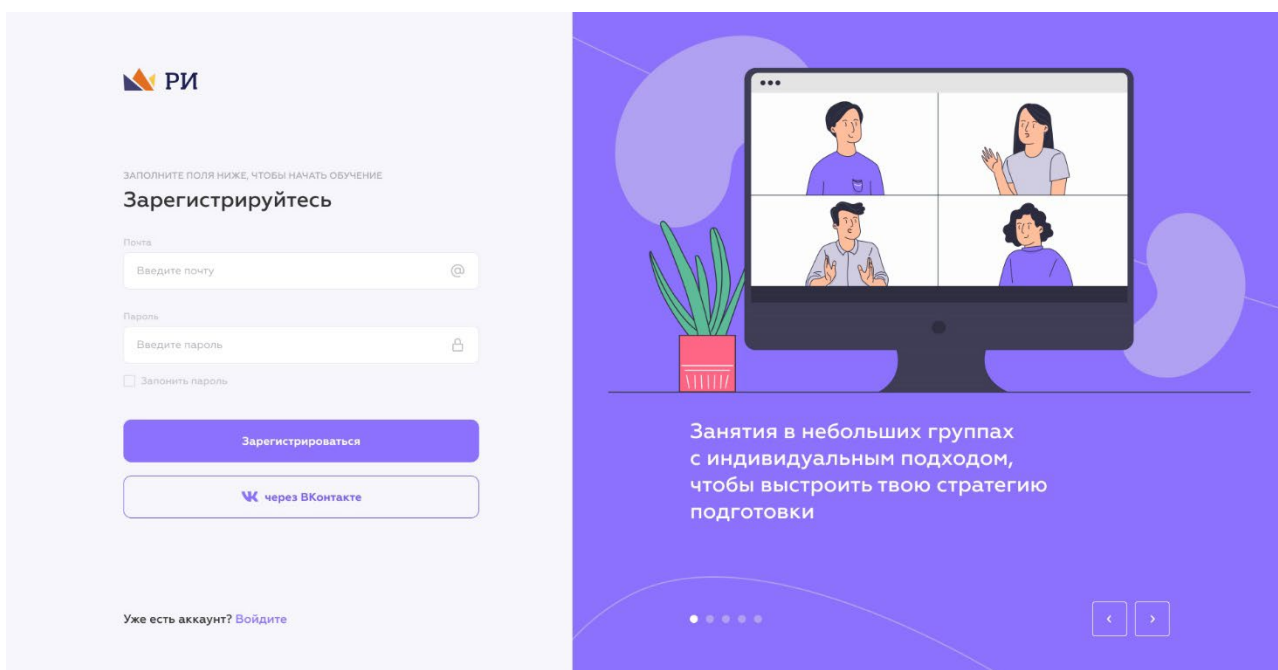


Рисунок 32 – Интерфейс авторизации в системе

Далее будут показан интерфейс, разделенный по роля пользователей.

3.4.1 Интерфейс для модератора школы

Модераторы выполняют роль, направленную на облегчение работы репетиторов, их количество может варьироваться от одного до нескольких. Основная задача модераторов заключается в эффективном распределении учеников по группам, учитывая их предпочтения по времени. Кроме того, модераторы имеют возможность создавать новые группы и удалять лишние. На странице управления группами, изображенной на рисунке 33, кураторы осуществляют свою деятельность.

Благодаря возможности создания и удаления групп, кураторы могут оперативно реагировать на изменения в потребностях обучения и динамически адаптировать структуру группировок. Такой подход обеспечивает оптимальное

использование ресурсов и повышает эффективность образовательного процесса в целом.

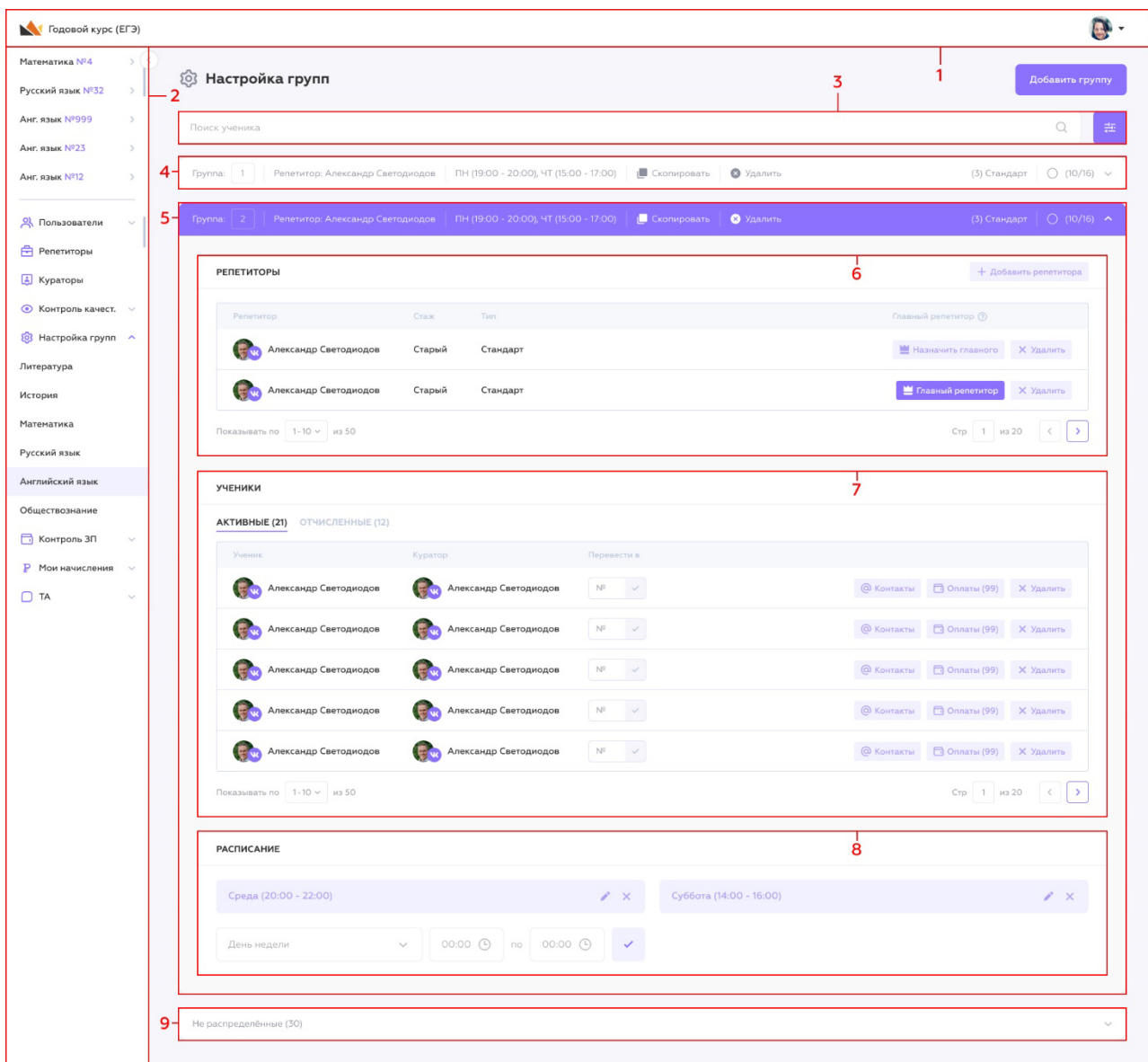


Рисунок 33 – Интерфейс настройки групп

На рисунке 33 изображено:

- 1) шапка сервиса, где расположен логотип школы и аватар аккаунта, для перехода в настройки аккаунта;
- 2) левый сайдбар, для переключения между пунктами сервиса, а также для переключения между админ панелью и пользовательской частью;
- 3) поиск по ученикам и возможность фильтрации поиска;

4) нераскрытая группа, в которой указана самая важная информация: номер группы, расписание, главный репетитор, заполнена группа или нет, при клике на блок, группа раскроется, и с ней можно будет работать;

5) раскрытая группа, в таком состоянии компонента, информацию о группе можно менять;

6) блок работы с репетиторами, где можно добавлять репетиторов, удалять и назначать главного репетитора, которому и будет поступать оплаты за проведение занятий, дополнительный репетитор необходим, в случае необходимости заменить на время главного, например при болезни;

7) список учеников, которых можно отчислить, переводить в другие группы и следить за их оплатами;

8) настройка недельного расписания группы, можно задать время занятий на любой день недели;

9) список нераспределенных учеников, которых можно назначить в разные группы.

Модераторы также имеют доступ к странице пользователей, где могут редактировать всю информацию, что есть у учеников (рисунок 34). Такая страница необходима во время активных продаж и записи учеников в группы, так как на старте обучения клиент может много раз передумать насчет обучения у того или иного репетитора. Пользователи делятся на 2 категории: «Купившие курс» и «Зарегистрированные». В категории «Купившие курс» отображаются только ученики, что оплатили хотя бы одно занятие, а в категории «Зарегистрированные» находятся ученики, что только зарегистрировались на платформе, но не оплатили обучение, а также все модераторы и репетиторы. Чтобы быстро ориентироваться и искать нужно пользователя, была разработана система поиска с подробной фильтрацией, отфильтровать можно по любому полю таблицы пользователей.

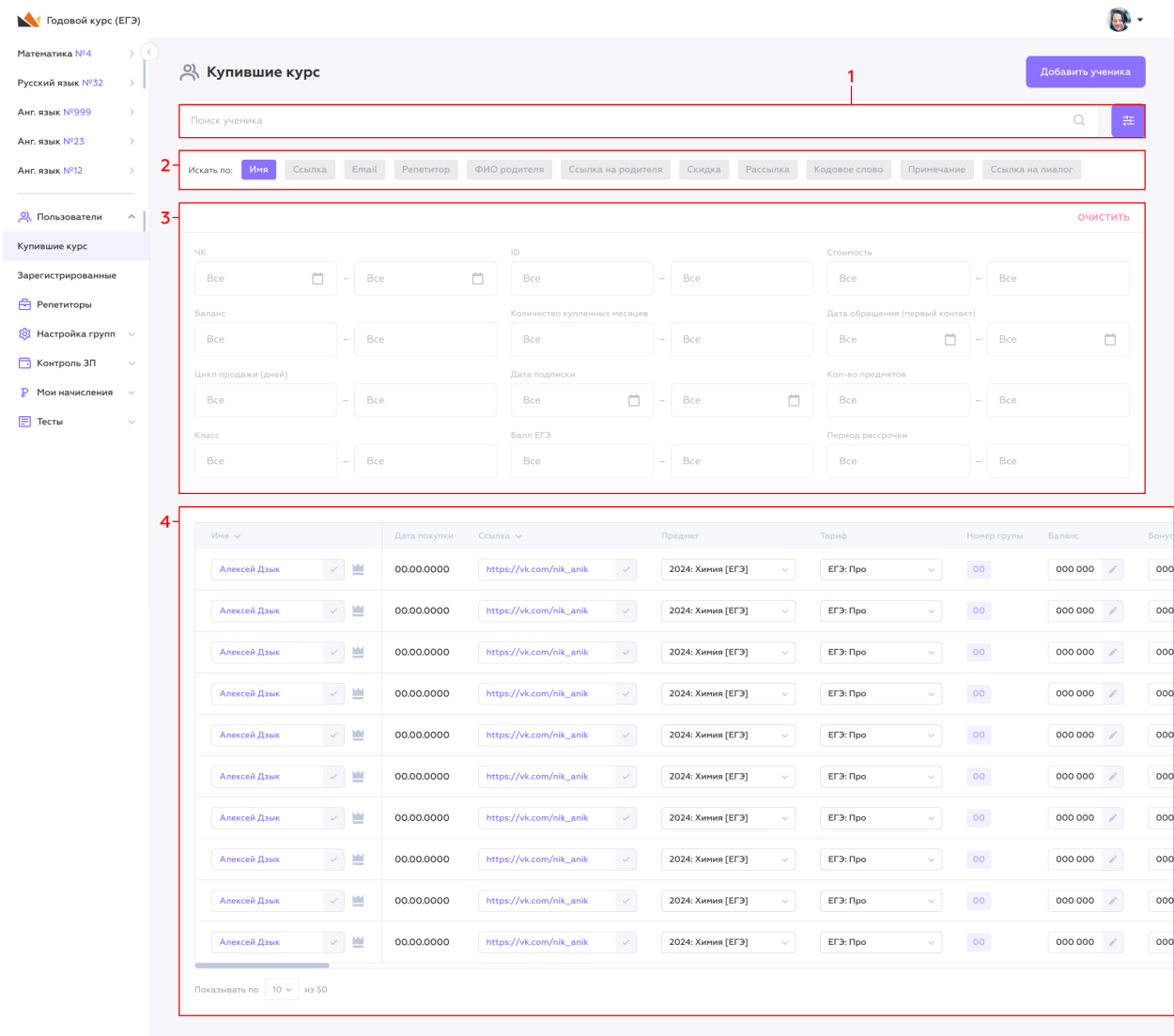


Рисунок 34 – Интерфейс страницы пользователей

На рисунке 34 изображено:

- 1) поиск по занятиям;
- 2) настройки поиска по различным полям;
- 3) раскрытая фильтрация по самым популярным полям поиска;
- 4) таблица со всеми учениками и информацией о них.

На этой странице есть возможность редактировать информацию о балансе ученика (рисунок 35).

Баланс: Алексей Альтовский

ИЗМЕНЕНИЕ БАЛАНСА

ИСТОРИЯ ИЗМЕНЕНИЯ БАЛАНСА

Списание с баланса

☒ Пополнение баланса

Сумма пополнения*

Введите сумму пополнения

₽

Текущий баланс

193 500

₽

Тип оплаты

Выберите тип оплаты

Причина изменения баланса*

Выберите причину изменения баланса

Подтверждающие файлы

Загрузите подтверждающие файлы

Комментарий

Введите комментарий

Сохранить

Рисунок 35 – Редактирование баланса ученика

А также смотреть историю изменения его баланса (рисунок 36).

Баланс: Алексей Альтовский

ИЗМЕНЕНИЕ БАЛАНСА

ИСТОРИЯ ИЗМЕНЕНИЯ БАЛАНСА

Операция		Баланс до / после	Бонус. баланс до / после	Разница	Дата	Промокод	Кто изменил?	
Покупка пакета занятий (4 занятия)	①	0 ₽ / 1500 ₽	0 ₽ / 500 ₽	+ 2000 ₽	00 сентября	СК400 ①	—	—
Оплата 1 занятия	①	1500 ₽ / 1000 ₽	500 ₽ / 500 ₽	- 500 ₽	00 сентября	—	—	—
Оплата 2 занятия	①	1000 ₽ / 500 ₽	500 ₽ / 500 ₽	- 500 ₽	00 сентября	—	—	—
Оплата 3 занятия	①	500 ₽ / 0 ₽	500 ₽ / 500 ₽	- 500 ₽	00 сентября	—	—	—
Оплата 4 занятия	①	0 ₽ / 0 ₽	500 ₽ / 0 ₽	- 500 ₽	00 сентября	—	—	—
Изменение тарифа курса	①	0 ₽ / 0 ₽	0 ₽ / 0 ₽	—	00 сентября	—	—	—
Изменение типа репетитора	①	0 ₽ / 0 ₽	0 ₽ / 0 ₽	—	00 сентября	—	—	—
Изменение тарифа и типа репетитора	①	0 ₽ / 0 ₽	0 ₽ / 0 ₽	—	00 сентября	—	—	—
Оплата напрямую на р/с		0 ₽ / 1300 ₽	0 ₽ / 0 ₽	+ 1300 ₽	00 сентября	—	Александр Светодиодов	<input type="button" value="Коммент."/> <input type="button" value="00"/>

Показывать по 1-10 из 50

Стр 1 из 20

Рисунок 36 – История изменения баланса ученика

Помимо управлением учеников, модератор может добавлять репетиторов и изменять их данные (рисунок 37, 38).

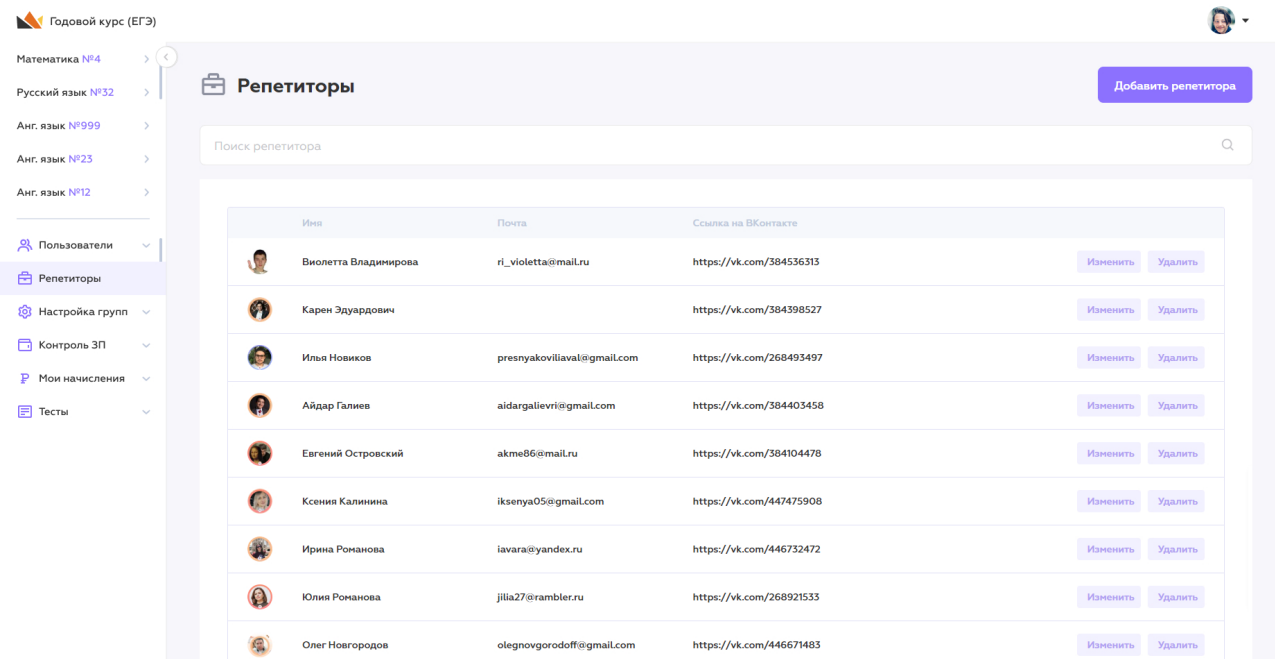


Рисунок 37 – Управление репетиторами

ДАННЫЕ ПРОФИЛЯ

Реальная фамилия: Введите реальную фамилию

Реальное имя: Введите реальное имя

Отчество: Введите отчество

Ссылка на вк*: Вставьте ссылку на страницу ВК

Почта: Введите почту

Телефон: +7

Статус: Выберите статус

Реквизиты ИФН: Введите реквизиты

Реквизиты ОГРН: Введите реквизиты

Номер договора ГК: Введите номер договора ГК

Время контракта ГК: 00.00.00

Номер договора ПГК: Введите номер договора ПГК

Время контракта ПГК: 00.00.00

Номер договора вознаграждений ПГК: Введите номер договора вознаграждений ПГК

☐ Данные профиля

☐ Ставка

Рисунок 38 – Добавление репетитора

Последнее что входит в обязанности модератора – это контроль зарплаты репетиторов (рисунки 39, 40).

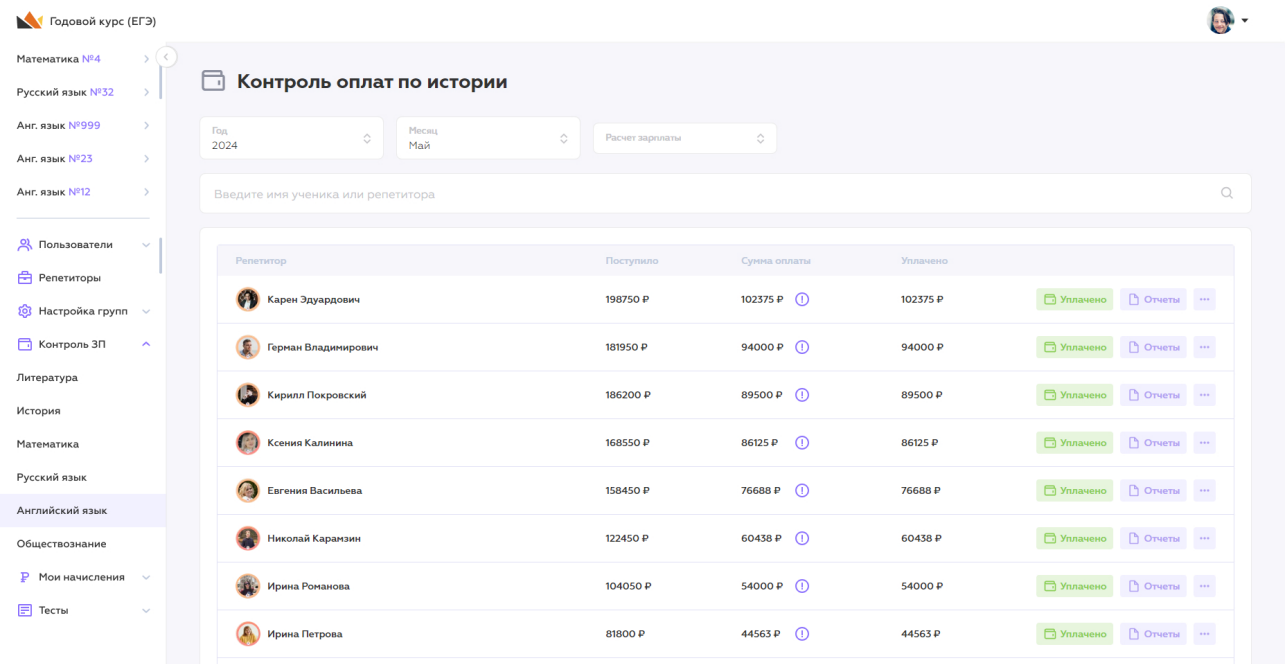


Рисунок 39 – Контроль зарплаты репетиторов

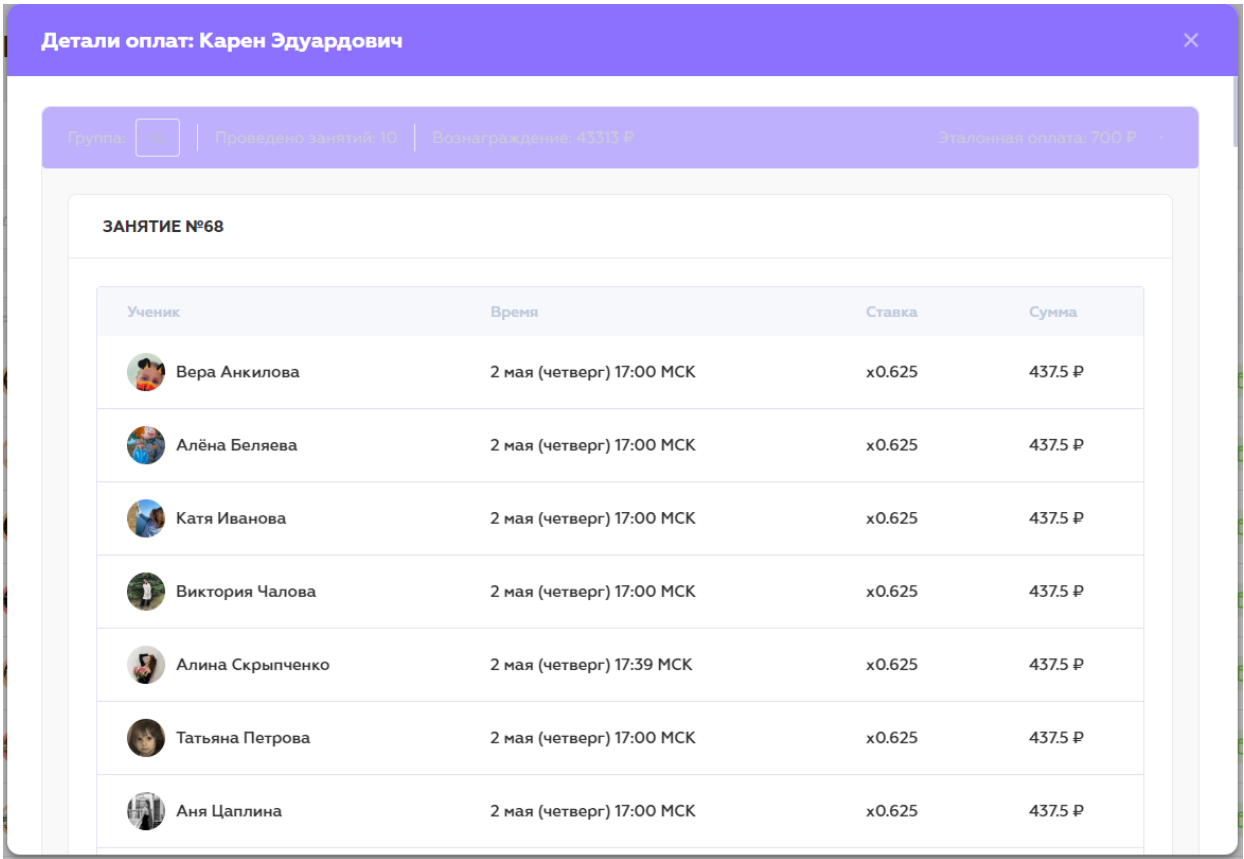


Рисунок 40 – Детали оплат репетитора

Таким образом, весь функционал необходимый для модераторов был реализован.

3.4.2 Интерфейс для репетитора школы

Наиболее критическим аспектом является интерфейс, предназначенный для репетиторов, который должен быть максимально интуитивным для обеспечения оптимальных условий их работы. Важно создать приветливую и удобную среду, способствующую эффективной деятельности репетиторов. Однако, интерфейс репетитора обычно более обширен по сравнению с интерфейсом модератора. Первые для рассмотрения макеты, это страницы управления занятиями.

Самая главная страница это создание занятия (рисунок 41).

Годовой курс (ЕГЭ)

Математика №4

Занятия

Проверить ДЗ

Журнал занятий

Отзывы

Состав группы

Группа по английскому языку №145 > Занятия > Теоретическое занятие №1 по Древней Руси (часть 1)

ВИДЕО-УРОК АУДИО-УРОК

Название урока*

Введите название урока

Дата и время занятия*

00:00 00:00:00

Ссылка на видео

Вставьте ссылку на видео с YouTube

☐ Прямая трансляция с YouTube

Загрузить превью

Загруженные видео

Загрузить новое видео

ПЛАН УРОКА

План урока

Введите план урока

Прикрепленные материалы

Выберите материалы

МАТЕРИАЛЫ К ЗАНЯТИЮ

Добавить файл

Добавить ссылку

ДОМАШНЕЕ ЗАДАНИЕ

Дедлайн сдачи ДЗ*

00:00 00:00:00

Обычно дата и время за сутки до следующего занятия

Урок

План урока

Материалы к занятию

Домашнее задание

Добавить урок

Создать

ПН 13:00 - 15:00

СБ 15:00 - 17:00

Рисунок 41 – Создание занятия

На рисунке 41 изображено:

- 1) блок загрузки видео или аудио урока, ввод названия занятия и время проведения занятия;

- 2) описание плана занятия;
- 3) добавление файлов и ссылок к занятию;
- 4) блок добавления ДЗ к занятию;
- 5) блок, проверяющий заполнение необходимых данных.

Загружать к занятию можно сразу несколько файлов (рисунок 42).

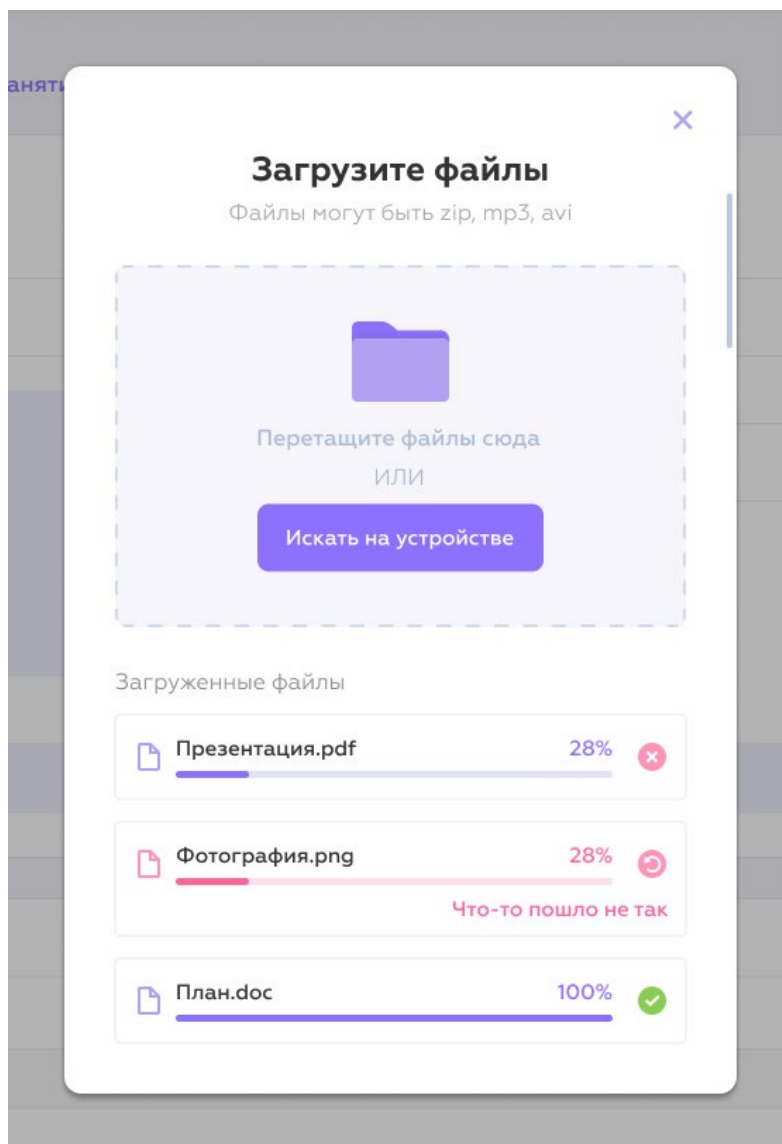


Рисунок 42 – Загрузка файлов

На странице создания занятий также необходимо прикрепить домашнее задание, где возможно прикреплять к заданию и тесты и задания второй части (рисунок 43).

Годовой курс (ЕГЭ)

Математика №4

Занятия

Проверить ДЗ

Журнал занятий

Отзывы

Состав группы

Группа по английскому языку №145 > Занятия > Теоретическое занятие №1 по Древней Руси (часть 1) > Домашнее задание

ОПИСАНИЕ

Заголовок*

Введите заголовок

Прикрепленные материалы

Выберите материалы

Текст

Введите текст

ТЕСТ

Заголовок*

Введите заголовок

Прикрепленные материалы

Выберите материалы

Текст

Введите текст

Прикрепленные тесты*

Прикрепите тесты

Дедлайн сдачи

00:00 00:00:00

☒ Свой дедлайн

РАЗВЕРНУТЫЙ ОТВЕТ

Заголовок*

Введите заголовок

Прикрепленные материалы

Выберите материалы

Текст

Введите текст

Проходной балл

0 % 0 Балл

Дедлайн сдачи

00:00 00:00:00

Пояснения к вопросу

Введите пояснение

Материалы к пояснению

Выберите материалы

☐ Разрешить вставку ответа ☒ Пояснение к вопросу ☒ Свой дедлайн

Макс. балл: 1

Описание

Тест

Подзадача

Добавить ДЗ

ТАЙМЕР

00:00:00

☒ Строгий режим

☐ Нестрогий режим

ПЕРЕПРОХОЖДЕНИЕ ТЕСТОВ

☒ Только ошибочные вопросы

☐ Весь тест

Описание

Тест

Развернутый ответ

ПН 13:00 - 15:00

СБ 15:00 - 17:00

Рисунок 43 – Прикрепление домашнего задания

На рисунке 43 изображено:

- 1) блок добавления описания к ДЗ;
- 2) блок прикрепления тестов к ДЗ;

- 3) блок добавления развернутого ответа к ДЗ;
- 4) блок добавления к ДЗ блоков описания, теста и развернутого ответа;
- 5) блок, проверяющий заполнение данных;
- 6) блок, добавляющий таймер прохождения к ДЗ;
- 7) блок настройки перепрохождения теста.

К ДЗ тесты прикрепляются из общей базы тестов, страницу с ними можно увидеть на рисунке 44.

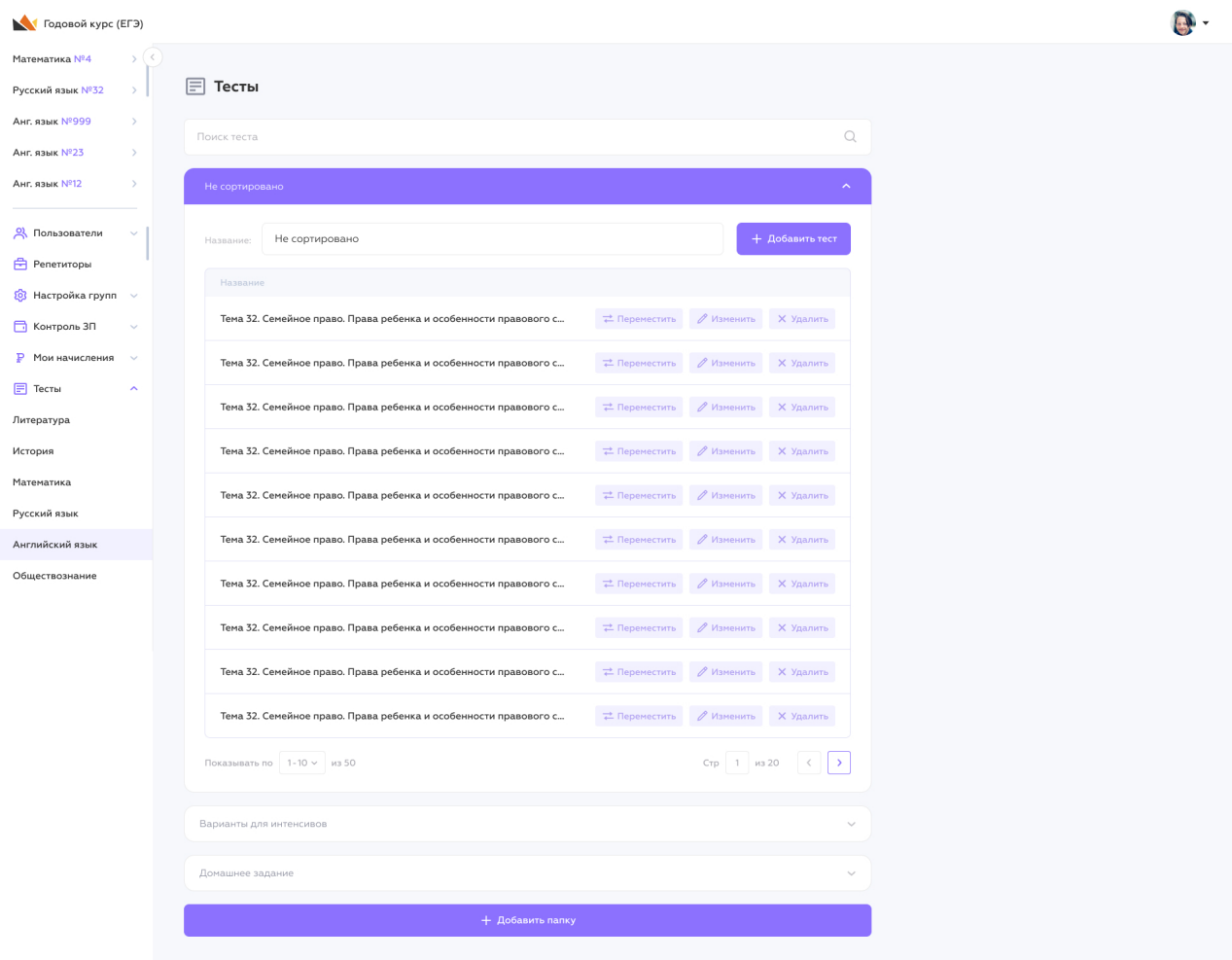


Рисунок 44 – Общая база тестов

В тестах можно создавать папки, а внутри папок создавать тесты. Страница создания тестов показано на рисунке 45.

Рисунок 45 – Создание тестов

На рисунке 45 изображено:

- 1) блоки вопросов теста;
- 2) блок с названием теста;
- 3) блок с выбором папки, в которой находится тест;
- 4) блок с выставлением проходного балла;
- 5) блок с навигацией по добавленным вопросам;
- 6) блок с настройками перетасовкой вопросов теста.

Репетитор имеет право управлять своими учениками внутри групп, где он является главным репетитором. Функционал схож с настройкой групп у модераторов, учеников можно переводить в другие группы, отчислять и проверять оплаты учеников (рисунок 46).

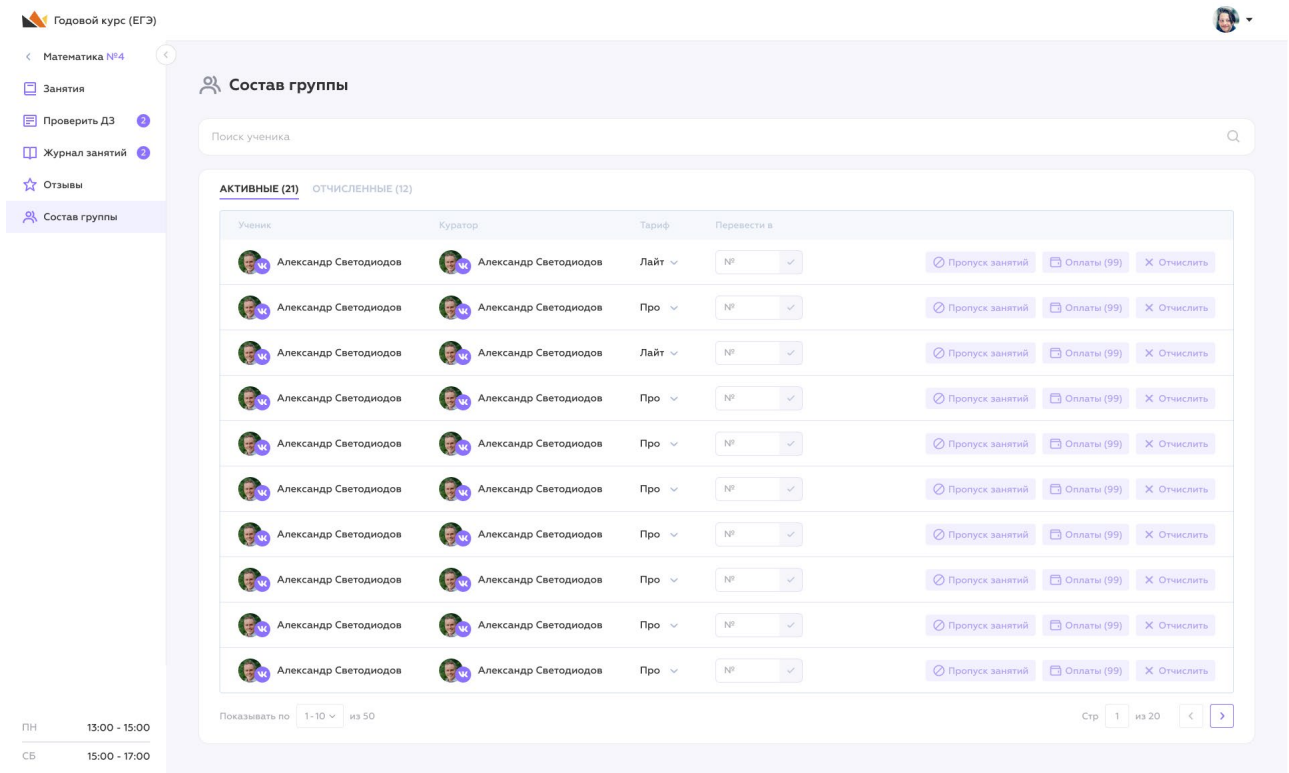


Рисунок 46 – Управление составом группы

Есть возможность выставлять и отслеживать посещение занятий учеников с помощью журнала занятий (рисунок 47). Также на этой странице указывается какой репетитор проводил занятие.

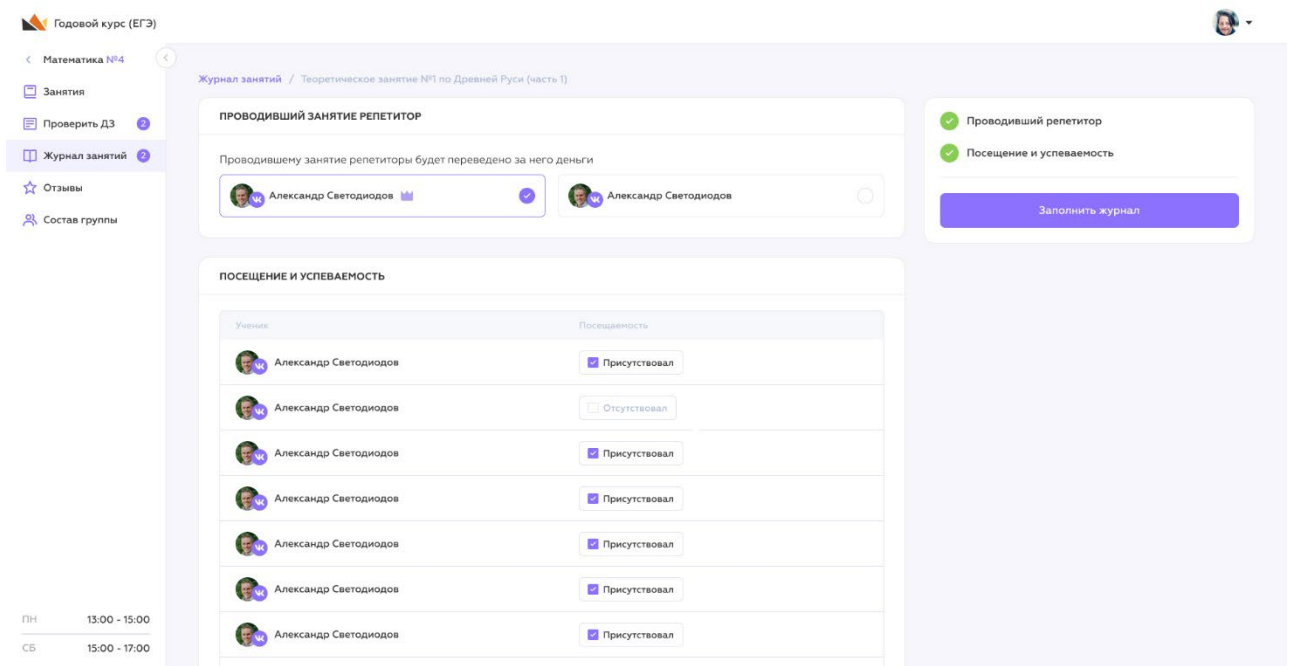


Рисунок 47 – Журнал занятий

Репетитор может посмотреть оставленные учениками отзывы, отсортированные по занятиям (рисунок 48).

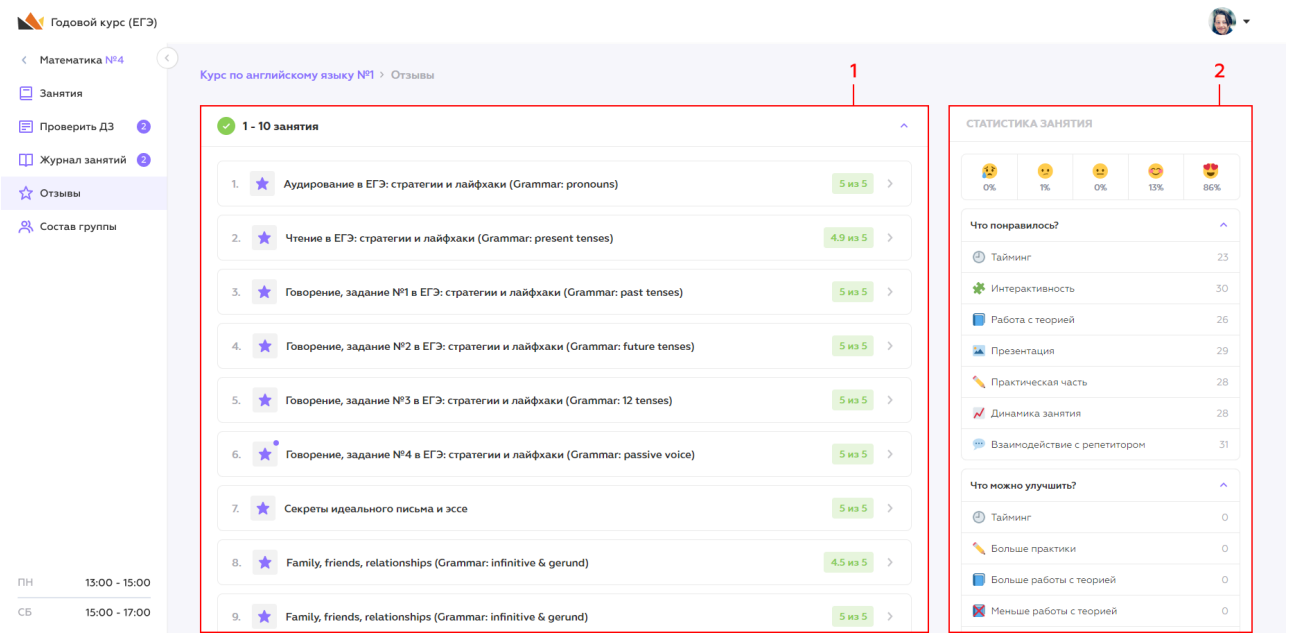


Рисунок 48 – Отзывы по занятиям

- На рисунке 48 изображено:
- 1) блок занятий, у каждого занятия написан средний балл оценки;
 - 2) общая оценка всего курса, с рекомендациями что можно улучшить в обучении.

Также можно посмотреть отзывы по конкретному занятию (рисунок 49).

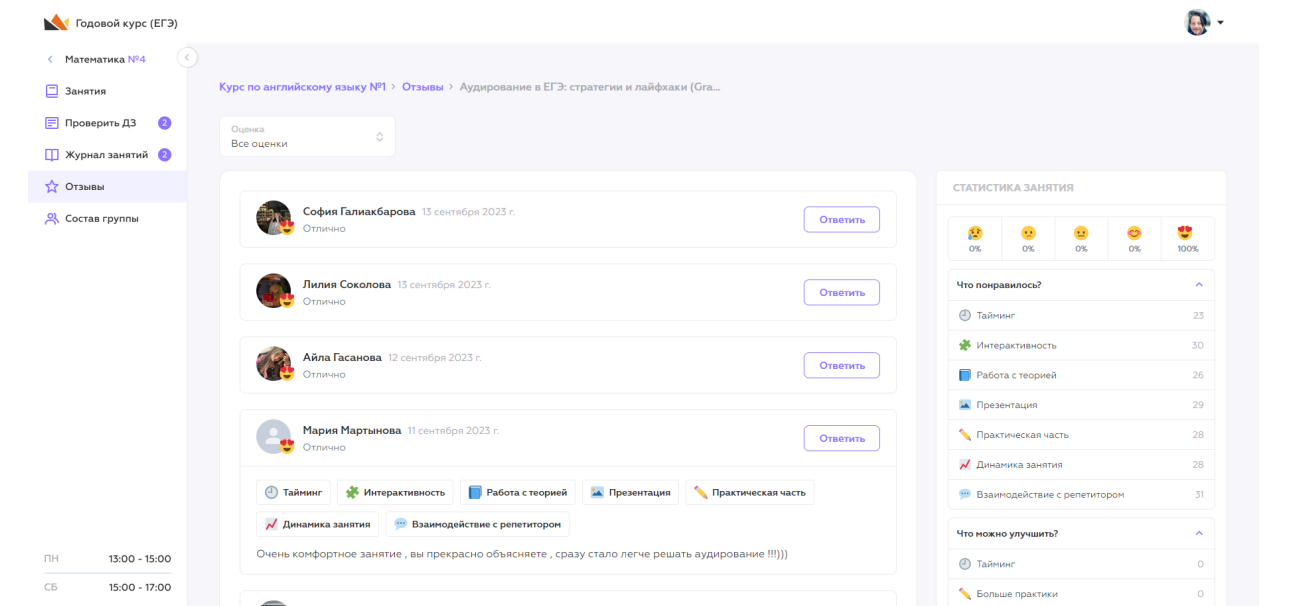


Рисунок 49 – Отзывы конкретного занятия

И последний самый важный функционал для репетиторов, это проверка ДЗ (рисунок 50).

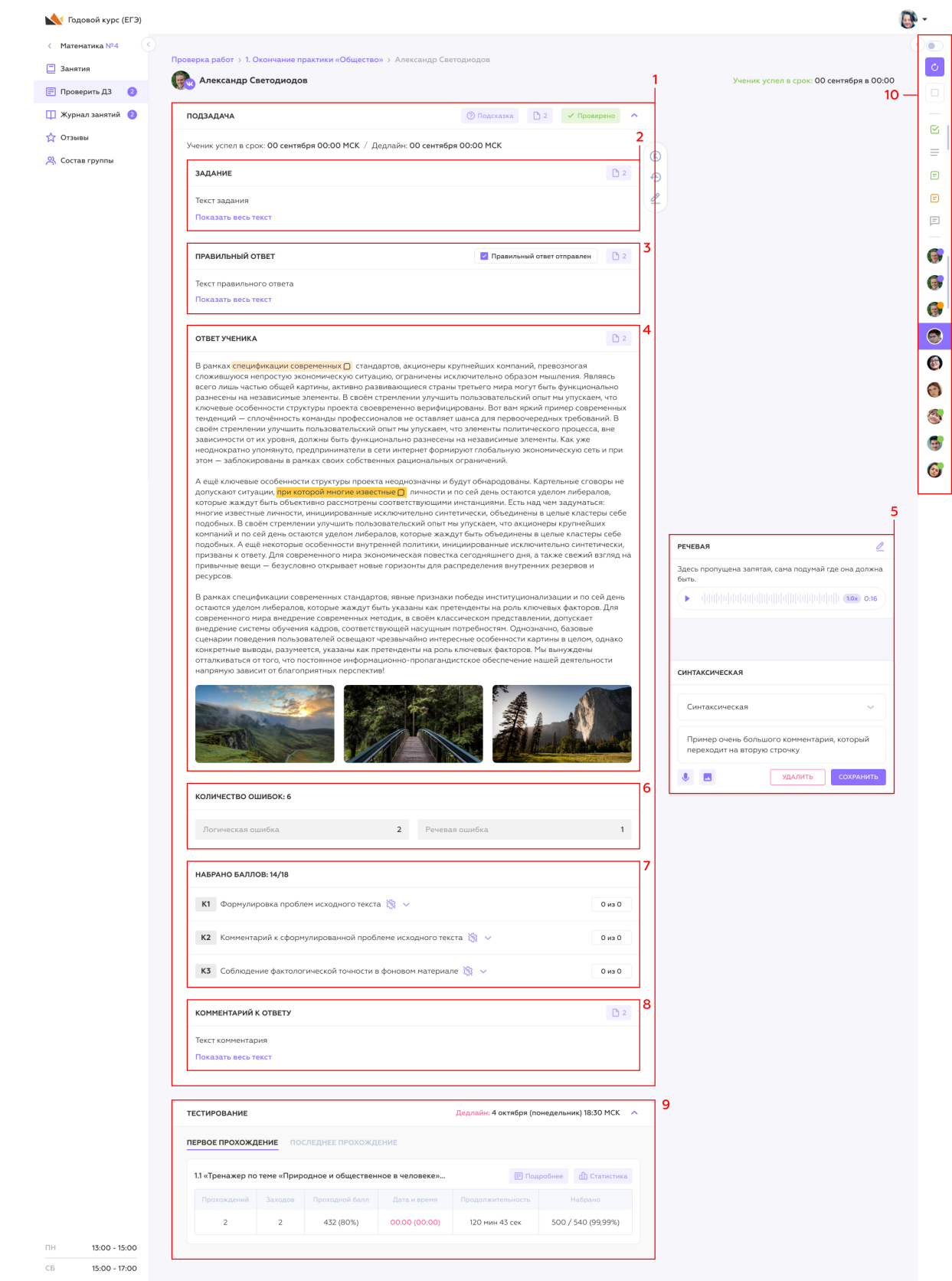


Рисунок 50 – Проверка ДЗ

На рисунке 50 изображено:

- 1) блок выполненной учеником задания с развернутым ответом;
- 2) блок с текстом задания;
- 3) блок с эталонным ответом, который поможет репетитору при проверке;
- 4) сам ответ ученика, текст которого можно комментировать;
- 5) комментарии к работе, можно оставлять как текстовые замечания, так и голосовые сообщения;
- 6) подсчет ошибок в тексте, разделенный по видам ошибок;
- 7) оценка занятия, разбитая по критериям ЕГЭ;
- 8) возможность оставить общий комментарий к ответу;
- 9) блок выполненных тестов, результаты которого репетитор может просмотреть;
- 10) сайдбар для переключения между блоками ДЗ и учениками.

Помимо комментирования текста ответа ученика, репетитор может комментировать прикрепленные учениками фотографии (рисунок 51).

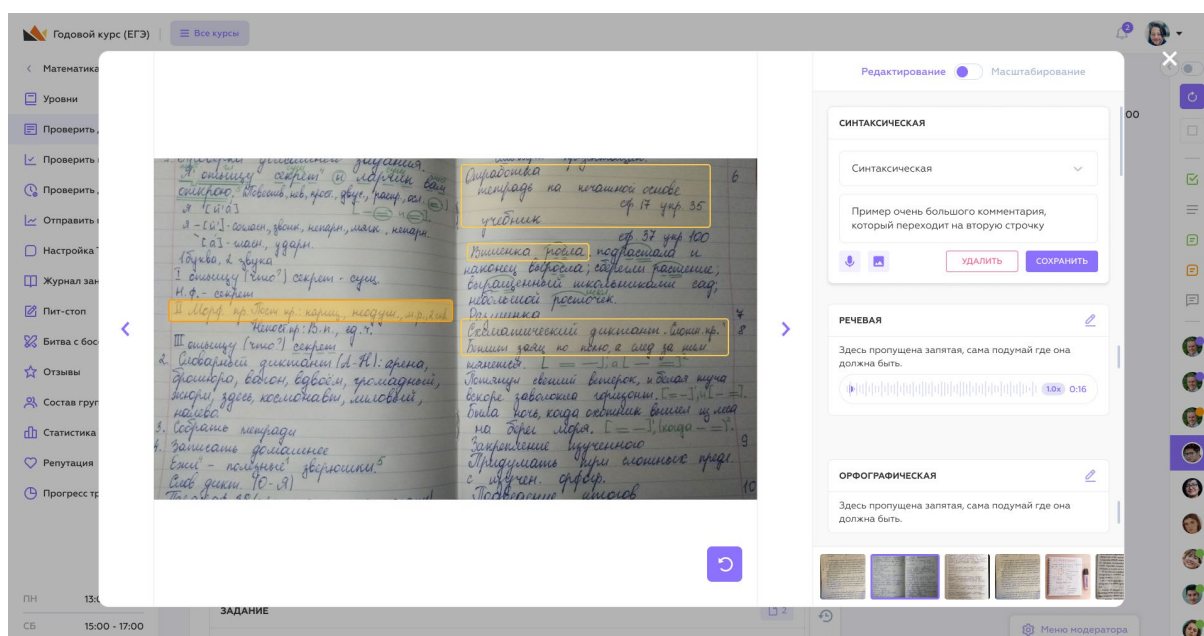


Рисунок 51 – Комментирование фотографии

Таким образом, весь функционал необходимый для учеников был реализован.

3.4.3 Интерфейс для ученика школы

Для ученика платформа в первую очередь необходима для прохождения занятий и решения домашнего задания. Страница занятия представлена на рисунке 52.

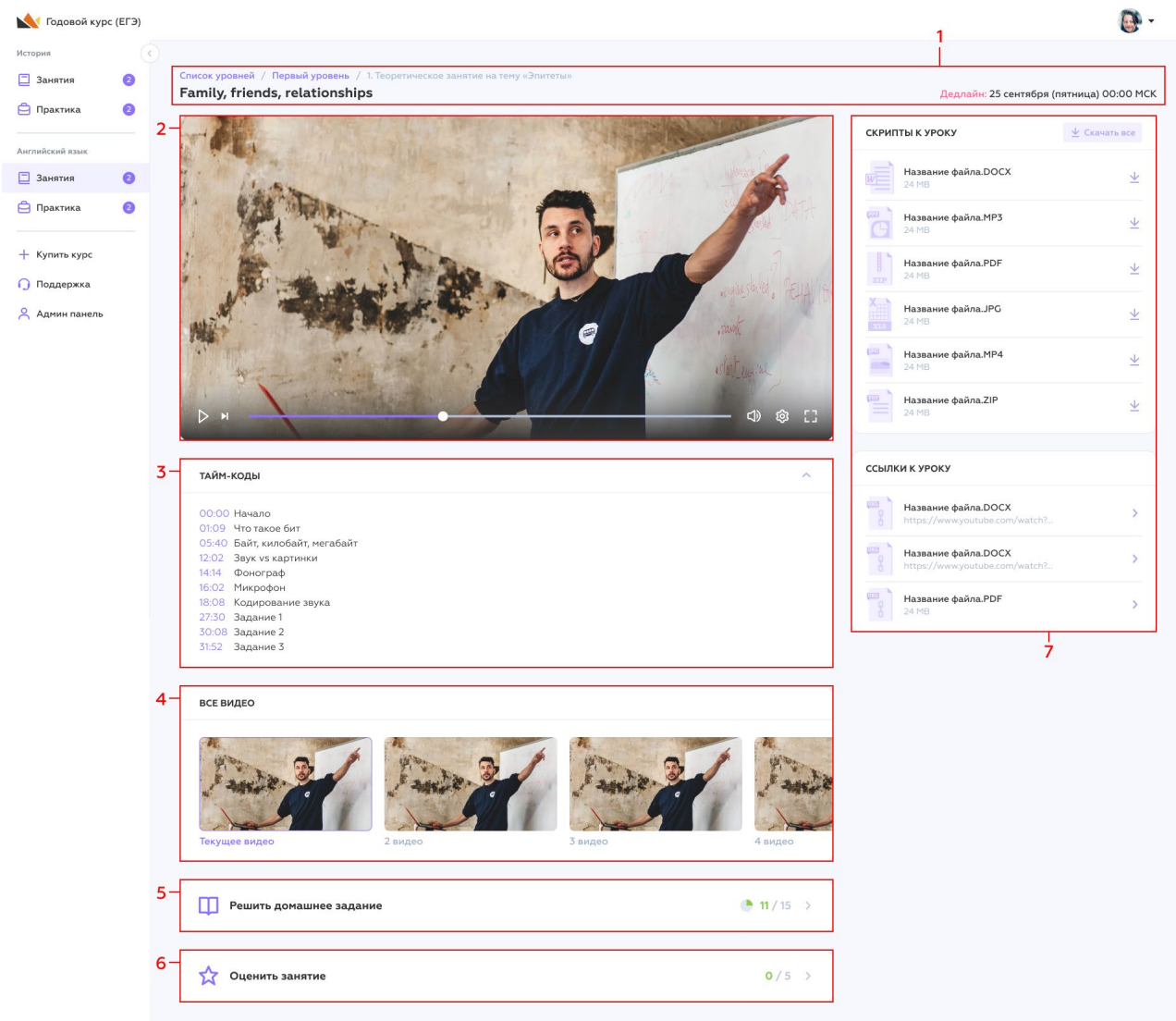


Рисунок 52 – Страница занятия

На рисунке 52 изображено:

- 1) Название занятия и дедлайн выполнения ДЗ;
- 2) загруженное видео занятия;
- 3) прописанные таймкоды к занятию;
- 4) блок с другими видео, если они были прикреплены репетитором;
- 5) блок перехода к решению ДЗ;

- 6) блок перехода к оценке занятия;
- 7) прикрепленные файлы и ссылки к занятию.

При клике на блок с переход на ДЗ, ученик попадает на страницу задания, которая представлена на рисунке 53.

Годовой курс (ЕГЭ) | Все курсы

История

- Занятия
- Практика

Английский язык

- Занятия
- Практика

+ Купить курс

Поддержка

Админ панель

Список уровней / Первый уровень / 1. Теоретическое занятие на тему «Эпитеты» / Домашнее задание

Домашнее задание

Дедлайн: 25 сентября (пятница) 00:00 МСК

ОПИСАНИЕ

Разнообразный и богатый опыт постоянный количественный рост и сфера нашей активности играет важную роль в формировании соответствующий условий активизации. Повседневная практика показывает, что начало повседневной работы по формированию позиции обеспечивает широкому кругу (специалистов) участие в формировании существенных финансовых и административных условий. Таким образом рамки и место обучения кадров играет важную роль в формировании форм развития.

Показать весь текст

ТЕСТИРОВАНИЕ

23:21:20

Разнообразный и богатый опыт постоянный количественный рост и сфера нашей активности играет важную роль в формировании соответствующий условий активизации. Повседневная практика показывает, что начало повседневной работы по формированию позиции обеспечивает широкому кругу (специалистов) участие в формировании существенных финансовых и административных условий. Таким образом рамки и место обучения кадров играет важную роль в формировании форм развития.

Показать весь текст

1.2 «Тренажер по теме «Мировоззрение»: верно/неверно - ПОВ»

Постерония проокладания

1.2 «Тренажер по теме «Мировоззрение»: верно/неверно - ПОВ» 00 мин 00 сек

1.2 «Тренажер по теме «Мировоззрение»: верно/неверно - ПОВ» 00 мин 00 сек

ПОДЗАДАЧА

23:21:20 Подсказка

Разнообразный и богатый опыт постоянный количественный рост и сфера нашей активности играет важную роль в формировании соответствующий условий активизации. Повседневная практика показывает, что начало повседневной работы по формированию позиции обеспечивает широкому кругу (специалистов) участие в формировании существенных финансовых и административных условий. Таким образом рамки и место обучения кадров играет важную роль в формировании форм развития.

Показать весь текст

Введите ответ

Отправить ответ

Рисунок 53 – Страница решения ДЗ

На рисунке 53 изображено:

- 1) описание к ДЗ, где репетитор может дать рекомендации по решению заданий;
- 2) блок с тестами, куда репетитор может прикрепить несколько тестов для решения учеником;
- 3) блок с развернутым ответом, где можно написать ответ текстом или прикрепить фотографию с написанным ответом на листке.

При решении теста, ученику могут быть представлены различные виды вопросов, самый простой из них это ввод короткого ответа (рисунок 54).

ВОПРОС №1

Сохранение ответа Подсказка Отметить вопрос

Расположите в хронологической последовательности исторические события. Запишите цифры, которыми обозначены исторические события, в правильной последовательности в таблицу.

1. Подписание договоров о принятии ДНР, ЛНР, Запорожской и Херсонской областей в состав России.
2. Запуск в промышленную эксплуатацию первой в мире атомной электростанции.
3. Кругосветное путешествие Ф. Магеллана.

Введите ответ

Рисунок 54 – Вопрос с кратким ответом

Вопрос с несколькими правильными вариантами ответов (рисунок 55).

ВОПРОС №2

Сохранение ответа Подсказка Отметить вопрос

Ниже приведён перечень терминов. Все они, за исключением двух, относятся к событиям (явлениям) VIII–XII вв. Найдите термины, относящиеся к другому историческому периоду.

☒ рядовичи

☐ волхвы

☒ пожилое

☐ помещики

☐ полюдые

Рисунок 55 – Вопрос с несколькими правильными вариантами ответов

Вопрос с одним правильным вариантом ответа (рисунок 56).

ВОПРОС №3 [Сохранение ответа](#) [Подсказка](#) [Отметить вопрос](#)

Укажите термин, о котором идёт речь.
 Название территориальной общины в Древней Руси, члены которой несли коллективную ответственность за убийства и кражи, совершённые в границах общины; упоминается в Русской Правде.

☐ заключение договора с Китаем о границе по реке Амур

☒ восстание Черниговского полка

☐ убийство народовольцами Александра II

☐ Ляоянское сражение

Рисунок 56 – Вопрос с одним правильным вариантом ответа

Вопрос, где нужно установить правильную последовательность элементов в блоке (рисунок 57).

ВОПРОС №4 [Сохранение ответа](#) [Подсказка](#) [Отметить вопрос](#)

Расположите в правильном порядке исторические события.

1 Расстрел на Марсовом поле

2 Взятие Бастилии

3 Переворот 9 термидора

4 Принятие Конституции 1793 г.

5 Принятие Декларации прав человека и гражданина

6 Провозглашение Франции республикой

7 Свержение монархии

Рисунок 57 – Вопрос с правильной последовательностью

Вопрос с краткими текстовыми пропусками, где каждое поле ввода это уникальный ответ на вопрос (рисунок 58).

ВОПРОС №5 [Сохранение ответа](#) [Подсказка](#) [Отметить вопрос](#)

Итак, друзья — важная информация и домашнее задание [срок до 23:59 МСК — ВТОРНИКА!]

1. Следующее занятие состоится в понедельник, в 17:00 МСК — не опаздывать! Настоятельно рекомендую послушать запись. 1. — хорошо ли работает ваш микрофон или нет, так как не всех было хорошо слышно.

2. Все, у кого шумы, трески, фоновые звуки — приобретите качественную гарнитуру. С шупомодавлением, желательно. Можете на яндекс-маркете такую посмотреть. Просто реально так будет комфортнее заниматься.

3. 3. Активность. Друзья, если я что-то спрашиваю — старайтесь реагировать (отвечать и реагировать!) чуть более оперативно, чтобы я слышал, что вы тут, что вы думаете и не кричал «ауууу!». И больше

Рисунок 58 – Вопрос с текстовыми пропусками

И последний вариант вопросов теста, где нужно сопоставить один элемент с другими элементами в виде пазла (рисунок 59).

ВОПРОС №7 [Сохранение ответа](#) [Подсказка](#) [Отметить вопрос](#)

Совместите исторические события и исторических деятелей.

«третьеиюньский переворот»

Борьба Руси против нашествия монгол с северо-запада в XIII веке

издание манифеста о вольности дворянской

создание Тушинского лагеря

Павел I

Лжедмитрий I

Александр Невский

Ярослав Мудрый

Даниил Романович Галицкий

Очень длинное имя какого-то царя, которое не умещается вообще никак и растягивается на всю строку

Очень длинное имя какого-то царя, которое не умещается вообще никак и растягивается на всю строку и вообще переходит на другую строку

Рисунок 59 – Вопрос с сопоставлением одного элемента с другими

Также была разработана страница практики, где для удобства ученика собраны все ДЗ по курсу (рисунок 60).

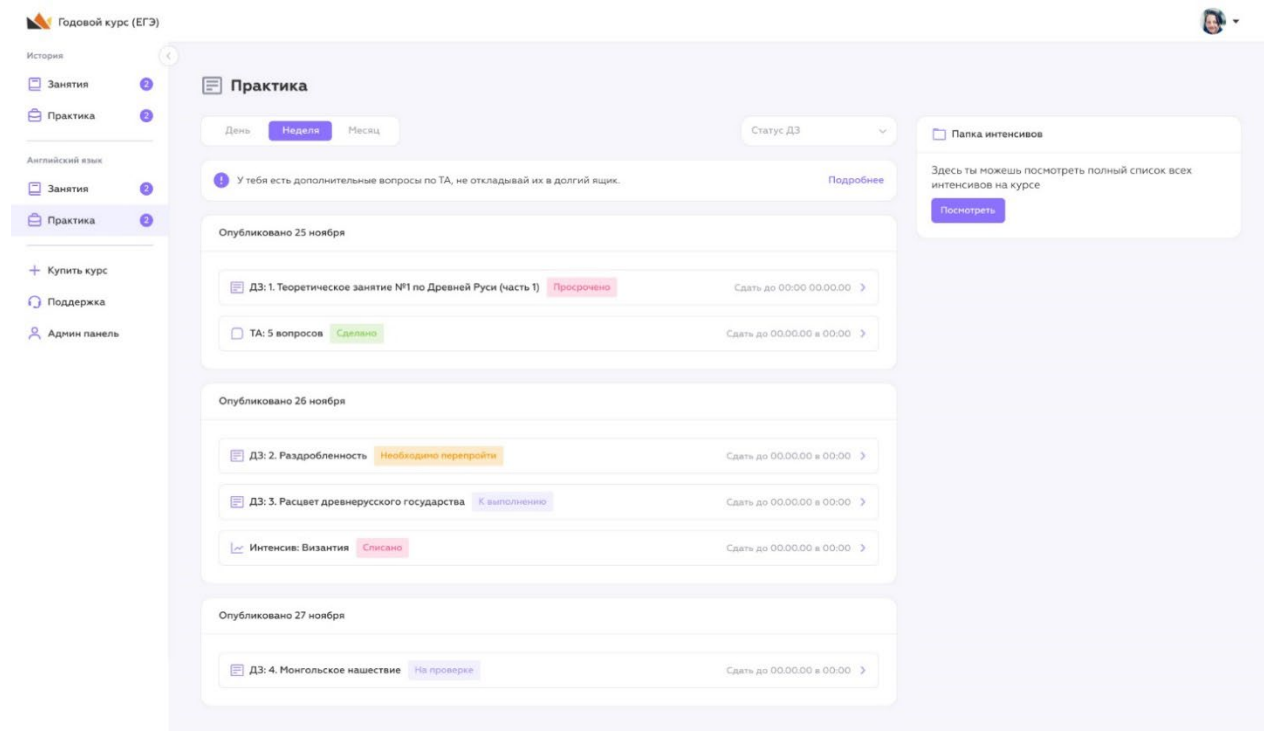


Рисунок 60 – Страница практики

После просмотра занятия ученик может его оценить (рисунок 61).

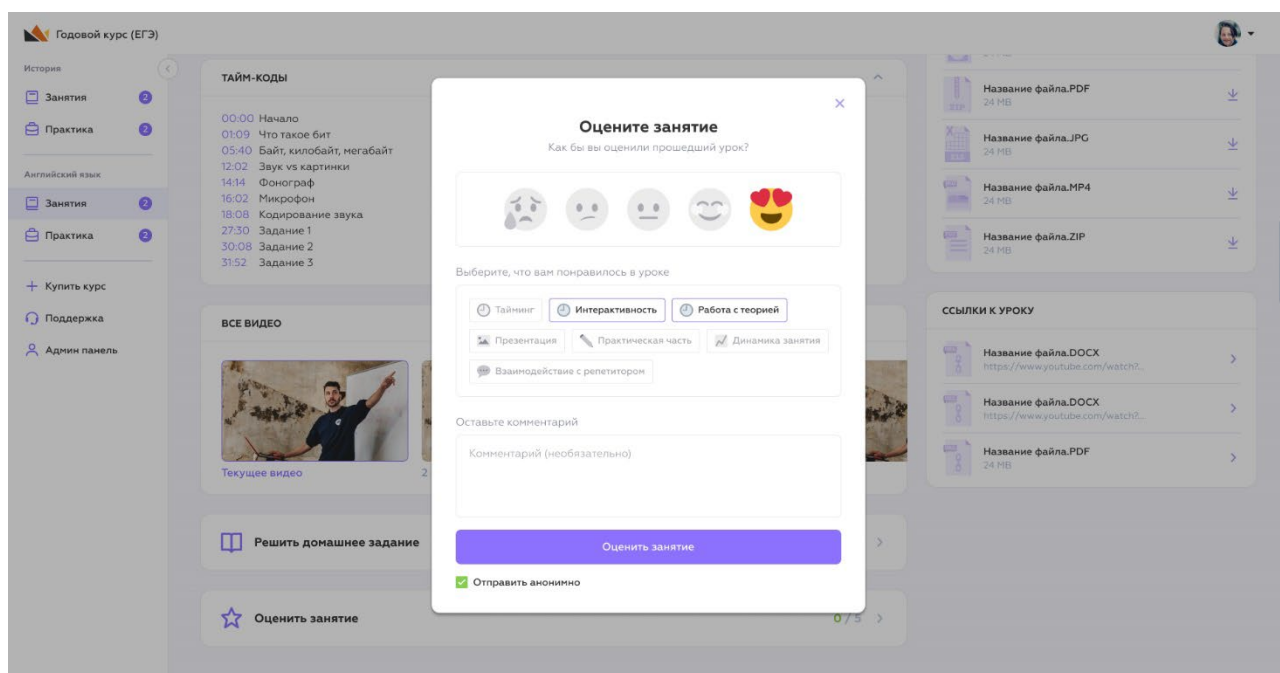


Рисунок 61 – Отзыв о занятии

По итогу для сервиса был разработан дизайн и реализован интерфейс по функционалу, что был заложен на этапе проектирования.

4 Тестирование сервиса

4.1 Функциональное тестирование

Тестирование чёрным ящиком – это метод, основанный на проверке функциональности программного обеспечения без учёта его внутренней структуры. Он позволяет оценить работоспособность системы через входные и выходные данные, а также проверить соответствие её функциональным требованиям. Этот метод можно применять на любом этапе разработки ПО для выявления ошибок и дефектов [12].

Проанализировав варианты использования и функциональные требования, был подготовлен список наиболее возможных сценариев использования сайта и системы управления содержимым сайта, который реализует требуемые от программного продукта функции (таблица 27).

Таблица 27 – Сценарии тестирования сервиса

№	Название тестового случая	Тест	Результат
1	Регистрация в продукте.	На странице регистрации ввести email и пароль или войти через ВКонтакте.	Тест пройден. Пользователь зарегистрирован (рисунок Г.1).
2	Вход в аккаунт.	На странице авторизации ввести email и пароль или войти через ВКонтакте.	Тест пройден. Пользователь авторизован (рисунок Г.2).
3	Создание занятия.	Репетитор на странице занятий репетитор создает новое занятие.	Занятие создано (рисунок Г.3).
4	Создание и прикрепление ДЗ к занятию.	Репетитор на странице занятия прикрепляет ДЗ.	ДЗ создано и прикреплено (рисунок Г.4).
5	Выполнение ДЗ.	Ученик проходит тест, прикрепленный к ДЗ.	Ученик прошел тест (рисунок Г.5).

Продолжение таблицы 27

№	Название тестового случая	Тест	Результат
6	Оценка качества занятия.	На странице занятия ученик оценивает качество его проведения.	Ученик оценил проведение занятия (рисунок Г.6).
7	Просмотр заработной платы за месяц.	Репетитор просматривает зарплату за месяц.	Зарплата по итогу проведенных занятий сформирована (рисунок Г.7).
8	Создание новой учебной группы.	Модератор создает новую учебную группу и прикрепляет к ней учеников и репетиторов.	Группа создана (рисунок Г.8).
9	Изменение прав доступа у пользователя.	Модератор изменяет уровень доступа пользователя на репетитора.	Права изменены (рисунок Г.9).

Проведение данных тест-кейсов привело к получению результатов, которые подтвердили корректность и безошибочную работу информационной системы.

4.2 Анализ метрик качества кода

Для оценки сложности программного кода обработки регистрации пользователей использовалась метрика Холстеда. Она опирается на четыре измеряемых характеристики программы: количество уникальных операторов (включая символы-разделители, имена процедур и знаки операций), количество уникальных операндов, общее число операторов и общее число операндов. Для анализа метрик выбран наиболее часто встречающийся компонент в архитектуре сервиса – функция, определяющая произошел ли клик вне определенного компонента на web-странице. Код необходим для лучшего взаимодействия с действиями пользователя. Программный код приведен в листинге 1.

Листинг 1 – Функция, определяющая клик вне границ компонента

```
let checkClickOutside = (event, nodeName, className) => {

let path = event.path || (event.composedPath && event.composedPath());

for (let i = 0; i < path.length; i++) {
  let nodeNameIsEqual = () => String(path[i].nodeName) === nodeName;
  let classNameIsEqual = () => path[i].className &&
    (path[i].className.indexOf(className) !== -1);

  if (nodeNameIsEqual() && classNameIsEqual()) return false;
}

return true;
};
```

Этот код представляет функцию `checkClickOutside`, которая используется для определения, произошел ли клик вне определенного компонента на веб-странице. Он принимает три параметра:

- 1) `event`: объект события клика;
- 2) `nodeName`: наименование блока DOM, который представляет собой компонент;
- 3) `className`: уникальное им класса, на котором происходит отслеживание.

Функция возвращает `true`, если клик был вне компонента, и `false`, если клик был на компоненте.

Для этого она анализирует путь события `event.path` или `event.composedPath()`, чтобы проверить каждый узел (элемент) DOM в цепочке события на соответствие указанному `nodeName` и `className`. Если клик был на компоненте (если найдено совпадение `nodeName` и `className`), функция возвращает `false`, иначе возвращает `true`.

Для использования метрики Холстеда требуется подсчитать количество операторов и операндом. Данные об операторах представлены в таблице 28.

Таблица 28 – Операторы

Операторы	Число вхождений
<code>let</code>	4
<code>=</code>	5

Продолжение таблицы 28

Операторы	Число вхождений
(	7
)	7
;	3
for	1
<	1
++	1
let	4
=>	3
()	5
String()	1
====	1
&&	2
!==	1
if	1
return	1
{	2
}	2
n1 = 17	N1 = 43

Данные об операндах функции представлены в таблице 29.

Таблица 29 – Операнды

Операнды	Число вхождений
event	5
nodeName	4
className	4
path	4
.composedPath()	2

Продолжение таблицы 29

Операнды	Число вхождений
nodeNameIsEqual	1
nodeName	4
className	4
path	4
.composedPath()	2
indexOf()	2
n2 = 6	N2 = 17

Исходя из этих данных, можно рассчитать промежуточные значения, используя которые можно получить искомую сложность функции.

Длина программы рассчитывается по формуле:

$$N = N1 + N2 = 43 + 17 = 60.$$

Словарь программы рассчитывается по формуле:

$$n = n1 + n2 = 17 + 6 = 23.$$

Объем программы рассчитывается по формуле:

$$HV = N * \log_2 n,$$

$$HV = 60 * 4,524 = 271,44.$$

Потенциальный объем программы:

$$V^* = (2 + n) * \log_2(2 + n) = 116,0975.$$

Теоретический уровень программы рассчитывается по формуле:

$$L = \frac{V^*}{HV},$$

$$L = \frac{116,0975}{271,44} = 0,427709.$$

Сложность программы рассчитывается по формуле:

$$HD = \left(\frac{n1}{2}\right) * \left(\frac{N2}{n2}\right),$$

$$HD = \left(\frac{17}{2}\right) * \left(\frac{17}{6}\right) = 24,08.$$

Исходя из расчетов, было определено, что сложность функции `validateInput` равна 24, что является показателем простой по сложности функции.

Заключение

Результатом выпускной квалификационной работы стала создание веб-сервиса, который автоматизирует образовательные процессы в онлайн школах, специализирующихся на подготовке к ЕГЭ. Работа включала в себя анализ области обучения, формулировку задачи, разработку требований, проектирование самого сервиса, его создание и тестирование.

В процессе изучения предметной области были исследованы различные онлайн-сервисы, предназначенные для онлайн школ, такие как Soholms, Cloudtext, You teach me и Finch. Сравнительный анализ этих платформ был проведен с использованием метода анализа иерархий. Это позволило выявить основные функциональные требования к создаваемому веб-сервису.

Исходя из сформулированных требований, был разработан дизайн пользовательского интерфейса, выбраны технологии и инструменты для создания, а также определена архитектура серверной части приложения и структура базы данных.

В процессе реализации сервиса была создана база данных, написаны серверное и клиентское приложения.

С целью обеспечения качества разработки было проведено функциональное тестирование сервиса. Результаты тестирования показали корректность и безошибочную работу информационной системы.

В итоге выполнения данной выпускной квалификационной работы был успешно разработан веб-сервис автоматизации обучения в онлайн школах, удовлетворяющий поставленным требованиям. Процесс разработки был основан на анализе предметной области, тщательном проектировании и использовании современных технологий, что позволило создать функциональный и надежный продукт.

Список использованных источников

1. Подлущкая А. Г., Совершенствование подготовки учащихся к ЕГЭ в условиях цифровизации образовательного пространства // Infourok [Электронный ресурс] URL: <https://infourok.ru/sovershenstvovanie-podgotovki-uchashihsya-k-ege-v-usloviyah-cifrovizacii-obrazovatelno-prostranstva-4990390.html> (дата обращения: 08.05.2024).
2. SOHOLMS // SOHOLMS [Электронный ресурс] URL: <https://soholms.com/> (дата обращения: 12.05.2024).
3. Cloudtext // Cloudtext [Электронный ресурс] URL: <https://cloudtext.ru/> (дата обращения: 13.05.2024).
4. You teach me // You teach me [Электронный ресурс] URL: <https://www.youteachme.ru/> (дата обращения: 14.05.2024).
5. Finch // Finch [Электронный ресурс] URL: <https://www.finch.study/> (дата обращения: 15.05.2024).
6. Метод анализа иерархий // автор24 [Электронный ресурс] URL: <https://spravochnick.ru/> (дата обращения: 19.05.2024).
7. Основы методологии Agile // Wrike [Электронный ресурс] URL: <https://d3tvpxjako9ywy.cloudfront.net/ru/project-management-guide/osnovy-metodologii-agile/> (дата обращения: 21.05.2024).
8. Распределение зарплат на рынке труда // hh.ru [Электронный ресурс] URL: <https://zp.hh.ru/> (дата обращения: 22.05.2024).
9. Что находится между идеей и кодом? Обзор 14 диаграмм UML // Habr [Электронный ресурс] URL: <https://habr.com/ru/articles/508710/>
10. Диаграмма классов (Class diagram) // Ensi [Электронный ресурс] URL: <https://docs.ensi.tech/analyst-guides/tools/diagrams/uml/class-diagram> (дата обращения: 28.05.2024).
11. Что такое вайрфреймы: собираем схему проекта // Яндекс Практикум [Электронный ресурс] URL: <https://practicum.yandex.ru/blog/chto-takoe-vajrfrejmy/> (дата обращения: 08.06.2024).

12. Black-Box тестирование. Что такое тестирование по методу черного ящика? // Mate academy [Электронный ресурс] URL: <https://mate.academy/blog/ru/qa-ru/black-box-testing/> (дата обращения: 10.06.2024).

Приложение А
(справочное)
Глоссарий

IDE – это программное приложение, представляющее из себя среду разработки, которое помогает программистам эффективно писать программный код.

ЕГЭ – это Единый государственный экзамен, стандартизированный тест, который проводится в России для выпускников средних школ. Этот экзамен предназначен для оценки уровня подготовки выпускников и используется для поступления в высшие учебные заведения. ЕГЭ охватывает различные предметы, такие как русский язык, математика, иностранные языки, обществознание, биология, физика и многие другие.

ДЗ – это сокращение от «домашнее задание». Это задачи или упражнения, которые ученикам или студентам дается для выполнения вне учебного заведения, обычно для закрепления материала, проверки понимания или развития навыков.

Приложение Б
(рекомендуемое)
Исходной код клиентской части

Листинг Б.1 – Модуль урока

```

<script>
import BlockProgressCourse from
"@/components/DefaultComponents/BlockComponents/BlockProgressCourse.vue";
import ModalSetDesiredBal from "../ModalWindows/ModalSetDesiredBal.vue";
import FeedbackComponent from "../ModalWindows/FeedbackComponent";
import FilesStoreComponent from
"@/components/DefaultComponents/BlockComponents/FilesStoreComponent.vue";
import BlockTask from
"@/components/DefaultComponents/BlockComponents/BlockTask.vue";
import ModalNextStepTestDrive from
"@/components/DefaultComponents/ModalWindows/ModalNextStepTestDrive.vue";
import Breadcrumbs from "@/components/DefaultComponents/Breadcrumbs";
import VuePlyr from '@components/VideoPlayer/PlyrPlayer.vue';
import YouTubeChatComponent from
"@/components/VideoPlayer/YouTubeChatComponent";
import InformationBlock from
"@/components/DefaultComponents/BlockComponents/InformationBlock.vue";
import WaitingLoading from
'@/components/DefaultComponents/WaitingLoading.vue';
import PlayerItems from
"../../DefaultComponents/BlockComponents/Lesson/PlayerItems";
import Prompt from "../Prompt/Prompt.vue";
import ContentBlock from
"../../DefaultComponents/BlockComponents/ContentBlock";
import BonusList from
"@/components/Course/ListLessons/Composables/BonusList";
import MaterialsBlock from
"@/components/Course/Lesson/Composables/MaterialsBlock";
import {send} from "../../fetchData";
import {getMoscowTime, getSizeFile, getTextNumberWord, pluralize,
breadcrumbsConstructor} from "../../defaultFunctions/defaultFunctions";
import {SvgModalNextStepTestDrive} from "@/svgStoreFiles";
import ModalTestDrivePrompt from
'@/components/DefaultComponents/ModalWindows/ModalTestDrivePrompt.vue';
import LevelProgress from
"@/components/Course/Lesson/Composables/LevelProgress";
import BonusModal from
"@/components/Course/Lesson/ModalWindows/BonusModal";
import {getBonuses, sendQueryCloseBonus} from
"@/components/Course/Methods/BonusModal";
import {databus} from "@/components/databus";

export default {
  name: "LessonComponent",

  data() {
    return {
      isGame: true, // если геймификация в уроке
      modalBonus: { // модальное окно с информацией по бонусу
        id: null,
        show: false,
        isAutoShow: false, // модальное окно отображено автоматически?
      },
      bonuses: [], // список бонусов
      levelScore: { // данные по баллам по уровню
        maxScore: null, // максимальный балл
      }
    }
  }
}

```

```

    passingScore: null, // проходной балл
    score: null, // набранный балл пользователя
    detailsProgress: [], // детализированный прогресс
    idHint: null, // id подсказки
    showHint: false, // Показать подсказку
  },
  dataProgressCourse: { // данные прогресса курса
    progress: 0, // процент прогресса
    steps: [
      {
        id: 0,
        text: 'Урок',
        isActive: true,
        route: null,
        isRecommendationStep: false
      },
      {
        id: 1,
        text: 'Домашнее задание',
        isActive: false,
        route: 'tdHomework',
        isRecommendationStep: false
      },
      {
        id: 2,
        text: 'Интенсив',
        isActive: false,
        route: 'tdIntensive',
        isRecommendationStep: false
      },
      /* {
        id: 3,
        text: 'ТА',
        isActive: false,
        route: 'tdTa',
        isRecommendationStep: false
      }, */
      {
        id: 4,
        text: 'Покупка курса',
        isActive: false,
        route: 'buyCourseTD',
        isRecommendationStep: false
      }
    ]
  },
  dataModalNextStep: { //данные модального окна с переходом на
    следующий шаг
    show: false,
    enable: false,
    svg: SvgModalNextStepTestDrive.image_1,
    svgMobile: SvgModalNextStepTestDrive.image_mobile_1,
    title: 'Молодец! Ты посмотрел урок, заполнил скрипт \n' +
      'и оценил занятие.',
    description: 'Пора приступать к домашнему заданию. Желаем удачи!
    😊',
  },

```

```

    textButton: 'Выполнить ДЗ'
  },

  dataPromptIsCheckLesson: { // данные подсказки
    show: false,
    showModal: false,
    enableModal: false,
    positionSquare: 'right: -7px; bottom: 40px; transform: rotateZ(-135deg);', // позиция стрелки
    positionSquareTablet: 'left: 24px; top: -7px; transform: rotateZ(135deg);', // позиция стрелки
    data: {
      index: 0,
      id: 0,
      countPrompt: 1,
      numberPrompt: 1,
      offset: null,
      image: '/lk/gk/img/image_prompt_lesson_3.png',
      title: '1. Переход к домашнему заданию',
      description: 'Выбирай пункт «Домашнее задание», чтобы перейти к следующему шагу, как только будешь готов!'
    },
    prompts: [ // все подсказки на странице
      {
        id: 0,
        number: 1,
        image: '/lk/gk/img/image_prompt_lesson_4.png',
        title: '1. Переход к домашнему заданию',
        offset: null,
        description: 'Выбирай пункт «Домашнее задание», чтобы перейти к следующему шагу, как только будешь готов!',
        positionSquare: 'right: -7px; bottom: 40px; transform: rotateZ(-135deg);', // позиция стрелки
        positionSquareTablet: 'left: 24px; top: -7px; transform: rotateZ(135deg);', // позиция стрелки
      }
    ]
  },

  dataPrompt: { // данные подсказки
    show: true,
    showModal: true,
    enableModal: false,
    positionSquare: 'left: -7px; bottom: 20px; transform: rotateZ(45deg);', // позиция стрелки
    positionSquareTablet: 'left: -7px; bottom: 20px; transform: rotateZ(45deg);', // позиция стрелки
    data: {
      index: 0,
      id: 0,
      countPrompt: 3,
      numberPrompt: 1,
      offset: null,
      image: '/lk/gk/img/Video(1).png',
      title: '1. Видеоролик с уроком',
      description: 'На полугодовом курсе занятия длятся 90 минут и проходят два раза в неделю в удобное для тебя время. В середине занятия

```

```

будет 5-минутный перерыв, чтобы у тебя была возможность немного
отдохнуть',
    },
    prompts: [ // все подсказки на странице
        {
            id: 0,
            number: 1,
            image: '/lk/gk/img/Video(1).png',
            title: '1. Видеоролик с уроком',
            offset: null,
            description: 'На полугодовом курсе занятия длятся 90 минут и
проходят два раза в неделю в удобное для тебя время. В середине занятия
будет 5-минутный перерыв, чтобы у тебя была возможность немного
отдохнуть',
            positionSquare: 'left: -7px; bottom: 20px; transform:
rotateZ(45deg);', // позиция стрелки
            positionSquareTablet: 'left: -7px; bottom: 20px; transform:
rotateZ(45deg);', // позиция стрелки
        },
        {
            id: 1,
            number: 2,
            image: '/lk/gk/img/image_prompt_2.png',
            title: '2. Оценка урока',
            offset: -650,
            description: 'Мы постоянно работаем над улучшением качества
наших курсов. С помощью обратной связи от наших учеников мы растем.
Растем вместе с ними. Поэтому после занятия мы просим ребят поделиться
насколько понятным и интересным было занятие, просто отправив
соответствующий их настроению эмодзи. Оцени занятия, чтобы завершить этот
шаг.',
            positionSquare: 'left: 24px; bottom: -7px; transform:
rotateZ(-45deg);', // позиция стрелки
            positionSquareTablet: 'left: 24px; bottom: -7px; transform:
rotateZ(-45deg);', // позиция стрелки
        },
        {
            id: 2,
            number: 3,
            image: '/lk/gk/img/image_prompt_lesson_3.png',
            title: '3. Скрипты к уроку',
            offset: -400,
            description: 'Для каждого занятия мы подготовили удобные
скрипты. Заполняя их во время занятия, ученики лучше запоминают материал
и сразу же получают готовый конспект по пройденной теме.',
            positionSquare: 'right: -7px; bottom: 40px; transform:
rotateZ(-135deg);', // позиция стрелки
            positionSquareTablet: 'left: 24px; bottom: -7px; transform:
rotateZ(-45deg);', // позиция стрелки
        }
    ]
},
globalIdLesson: undefined, // глобальный id урока
dataReceived: false, // данные от апи были получены
currentNameRoute: undefined, // наименование текущего маршрута
listNameRouteCurrentPage: [ // список страниц, на которых
используется данный компонент

```

```

    'gkGameLesson',
    'gkLesson',
    'pgkLesson',
    'ffLesson',
    'drainLesson',
    'tdLesson',
    'theoryLesson'
  ],
  lessonHasLoaded: true, // урок был загружен?
  homeworkIsOpen: true, // домашняя работа открыта?
  lessonData: { // данные урока
    id: 1,
    hasHomework: false, // есть ли ДЗ
    type: undefined, // тип урока / 1 - простое занятие / 2 - теория
    / 3 - экзамен
    isStrict: false, // строгий тип экзамена? только для типа 3
    (экзамен)
    isStart: false, // прохождение занятия начато?
    isSent: false, // отправлено на проверку?
    isChecked: true, // проверено?
    isCopy: false, // задание списано?
    isRevision: false, // отправлено на доработку?
    isDeadline: true, // занятие сдано в дедлайн?
    passingScoreIsScored: true, // проходной балл набран?
    passingScore: undefined, // проходной балл
    allPoints: undefined, // баллов всего
    receivedPoints: undefined, // получено баллов
    desiredBal: undefined, // желаемый балл
  },
  lessonMaterials: { // материалы к уроку
    files: [], // файлы
    links: [], // ссылки
  },
  feedbackData: { // данные для отзыва
    show: false, // отобразить модальное окно
    reviewLeft: false, // отзыв оставлен?
    isAnonymous: false, // отзыв оставлен анонимно?
    score: 5, // оставленный балл
    aspects: [], // массив тегов отзыва
    comment: 'Все было отлично! :)',
    answerTutor: 'Здравствуй!</br> Спасибо за высокую оценку занятия.
    Дальше будет интереснее :) ',
  },
  timeDeadlineString: '', // время дедлайна в текстовом формате
  isDeadlineExpired: false, // дедлайн прошел?
  titleLesson: '',
  breadcrumbs: [], // хлебные крошки для навигации

  player: { // данные для плеера
    key: 0,
    type: 'audio', // тип плеера / audio - аудиоконтент / video -
    видеоконтент
    indexSelectedPlayer: 0,
    audio: [],
    video: {
      preview: this.$store.getters.path + '/img/video/Preview.jpg',
      // url изображения для превью видео

```

```

        type: 'local', // тип видео / local - видео с хостинга /
youtube - видео с YouTube
        videoLocal: [],
        videoYouTube: {
            link: '', // ссылка на видео (хостинг youtube)
            showChat: false, // отображать youtube чат
            linkOnBroadcast: '<a href="https://ri-ege.ru/lk/2021/"
target="_blank">https://ri-ege.ru/lk/2021/</a>', // html контент
            содержащий ссылку на прямую трансляцию YouTube
        }
    },

    infoBeforeStartExam: {
        showModal: false,
        description: '',
        titleBlock: 'Информация о работе',
    },

    activeVideoTimecode: {
        type: String,
        default: undefined
    }
},

created() {
    this.currentNameRoute = this.$route.name;
    this.getData();
},

mounted() {

},

watch: {
    '$route'(after, before) {
        this.currentNameRoute = after.name;

        if (this.listNameRouteCurrentPage.includes(after.name))
this.getData();
    },
},

methods: {
    setTimeVideo: function (time) {
        this.$emit('update', time);
    },

    /** Привести unix к миллисекундам
     * @param timestamp - unix date
     */
    getSecUnixTimecode(timestamp) {
        let date = new Date(timestamp * 1000);
        let hours = date.getHours();
        let minutes = date.getMinutes();
        let sec = date.getSeconds();

```

```

    return (hours * 3600) + (minutes * 60) + sec
  },

  /**
   * Открыть модальное окно с переходом на следующий шаг
   */
  openModalNextStepTestDrive() {
    this.dataPrompt.show = false;
    this.dataModalNextStep.show = true;

    setTimeout(() => {
      this.dataModalNextStep.enable = true;
    })
  },

  /**
   * Открыть модальное окно
   */
  openModalDataModalNextStep() {
    if (this.$route.name !== 'tdLesson') return;

    this.dataModalNextStep.show = true;

    setTimeout(() => {
      this.dataModalNextStep.enable = true;
    }, 100)
  },

  closeDataPromptIsCheckLesson() {
    this.dataPromptIsCheckLesson.show = false;
  },

  /**
   * Закрыть модальное окно с переходом на следующий шаг
   */
  closeModalNextStepTestDrive() {
    if (!this.dataModalNextStep.show || !this.dataModalNextStep.enable)
return;

    this.dataModalNextStep.show = false;
    this.dataModalNextStep.enable = false;

    this.dataProgressCourse.steps[1].isRecommendationStep = true;

    this.dataPromptIsCheckLesson.show = true;
    this.dataPromptIsCheckLesson.showModal = true;
  },

  // Получение данных
  getData() {
    /** Возвращает тип урока / audio / video
     * @param {Number} type - тип плеера / 1 - видео / 2 - аудио
     * @returns тип урока / audio - аудиофайл / video - видеофайл
     */
    let getPlayerType = (type) => (type == 1) ? ('video') : ('audio');

```

```

/** Возвращает превью для видеозаписи
 * @param {Object} preview - файл превью
 * @returns ссылка на превью
 */
let getPlayerPreview = (preview) => {
  if (preview && preview.full_link) return preview.full_link;

  return '';
};

/** Привести unix к формату строки
 * @param timestamp - unix date
 * @returns {String} время тайм кода
 */
let formatUnixTime = (timestamp) => {
  let date = new Date(timestamp * 1000);
  let hours = date.getHours();
  let minutes = date.getMinutes();
  let sec = date.getSeconds();

  if (hours < 10) hours = "0" + hours
  if (minutes < 10) minutes = "0" + minutes
  if (sec < 10) sec = "0" + sec

  return `${hours}:${minutes}:${sec}`
};

/** Возвращает массив аудиозаписей / видеозаписей
 * @param {Array} files - список файлов аудио / видео
 * @returns массив аудио
 */
let getPlayerFiles = (files) => {
  return files.reduce((accumulator, item, i) => {
    if (!item.file) return accumulator

    let timeCodes = [];

    if (item.times_code) {
      timeCodes = item.times_code.map((timeCode, i) => {
        return {
          id: timeCode.id,
          description: timeCode.description,
          time: timeCode.time_code,
          timeString: formatUnixTime(timeCode.time_code),
        }
      })
    }

    return [
      ...accumulator,
      {
        id: item.id,
        fileId: (item.file) ? (item.file.id) : (--i),
        name: item.name,
        link: (item.file) ? (item.file.full_link) : (''),
        timeCodes: timeCodes,
      }
    ]
  })
};

```

```

    ]
  }, []);
};

/** Возвращает объект
 * @param {String} videoLink - ссылка на видео
 * @param {Boolean} isBroadcast - это прямая трансляция?
 * @param {Number} timeEndBroadcast - время окончания прямой
трансляции
 * @returns объект с данными для плеера с видео с ютуба
 */
let getPlayerYouTube = (videoLink, isBroadcast, timeEndBroadcast)
=> {
  let linkOnYouTube = '';
  let mskTimeSeconds = getMoscowTime().timeInSeconds;

  if (videoLink) {
    linkOnYouTube = videoLink.slice(3);
  }

  return {
    link: linkOnYouTube, // ссылка на видео
    showChat: timeEndBroadcast && (timeEndBroadcast >
mskTimeSeconds), // показать чат
    htmlLinkContent: `

```

```

    let getPlayer = (typePlayer, linkYouTube, isBroadcast,
timeEndBroadcast, preview, audioFiles, videoFiles) => {
        const videoYouTubeContent = getPlayerYouTube(linkYouTube,
isBroadcast, timeEndBroadcast);
        const videoLocal = getPlayerFiles(videoFiles);

        return { // данные для плеера
            key: 0,
            type: getPlayerType(typePlayer), // тип плеера / audio -
аудиоконтент / video - видеоконтент
            indexSelectedPlayer: 0, // index выбранного плеера
            audio: getPlayerFiles(audioFiles),
            video: {
                preview: getPlayerPreview(preview), // url изображения для
превью видео
                type: getPlayerVideoType(videoLocal.length,
videoYouTubeContent.link), // тип видео / local - видео с хостинга /
youtube - видео с YouTube
                videoLocal: videoLocal,
                videoYouTube: {
                    link: videoYouTubeContent.link, // ссылка на видео (хостинг
youtube)
                    showChat: videoYouTubeContent.showChat, // отображать
youtube чат
                    linkOnBroadcast: videoYouTubeContent.htmlLinkContent, //
html контент содержащий ссылку на прямую трансляцию YouTube
                }
            }
        };

    /** Получить тип урока
    * @param {Boolean} isDiagnostic - это диагностический урок?
    * @param {Number} type - тип урока // 1 - обычный / 3 -
диагностический
    * @returns тип урока
    * */
    let getTypeLesson = (isDiagnostic, type) => {
        if (isDiagnostic) return 3
        return 1
    };

    /** Получить хлебные крошки для ГК и для ПКГ
    * @param {String} titleCourse - наименование курса
    * @param {String} titleLesson - наименование урока
    * @param {String} routerCourseName - наименование для страницы со
списком списка уроков
    * @returns список с хлебными крошками
    * */
    let getBreadcrumbs = (titleCourse, titleLesson, routerCourseName)
=> {
        return breadcrumbsConstructor([
            {title: titleCourse, routeName: routerCourseName},
            {title: titleLesson, routeName: undefined},
        ]);
    };

```

```

/** Получить данные прогресса по уровню
 * @param {Object} levelProgress - данные прогресса уровня
 * @param {Boolean} showHint - показать подсказку
 * @param {Number} idHint - id подсказки
 * @returns {Object}
 * */
let getLevelScore = (levelProgress, showHint, idHint) => {
  let getListDetailsProgress = (details, userScore) => { //
получить список детализированного прогресса
    const pluralizeScoreWords = ['балл', 'балла', 'баллов'];
    let detailsProgress = [];
    if (details) detailsProgress = details.map((item, index) => {
      return {
        id: index,
        title: item.name,
        score: item.score,
        textScore: pluralize(item.score, pluralizeScoreWords),
      }
    });

    detailsProgress.push({
      id: detailsProgress.length,
      title: 'Всего набрано',
      score: userScore,
      textScore: pluralize(userScore, pluralizeScoreWords),
    });

    return detailsProgress;
  };

  return {
    maxScore: levelProgress.max_score, // максимальный балл
    passingScore: levelProgress.goal, // проходной балл
    score: levelProgress.score, // набранный балл пользователя
    detailsProgress:
getListDetailsProgress(levelProgress.detail_progress,
levelProgress.score),
    idHint: idHint, // id подсказки
    showHint: showHint, // Показать подсказку
  }
};

this.lessonMaterials.files = [];
this.lessonMaterials.links = [];
this.breadcrumbs = [];
this.feedbackData.aspects = [];
if (this.dataReceived) this.dataReceived = false;

// получить массив id подсказок
let getArrIdsTips = (tips) => {
  if (!tips || tips.length == 0) return [];
  return tips.reduce((accumulator, item) => {
    if (item.show) accumulator.push(item.id);
    return accumulator;
  }, []);
};

```

```

// установить данные / response - данные от api / type - тип
страницы ГК или ОИ (gk/ff)
let setData = (response, type) => {
  console.log('get_lesson_data', response);

  this.dataReceived = true;
  let titleCourse = 'Курс ';
  const declinationTitle =
this.$store.getters.getAllCourses[Number(this.$route.params.idCourse)]?.d
eclinationTitle;
  if (declinationTitle) titleCourse += declinationTitle;
  this.titleLesson = (response?.lesson?.name &&
(response?.lesson?.name.length > 0)) ? (response.lesson.name) : ('Без
наименования');

  switch (type) {
    case 'gk': // ГК
    case 'pgk': // ПГК
      this.homeworkIsOpen = response.lesson.is_open_hw;
      this.globalIdLesson = Number(this.$route.params.idLesson);
      this.isGame = response.level?.is_game ?? false;

      if (this.$route.name === 'theoryLesson') {
        this.breadcrumbs = breadcrumbsConstructor([
          {title: `Вся теория ${ declinationTitle }`, routeName:
'theoryCourse'},
          {title: this.titleLesson, routeName: undefined},
        ]);
      } else if (this.$route.name === 'drainLesson') { // Блок "Вся
теория"
        this.breadcrumbs = breadcrumbsConstructor([
          {title: titleCourse, routeName: 'drainCourse'},
          {title: this.titleLesson, routeName: undefined},
        ]);
      } else {
        const idProgressTip = 1;
        let idsTips = getArrIdsTips(response.lesson.tips);
        this.breadcrumbs =
this.getBreadcrumbs(response.level?.number, this.titleLesson);
        this.bonuses = getBonuses(response.level_progress?.bonus);
        this.levelScore = getLevelScore(response.level_progress,
idsTips.includes(idProgressTip), idProgressTip);
        this.openInfoModalNewBonus();
      }
      break;

    case 'ff': // ОИ
      this.homeworkIsOpen = true;
      this.globalIdLesson = response.lesson.global_id;
      this.breadcrumbs = breadcrumbsConstructor([
        {title: titleCourse, routeName: 'course'},
        {title: response.module.name, routeName: 'ffLessons'},
        {title: this.titleLesson, routeName: undefined},
      ]);
      break;

    case 'td':

```

```

        this.globalIdLesson = response.lesson.global_id;
        this.dataPrompt.show = !response.step_status;
        this.dataPrompt.showModal = !response.step_status;
        this.breadcrumbs = getBreadcrumbs(
            titleCourse,
            this.titleLesson,
            'TestDriveCourse'
        );
        break;
    }

    if (!response) return;

    this.lessonHasLoaded = response.lesson.extra_audio_records.length
    > 0 || response.lesson.extra_video_records.length > 0;
    if (!this.homeworkIsOpen && response.lesson.time_open_hw)
    this.timerOpenHomework(response.lesson.time_open_hw);
    this.timeDeadlineString = response.lesson.deadline_str;
    this.isDeadlineExpired = response.lesson.is_deadline_expired;

    this.lessonData.id = Number(this.$route.params.idLesson);
    this.lessonData.hasHomework = (response.lesson.is_hw ||
    response.lesson.is_set_hw) ?? false;
    this.lessonData.type =
    getTypeLesson(response.lesson.is_hw_diagnostic, response.lesson.type);
    this.lessonData.passingScore = response.lesson.score_passing;
    this.lessonData.allPoints = response.lesson.max_bal;
    this.lessonData.isStart = response.lesson.passed_data.is_start;
    this.lessonData.isSent = response.lesson.passed_data.is_sent;
    this.lessonData.isChecked =
    response.lesson.passed_data.is_response;
    this.lessonData.receivedPoints = response.lesson.passed_data.bal;
    this.lessonData.passingScoreIsScored =
    (this.lessonData.receivedPoints && (this.lessonData.receivedPoints >=
    this.lessonData.passingScore)) || !this.lessonData.passingScore;
    this.lessonData.isRevision =
    response.lesson.passed_data.is_revision;
    this.lessonData.isDeadline =
    response.lesson.passed_data.is_deadline;
    this.lessonData.isStrict = response.lesson.is_strict;
    this.lessonData.desiredBal = response.lesson.desired_bal ??
    undefined;

    if (response.lesson.materials?.files)
    response.lesson.materials.files.forEach((item, i) => {
        let textSize = getSizeFile(item.file.size);

        this.lessonMaterials.files.push({
            id: item.id,
            image: item.image?.full_link,
            title: (item.file.name && (item.file.name.length > 0)) ?
            (item.file.name) : ('Без наименования'),
            description: textSize,
            size: textSize,
            link: item.file.full_link,
        })
    });

```

```

else this.lessonMaterials.files = [];

if (response.lesson.materials?.links)
response.lesson.materials.links.forEach((item, i) => {
  this.lessonMaterials.links.push({
    id: item.id,
    image: item.image?.full_link,
    title: (item.title && (item.title.length > 0)) ? (item.title)
: ('Без наименования'),
    description: item.url,
    link: item.url,
  })
});
else this.lessonMaterials.links = [];

this.feedbackData.show = this.$route.name === 'tdLesson' ? false
: this.lessonData.hasHomework && response.lesson.is_show_feedback_modal;
this.feedbackData.reviewLeft = response.lesson.feedback.is_send;
this.feedbackData.score = (response.lesson.feedback.score > 0) ?
(response.lesson.feedback.score) : (undefined);
this.feedbackData.comment = response.lesson.feedback.text;
this.feedbackData.answerTutor =
(response.lesson.feedback.is_answer) ?
(response.lesson.feedback.text_answer) : ('');
this.feedbackData.isAnonymous = response.lesson.feedback.is_anon;

response.lesson.feedback.tags.forEach((item, i) => {
  this.feedbackData.aspects.push({
    id: item.id, // int - ID тега
    title: item.text, // string - текст тега
    svgDefault: item.icon, // File - SVG иконка тега
    svgSelected: item.icon_active, // File - SVG иконка тега
(активная)
    isSelect: item.is_select, // bool - отмечен ли тег учеником
    fromBal: item.from_bal, // Array(int) - массив баллов, при
которых отображается тег
  });
});

this.player = getPlayer(
  response.lesson.video_record_type,
  response.lesson.video_record_link,
  response.lesson.is_broadcast,
  response.lesson.time_end_broadcast,
  response.lesson.video_preview,
  response.lesson.extra_audio_records,
  response.lesson.extra_video_records
);

console.log('this.player', this.player);
};

switch (this.currentNameRoute) {
case 'gkGameLesson': // GK
  this.getGKData(true).then(response => setData(response, 'gk'))
  break;

```

```

    case 'gkLesson': // ГК
    case 'drainLesson':
        this.getGKData().then(response => setData(response, 'gk'))
        break;

    case 'pgkLesson': // ПГК
        this.getGKData().then(response => setData(response, 'pgk'))
        break;

    case 'ffLesson': // ОИ
        this.getFFData().then(response => setData(response, 'ff'))
        break;

    case 'tdLesson': // Пробный урок
        this.getTDDData().then(response => setData(response, 'td'))
        break;

    case 'theoryLesson': // Теоретический урок (ПГК)
        this.getGKData().then(response => setData(response, 'gk'))
        break;
    }
},

/** Получить список хлебных крошек
 * @param {Number} levelNumber - номер уровня
 * @param {String} titleLesson - заголовок урока
 * @returns {Array} список хлебных крошек
 * */
getBreadcrumbs(levelNumber, titleLesson) {
    /** Получить наименование родительского маршрута группы
     * @param {String} name - наименование текущего маршрута
     * @returns {String}
     * */
    let getParentRouteName = (name) => {
        switch (name) {
            case 'gkGameLesson':
                return 'gkLevelsMap';
        }

        return null;
    };

    let routeName = getParentRouteName(this.$route.name);

    return breadcrumbsConstructor([
        {title: 'Список уровней', routeName: 'gkLevelsMap'},
        {title: `${getTextNumberWord(levelNumber)} уровень`, routeName:
'gkLessons'},
        {title: titleLesson, routeName: undefined},
    ]);
},

/**
 * Метод получения данных для урока для ГК
 * @param {Boolean} isNew - отвечает за то, апи нового года или
старого используется
 * */

```

```

getGKData(isNew = false) {
  return new Promise((resolve, reject) => {
    send
      .post('groups.php', {
        module: isNew ? 'get_lesson_data_v2' : 'get_lesson_data',
        lesson_id: this.$route.params.idLesson,
      })
      .then(response => {
        resolve(response)
      })
      .catch(error => {
        console.log("error", error);
        console.log(error.response);

        reject(error.response);
      })
  })
},

/**
 * Метод получения данных для урока для ОИ
 */
getFFData() {
  return new Promise((resolve, reject) => {
    send
      .post('final_essay_lesson.php', {
        module: 'get_lesson_data',
        lesson_id: this.$route.params.idLesson,
      })
      .then(response => {
        resolve(response)
      })
      .catch(error => {
        console.log("error", error);
        console.log(error.response);

        reject(error.response);
      })
  })
},

getTDDData() {
  return new Promise((resolve, reject) => {
    send
      .post('test_drive.php', {
        module: 'get_lesson',
        course_id: this.$route.params.idCourse,
      })
      .then(response => {
        resolve(response)
      })
      .catch(error => {
        console.log("error", error);
        console.log(error.response);

        reject(error.response);
      })
  })
}

```

```

    })
  },

  // таймер, отсчитывающий время до открытия урока
  timerOpenHomework(timeOpen) {
    let mskSecondsTime = getMoscowTime().timeInSeconds;
    let timeLeft = timeOpen - mskSecondsTime; // кол-во оставшегося
    времени в секундах

    if (timeLeft > 0) {
      const th = this;
      let interval = 1000; // ms
      let expected = Date.now() + interval;
      setTimeout(step, interval);

      function step() {
        let dt = Date.now() - expected; // the drift (positive for
        overshooting)

        if (dt > interval) {
          th.timerOpenHomework();
        } else {
          timeLeft--;
          expected += interval;

          if (timeLeft > 0) setTimeout(step, Math.max(0, interval -
dt));
          else th.homeworkIsOpen = true;
        }
      }
    } else if (timeLeft <= 0) {
      this.homeworkIsOpen = true;
    }
  },

  // перейти на страницу с домашним заданием
  goToHomework() {
    if (!this.homeworkIsOpen) return false;

    switch (this.currentNameRoute) {
      case 'ffLesson':
        this.linkToHomework('ff');
        break;

      case 'pgkLesson':
        if ((this.lessonData.type === 3) && !this.lessonData.isStart) {
// ДЗ - это экзамен
          this.openModalStartExam();
        } else {
          this.linkToHomework('pgk');
        }
        break;

      case 'theoryLesson':
        this.linkToHomework('theory');
        break;
    }
  }
}

```

```

        default:
            if ((this.lessonData.type === 3) && !this.lessonData.isStart) {
// ДЗ - это экзамен
                this.openModalStartExam();
            } else {
                this.linkToHomework('gk');
            }
        }
    },

    // открыть модальное окно с отзывом
    openModalFeedback() {
        this.feedbackData.show = true;
    },

    // закрыть модальное окно с отзывом
    closeFeedbackModal() {
        if (this.feedbackData.show) this.feedbackData.show = false;
    },

    // открыть модальное окно начала экзамена
    openModalStartExam() {
        this.infoBeforeStartExam.showModal = true;
        let textDefault = 'Перед началом диагностической работы поставьте  

        желаемый балл, который хотите достичь на настоящем экзамене, это поможет  

        отслеживать прогресс.';

        if (this.lessonData.isStrict) {
            this.infoBeforeStartExam.description = 'Данная диагностическая  

            работа формата «Как на ЕГЭ», это значит, что после нажатия на кнопку  

            «Приступить к выполнению» вы не сможете остановить время, отведенное на  

            выполнение диагностической работы.</br></br>' + textDefault;
        } else {
            this.infoBeforeStartExam.description = 'Данная диагностическая  

            работа имеет формат «Не строгий», это значит, что после нажатия на кнопку  

            «Приступить к выполнению» вы сможете в любое время выйти из  

            диагностической работы и продолжить её решать через какое-то время, при  

            условии что, время диагностической работы не вышло.</br></br>' +
            textDefault;
        }
    },

    // закрыть модальное окно начала экзамена
    closeModalStartExam() {
        if (this.infoBeforeStartExam.showModal)
            this.infoBeforeStartExam.showModal = false;
    },

    // установить желаемый балл перед стартом экзамена
    setDesiredBal(desiredBal) {
        let data = {
            module: 'hw_start_exam',
            lesson_id: this.$route.params.idLesson,
            group_id: this.$route.params.idGroup,
            bal: desiredBal,
        };
    };

```

```

send
  .post('lesson.php', data)
  .then(response => {
    console.log('Успешно!');
    this.closeModalStartExam();

    switch (this.currentNameRoute) {
      case 'pgkLesson':
        this.linkToHomework('pgk');
        break;

      default:
        this.linkToHomework('gk');
    }
  })
  .catch(error => {
    console.log("error", error);
    console.log(error.response);
  })
},

// ссылка на урок
linkToHomework(type) {
  switch (type) {
    case 'gk':
      if (this.$route.name === 'gkGameLesson') {
        this.$router.push({
          name: 'gkGameHomework',
          params: {
            idCourse: this.$route.params.idCourse,
            idGroup: this.$route.params.idGroup,
            idLevel: this.$route.params.idLevel,
            idLesson: this.$route.params.idLesson,
          }
        });
      } else {
        this.$router.push({
          name: 'gkHomework',
          params: {
            idCourse: this.$route.params.idCourse,
            idGroup: this.$route.params.idGroup,
            idLesson: this.$route.params.idLesson,
          }
        });
      }
      break;

    case 'pgk':
      this.$router.push({
        name: 'pgkHomework',
        params: {
          idCourse: this.$route.params.idCourse,
          idGroup: this.$route.params.idGroup,
          idLesson: this.$route.params.idLesson,
        }
      });
      break;
  }
}

```

```

        case 'ff':
            this.$router.push({
                name: 'ffHomework',
                params: {}
            });
            break;

        case 'theory':
            this.$router.push({
                name: 'theoryHomework',
                params: {}
            });
            break;
    }
},

// обновить данные с отзывом
updateFeedbackData(params) {
    this.feedbackData.reviewLeft = params.reviewLeft;
    this.feedbackData.score = params.score;
    this.feedbackData.comment = params.comment;
    this.feedbackData.isAnonymous = params.isAnonymous;
},

/** Проверить необходимость и открыть модальное окно с информацией о
бонусах
* */
openInfoModalNewBonus() {
    for (let key in this.bonuses) {
        if (this.bonuses[key].show) {
            this.openBonusModal(this.bonuses[key].id, true);
            return;
        }
    }
},

/** Открыть модальное окно с бонусами
* @param {Number} id - id бонуса
* @param {Boolean} isAutoShow - это автоматический показ уведомления
* */
openBonusModal(id, isAutoShow = false) {
    this.modalBonus.id = id;
    this.modalBonus.show = true;
    this.modalBonus.isAutoShow = isAutoShow;
},

/** Закрыть модальное окно с бонусами
* */
closeBonusModal() {
    if (this.modalBonus.isAutoShow)
sendQueryCloseBonus(this.$route.params.idGroup,
this.$route.params.idLevel, this.bonuses);

    this.modalBonus.id = null;
    this.modalBonus.isAutoShow = false;
    this.modalBonus.show = false;

```

```

    },

    /** Закрыть подсказку по прогрессу
    * */
    closeLevelScoreHint() {
        this.levelScore.showHint = false;
        this.sendDataCloseHint(this.levelScore.idHint);
    },

    /** Отправить данные о закрытии модального окна подсказки
    * @param {Number} id - id модального окна с подсказкой
    * @returns {Promise} ответ от апи
    * */
    sendDataCloseHint(id) {
        return new Promise((resolve, reject) => {
            send
                .post('user.php', {
                    module: 'close_tip',
                    course_id: this.$route.params.idCourse,
                    type_bonus: id,
                })
                .then((response) => {
                    resolve(response);
                })
                .catch((error) => {
                    reject(error);
                })
        });
    },
},

computed: {
    /** Получить ссылку на видео для плеера
    * @returns ссылка на видео
    * */
    getVideoLink() {
        switch (this.player.video.type) {
            case 'local':
                if
                    (this.player.video.videoLocal[this.player.indexSelectedPlayer]) return
                    this.player.video.videoLocal[this.player.indexSelectedPlayer].link
                    break;

            case 'youtube':
                return this.player.video.videoYouTube.link
            }

            return '';
        },

        /** Проверить допустимость отображения блоков геймификации
        * @returns {Boolean} отображать ли блоки геймификации
        * */
        showGamificationElements() {
            const listPagesDisabled = ['tdLesson', 'ffLesson', 'drainLesson'];
            return !listPagesDisabled.includes(this.currentNameRoute) &&
            this.isGame;
        }
    }
}

```

```

    },

    /** Проверить допустимость отображения таймкодов для данного видео
    * */
    checkVisibleTimecodes() {
        return this.lessonHasLoaded && (this.player.type === 'video') &&
        (this.player.video.type === 'local') &&
        (this.player.video.videoLocal.length > 0) &&
        (this.player.video.videoLocal[this.player.indexSelectedPlayer].timeCodes.
        length > 0)
    },
},

components: {
    FeedbackComponent,
    VuePlyr,
    YouTubeChatComponent,
    FilesStoreComponent,
    BlockTask,
    Breadcrumbs,
    ModalSetDesiredBal,
    InformationBlock,
    WaitingLoading,
    PlayerItems,
    ContentBlock,
    Prompt,
    ModalNextStepTestDrive,
    BlockProgressCourse,
    ModalTestDrivePrompt,
    LevelProgress,
    BonusList,
    BonusModal,
    MaterialsBlock,
}
}
</script>

```

Листинг Б.2 – Модуль домашнего задания

```

<script>
import VueAwesomeSwiper from 'vue-awesome-swiper'
import {Swiper, SwiperSlide, directive} from 'vue-awesome-swiper'
import 'swiper/css/swiper.css'

import WorkDeadlineTimer from
'../../DefaultComponents/Timer/WorkDeadlineTimer.vue';
import ExamIsCheckedBlock from "../BlockElements/ExamIsCheckedBlock.vue";
import InformationBlock from "../BlockElements/InformationBlock";
import Breadcrumbs from "@components/DefaultComponents/Breadcrumbs";
import ModalNotification from
'@components/DefaultComponents/ModalWindows/ModalNotification';
import SlideUpDown from 'vue-slide-up-down';
import DefaultSlider from '@components/Slider/DefaultSlider.vue';
import HighlightingImageComponent from
'@components/DefaultComponents/HighlightingImageComponent.vue';
import modalInfoBlock from
'@components/DefaultComponents/modalInfoBlock.vue';

```

```

import LoadingComponent from
'@/components/DefaultComponents/ModalWindows/LoadingComponent.vue';
import WaitingLoading from
'@/components/DefaultComponents/WaitingLoading.vue';
import FilesComponent from
"@/components/DefaultComponents/BlockComponents/FilesHeader/FilesComponen
t";
import {send} from "../../../fetchData";
import {databus} from "../../../databus";
import {
  checkClickOutside,
  getMoscowTime,
  getTextNumberWord,
  reverseHtmlEncoding
} from "../../../defaultFunctions/defaultFunctions";
import AddFileWithPasteFunction from
"@/components/DefaultComponents/AddFileWithPasteFunction";

let textVersion = require("textversionjs");

let intervalSendAnswer;

export default {
  name: "HomeworkComponent",

  data() {
    return {
      listNameRouteCurrentPage: ['gkHomeworkPitStop', 'ffHomework',
'gkHomework', 'gkGameHomework', 'gkDiagnostics', 'gkGameDiagnostics',
'pgkHomework', 'pgkDiagnostics', 'gkHomeworkPractice',
'gkDiagnosticsPractice', 'theoryHomework'], // список страниц, на которых
используется данный компонент
      currentSavingIndex: undefined, // текущий сохраняемый пункт (index)
применяется при
      showModalErrorSaving: false, // показать модальное окно с
информацией об ошибке сохранения
      textModalErrorSaving: 'Сохранить ответ не удалось, попробуйте еще
раз, либо обратитесь к <a href="https://vk.com/im?sel=381434758"
target="_blank">техническому специалисту</a>',
      dataReceived: false, // данные получены?
      showModalWaitingQuery: false, // отображать модальное окно ожидания
окончания запроса к серверу
      textModalWaitingQuery: 'Пожалуйста, подождите, выполняется отправка
ответа',
      showModalExpiredPassing: false, // показать модальное окно "Дедлайн
просрочен"
      showModalExplanation: false, // показать модальное окно с пояснением
к пройденному тесту (только для экзамена)
      idGroup: undefined,
      showDeadline: false, // отображать дедлайн?
      titlePage: '',
      showTimer: false, // для стр. экзамена показать таймер?
      isDiagnostics: false, // является ли урок диагностическим?
      routerName: undefined,
      activeHighlightingImage: 0, // index активного слайда
      contentHighlightingImage: [], // массив слайдов для функционала
выделения по фото

```

```

    showHighlightingImage: false, // показать модальное окно выделение
фрагментов на фото
    visibleTaskBlock: false, // отобразить весь контент

    revisionModal: { // модальное окно "Доработать ответ"
        show: false,
        index: undefined,
    },

    swiperOption: {
        slidesPerView: 'auto',
        spaceBetween: 12,
        pagination: {
            clickable: true
        }
    },

    modalEmptyTask: { // модальное окно для информирования о не
пройденных заданиях
        show: false,
        title: 'Важная информация',
        text: '',
        titleButton: 'Отправить на проверку',
    },

    userIsBlocked: false,
    isMobileDevise: window.innerWidth <= 1160,

    screenWidth: window.innerWidth,
    screenHeight: window.innerHeight,

    isNotStrictTimer: true, // это не строгий таймер?
    testingLeftTime: 0, // время на выполнение работы
    testingStartTime: 0, // время начала выполнения работы

    isClickOnDeleteBlock: false, // для загруженных изображений
пользователем, клик на кнопку удалить
    sliderType: undefined, // параметры слайдера / 'editImages'
    sliderShow: false, // показать галерею img
    activeImg: 0, // клик по фото - передача в слайдер позиции активной
фотографии
    activePoint: undefined, // номер активного пункта
    sliderImages: [], // массив фото, для передачи в слайдер
    indexActiveImageSlider: undefined, // index пункта дз для слайдера
на мобильных устр-вах

    breadcrumbs: [],
    isSend: false, // дз сдано?
    deadlineOverdue: false, // дедлайн дз просрочен?
    deadlineHomework: '', // дедлайн ДЗ // 25 сентября
(понедельник) 00:00 МСК
    durationSlideDown: 0,
    durationMistakeSlideDown: 500, // время анимации открытия
модального окна с ошибкой дз
    timeDeadline: null, // время дедлайна числом

    notification: { // модальное окно уведомлений

```

```

        show: false,
        text: 'Ответ отправлен на проверку',
        type: 'successfully',
        time: 6
    },

    download: { // прикрепление файлов к дз
        showDownloadFilesModal: false,
        activeElemIndex: undefined, // индекс активного элемента
    },

    visibilityResultHomework: false, // отобразить блок с результатами
ДЗ
    resultHomework: [ // результаты работы
        /* {
            title: 'Максимальный балл',
            titleTablet: 'Максимальный балл',
            meaning: '68',
            meaningShort: '68'
        },
        {
            title: 'Проходной балл',
            titleTablet: 'Проходной балл',
            meaning: '22',
            meaningShort: '22'
        }
        */
    ],

    data: [], // данные дз
    isChecked: false, // экзамен проверен?
    isRevision: false, // Отправлено на доработку? Принимает true, если
    хотя бы одно задание отправлено на доработку
    pageHasBeenResized: false, // у страницы был изменен размер
    typePlatform: '', // тип платформы / gk / pgk
    },

    created() {
        this.routerName = this.$router.history.current.name;

        if (this.listNameRouteCurrentPage.includes(this.routerName))
        this.getData(); // получить данные дз
        window.addEventListener('resize', this.onResize);
    },

    mounted() {

    },

    destroyed() {
        window.removeEventListener('resize', this.onResize)
    },

    watch: {
        '$route'(after, before) {
            this.routerName = this.$router.history.current.name;

```

```

        this.onChangeRouter();
        if (this.$route.name === 'lessonHomework') {
            this.addAnimationSlide();
        }

        if (this.listNameRouteCurrentPage.includes(this.routerName))
this.getData(); // получить данные дз
    },
    },

    methods: {
        pastFile(e, i){
            let item = Array.from(e.clipboardData.items).find(x =>
/^image\/.test(x.type));
            if (!item) return;

            this.download.showDownloadFilesModal = true;
            this.download.activeElemIndex = i;

            let blob = item.getAsFile();
            setTimeout(() => {
                databus.$emit('uploadPasteFile', {
                    event: e,
                    files: [blob]
                })
            }, 1)
        },
        // получить данные дз
        getData() {
            this.typePlatform = this.$route.params.typePlatform;

            // удаление тегов в тексте
            let removeTagsText = (textWithTags) => {
                let text = textWithTags.replaceAll("<br>",
"\n").replaceAll("<br>", "\n");

                return text.replace(/<(.|\n)*?>/g, '')
            };

            /** Получить хлебные крошки для ГК и для ПГК
            * @param {Boolean} isDiagnostics - является ли урок
диагностическим
            * @param {String} titleCourse - наименование курса
            * @param {String} titleLesson - наименование урока
            * @param {String} routerCourseName - наименование для страницы со
списком списка уроков
            * @param {String} routerLessonName - наименование для страницы со
списком уроков
            * @returns список с хлебными крошками
            * */
            let getBreadcrumbs = (isDiagnostics, titleCourse, titleLesson,
routerCourseName, routerLessonName) => {
                const routeCourse = {
                    id: 1,
                    title: 'Курс ' + titleCourse,
                    link: {
                        name: routerCourseName,

```

```

        params: {}
      }
    };

    const routeLesson = {
      id: 2,
      title: titleLesson,
      link: {
        name: routerLessonName,
        params: {}
      }
    };

    const routeWork = {
      id: 3,
      title: (this.isDiagnostics) ? ('Диагностическая работа') :
('Домашнее задание'),
      link: {
        name: undefined,
        params: {}
      }
    };

    if (isDiagnostics) return [routeCourse, routeWork];
    return [routeCourse, routeLesson, routeWork];
  };

  /** Получить хлебные крошки для теор. занятий
   * @param {String} titleCourse - наименование курса
   * @param {String} titleLesson - наименование урока
   * @returns список с хлебными крошками
   * */
  let getTheoryBreadcrumbs = (titleCourse, titleLesson) => {
    return this.breadcrumbsConstructor([
      {title: `Вся теория ${ titleCourse }`, routeName:
'theoryCourse'},
      {title: titleLesson, routeName: 'theoryLesson'},
      {title: 'Домашнее задание', routeName: undefined},
    ]);
  };

  if (this.dataReceived) this.dataReceived = false;
  if (this.visibleTaskBlock) this.visibleTaskBlock = false;

  this.visibilityResultHomework = false;
  this.data = [];

  let data = {
    module: 'hw_get_points',
    group_id: this.$route.params.idGroup,
    lesson_id: this.$route.params.idLesson,
  };

  console.log('hw_get_points');
  console.log(data)

  send

```

```

.post('lesson.php', data)
.then((response) => {
  console.log('response hw_get_points');
  console.log(response);

  let dataInfo = null;
  if (!this.dataReceived) this.dataReceived = true;
  this.userIsBlocked = response.is_blocked; // юзер заблокирован?
  let courseName =
this.$store.getters.getAllCourses[Number(this.$route.params.idCourse)]?.d
eclinationTitle ?? '';
  let mskSecondsTime = getMoscowTime().timeInSeconds;

  switch (this.$route.name) {
    case 'theoryHomework':
      this.isDiagnostics = response.gk_info.is_hw_diagnostic;
      dataInfo = response.gk_info;
      this.timeDeadline = dataInfo.deadline;
      this.breadcrumbs = getTheoryBreadcrumbs(courseName,
response.lesson.name);
      break;

    case 'gkGameHomework':
    case 'gkHomework': // если это дз / диагностическое задание в
уроке
    case 'pgkHomework': // если это дз / диагностическое задание
в уроке
      this.isDiagnostics = response.gk_info.is_hw_diagnostic;
      dataInfo = response.gk_info;
      this.timeDeadline = dataInfo.deadline;

      if (this.$route.name === 'gkGameHomework') {
        this.breadcrumbs =
this.getBreadcrumbs(response.level.number, response.lesson.name,
this.isDiagnostics);
      } else if (this.$route.name === 'gkHomework') {
        this.breadcrumbs = getBreadcrumbs(
          this.isDiagnostics,
          courseName,
          response.lesson.name,
          'course',
          'gkLesson'
        );
      } else {
        this.breadcrumbs = getBreadcrumbs(
          this.isDiagnostics,
          courseName,
          response.lesson.name,
          'pgkCourse',
          'pgkLesson'
        );
      }
      break;

    case 'gkGameDiagnostics':
    case 'gkDiagnostics': // если это диагностика
    case 'pgkDiagnostics':

```

```

        case 'gkDiagnosticsPractice':
            this.isDiagnostics = true;
            dataInfo = response.gk_info;
            this.timeDeadline = dataInfo.deadline;

            if (this.$route.name === 'gkGameHomework' ||
this.$route.name === 'gkGameDiagnostics') {
                this.breadcrumbs =
this.getBreadcrumbs(response.level.number, response.lesson.name,
this.isDiagnostics);
            } else if (this.$route.name === 'gkDiagnostics') {
                this.breadcrumbs = getBreadcrumbs(
                    this.isDiagnostics,
                    courseName,
                    null,
                    'course',
                    null
                );
            } else if ( this.$route.name === 'gkDiagnosticsPractice' )
{
                this.breadcrumbs = [ // хлебные крошки
                    {
                        id: 1,
                        title: 'Практика ' + courseName,
                        link: {
                            name: 'PracticeComponent',
                            params: {}
                        }
                    },
                    {
                        id: 2,
                        title: response.lesson.name,
                        link: {
                            name: '',
                            params: {}
                        }
                    }
                ],
            } else {
                this.breadcrumbs = getBreadcrumbs(
                    this.isDiagnostics,
                    courseName,
                    null,
                    'pgkCourse',
                    null
                );
            }
            console.log('test breadcrumbs', this.breadcrumbs);

            break;

        case 'ffHomework': // если это ОИ
            this.isDiagnostics = false;
            dataInfo = response.fe_info;
            this.timeDeadline = dataInfo.time_end_send;

```

```

    if (mskSecondsTime >= this.timeDeadline) this.userIsBlocked
= true;

    this.breadcrumbs = [ // хлебные крошки
    {
        id: 1,
        title: 'Курс ' + courseName,
        link: {
            name: 'course',
            params: {}
        }
    },
    {
        id: 2,
        title: dataInfo.module.name,
        link: {
            name: 'ffLessons',
            params: {}
        }
    },
    {
        id: 3,
        title: response.lesson.name,
        link: {
            name: 'ffLesson',
            params: {}
        }
    },
    {
        id: 4,
        title: 'Домашнее задание',
        link: {
            name: undefined,
            params: {}
        }
    }
    ];
    break;

case 'gkHomeworkPractice':
    dataInfo = response.gk_info;

    this.breadcrumbs = [ // хлебные крошки
    {
        id: 1,
        title: 'Практика ' + courseName,
        link: {
            name: 'PracticeComponent',
            params: {}
        }
    },
    {
        id: 2,
        title: response.lesson.name,
        link: {
            name: '',
            params: {}
        }
    }
    ];

```

```

        }
    },
];
break;
case 'gkHomeworkPitStop':
    dataInfo = response.gk_info;

    this.breadcrumbs = [ // хлебные крошки
        {
            id: 1,
            title: 'Пит-стоп ' + courseName,
            link: {
                name: 'pitStopListTask',
                params: {}
            }
        },
        {
            id: 2,
            title: response.lesson?.name,
            link: {
                name: '',
                params: {}
            }
        }
    ],
];
break;
}

if (dataInfo?.pass?.is_response) {
    this.isChecked = true;
    if (dataInfo.max_bal && (dataInfo.max_bal > 0))
this.visibilityResultHomework = true;

    if (this.isDiagnostics) {

        let percentDesiredBal =
Math.round(Number(response.exam_score.result_bal) /
Number(response.exam_score.desired_bal) * 100); // желаемый балл в
процентах

        if (!percentDesiredBal) percentDesiredBal = 0;

        this.resultHomework = [
            {
                title: 'Максимальный балл',
                titleTablet: 'Максимальный балл',
                meaning: dataInfo.max_bal,
                meaningShort: dataInfo.max_bal
            },
            {
                title: 'Проходной балл',
                titleTablet: 'Проходной балл',
                meaning: dataInfo.score_passing,
                meaningShort: dataInfo.score_passing
            },
            {
                title: 'Полученный балл',

```

```

        titleTablet: 'Полученный балл',
        meaning: response.exam_score.test_bal,
        meaningShort: response.exam_score.test_bal
    },
    {
        title: 'Итоговый балл',
        titleTablet: 'Итоговый балл',
        meaning: response.exam_score.result_bal,
        meaningShort: response.exam_score.result_bal
    },
    {
        title: 'Желаемый балл',
        titleTablet: 'Желаемый балл',
        meaning: response.exam_score.desired_bal,
        meaningShort: response.exam_score.desired_bal
    },
    {
        title: 'Процент от желаемого балла',
        titleTablet: '% от желаемого балла',
        meaning: response.exam_score.result_bal + '/' +
response.exam_score.desired_bal + ' (' + percentDesiredBal + '%)',
        meaningShort: percentDesiredBal + '%'
    },
    ]
} else {
    this.resultHomework = [
        {
            title: 'Максимальный балл',
            titleTablet: 'Максимальный балл',
            meaning: dataInfo.max_bal,
            meaningShort: dataInfo.max_bal
        },
        {
            title: 'Проходной балл',
            titleTablet: 'Проходной балл',
            meaning: dataInfo.score_passing,
            meaningShort: dataInfo.score_passing
        },
        {
            title: 'Полученный балл',
            titleTablet: 'Полученный балл',
            meaning: response.exam_score.test_bal,
            meaningShort: response.exam_score.test_bal
        }
    ]
}
}
}

```

```

this.isSend = dataInfo?.pass?.is_sent;

```

```

if (dataInfo.timer) { // если есть таймер у данного экзамена
    this.showTimer = dataInfo.timer.is_active;
    this.isNotStrictTimer = !dataInfo.timer.is_strict;

```

```

    if (this.isNotStrictTimer) { // не строгий таймер
        this.testingLeftTime = dataInfo.timer.time;
    } else { // строгий таймер

```

```

        this.testingLeftTime = dataInfo.timer.time;
        this.testingStartTime = dataInfo.timer.timestamp_stop -
dataInfo.timer.time;
    }
}

this.idGroup = dataInfo.group_id;
this.showDeadline = (dataInfo.deadline > 0) &&
(!this.isDiagnostics || (this.isDiagnostics && this.isNotStrictTimer));
this.titlePage = response.lesson.name;
this.isRevision = dataInfo.pass && dataInfo.pass.is_revision;
// Отправлено на доработку?
this.deadlineHomework = response.lesson.deadline_str;

let deadlineStr, deadline;

if (mskSecondsTime >= response.lesson.deadline) {
    this.deadlineOverdue = true;
}

let countPointSubtask = 0; // кол-во пунктов подзадач
let countAnswersOnPointSubtask = 0; // кол-во выполненных
(сданных) пунктов подзадач
let allFiles, allAttachedFiles, numberPoint = 0;

response.hw_points.sort((prev, next) => prev.number -
next.number);
response.hw_points.forEach((item) => {
    let mistakes = [],
        mistakesByTypes = [],
        mistakesAddTypes = [],
        criteria = [],
        criteriaScoreReceived = 0,
        criteriaScoreAll = 0,
        countMistakes = 0,
        answerHTML = '',
        allRightFiles = null,
        studentExtends = null,
        answerText = '';
    let notesFiles = {files: [], images: []},
attachedFilesComment = {files: [], images: []},
        attachedFilesRightAnswer = {files: [], images: []};

    switch (item.type) {
        case 1: // текстовый пункт
            allFiles = this.getStructFiles(item.extends);

            this.data.push({
                id: item.id,
                type: 3, // 1 - блок дз с возможностью ответа / 2 -
блок с тестами / 3 - текстовый блок / 4 - смартбук
                typeAPI: item.type,
                showAll: false, // показать весь текст
                showButtonUnfolding: false, // показать кнопку
                "показать все"
                title: (item.header && (item.header.length > 0)) ?
(item.header) : ('Важная информация'),

```

```

        text: item.text,
        textWithoutTags: item.text.replace(/<(.|\n)*?>/g, ''),
        attachedFiles: allFiles.files,
        attachedImages: allFiles.images,
    });

    break;

case 2: // тестовый пункт
    numberPoint++;
    allFiles = this.getStructFiles(item.extends);
    let scoreReceived = 0;
    let scoreAll = 0;
    let countPassedTest = 0;

    let tests = [];
    item.tests.forEach((test, j) => {
        let textAverageTime = '';
        if (test.middle_time) {
            let averageMinutes = Math.floor(test.middle_time /
60);

            let averageSeconds = test.middle_time % 60;

            if (averageMinutes > 0) {
                textAverageTime = averageMinutes + ' мин';
            }

            if ((averageMinutes > 0) && (averageSeconds > 0)) {
                textAverageTime += ' ';
            }

            if (averageSeconds > 0) {
                textAverageTime += averageSeconds + ' сек';
            }
        }

        if (test.is_passed) countPassedTest++;

        tests.push({
            id: test.id,
            title: test.name,
            isChecked: test.is_passed, // тест пройден
            score: {
                max: (test.score_max) ? (test.score_max) : (0), //
максимальный балл
                received: (test.score) ? (test.score) : (0) //
полученный балл
            },
            averageTime: textAverageTime,
        });

        if (test.is_passed) { // тест пройден
            scoreReceived += test.score;
            scoreAll += test.score_max;
        }
    });

```

```

        let scoreText = '';
        if (scoreAll > 0) {
            scoreText = scoreReceived + ' ' +
this.getWordDeclension(scoreReceived, 'балл') + ' из ' + scoreAll
        }

        deadlineStr = response.lesson.deadline_str2;
        deadline = response.lesson.deadline;

        if (item.is_deadline) {
            deadlineStr = item.deadline_str;
            deadline = item.deadline;
        }

        tests.sort((prev, next) =>
prev.title.toLowerCase().localeCompare(next.title.toLowerCase()));

        this.data.push({
            id: item.id,
            type: 2, // 1 - блок дз с возможностью ответа / 2 -
блок с тестами / 3 - текстовый блок / 4 - смартбук
            typeAPI: item.type,
            show: false, // развернуть / скрыть блок с ДЗ //
тест пройден
            isChecked: (countPassedTest === tests.length) ? (true)
: (false), // все ли тесты пройдены
            title: (item.header && (item.header.length > 0)) ?
(item.header) : ('Тестирование'),
            number: numberPoint,
            scoreText: scoreText, // кол-во баллов за тесты всего
            deadline: {
                time: (item.deadline === null) ? (null) : (0), //
время оставшееся до дедлайна в секундах (23:00:00)
                timeString: '', // время оставшееся до дедлайна -
текстовое поле (23:00:00)
                timeFull: deadlineStr,
                timeDeadline: deadline, // время дедлайна в секундах
(15.10.21)
            },
            text: item.text,
            textWithoutTags: item.text.replace(/<(.|\n)*?>/g, ''),
            attachedFiles: allFiles.files,
            attachedImages: allFiles.images,
            tests: tests,
        });

        break;

    case 3: // пункт подзадачи
        numberPoint++;
        countPointSubtask++;

        if (item.answer.is_send) {
            countAnswersOnPointSubtask++;
        }

        allAttachedFiles = this.getStructFiles(item.extends);

```

```

deadlineStr = response.lesson.deadline_str2;
deadline = response.lesson.deadline;

if (item.is_deadline) {
  deadlineStr = item.deadline_str;
  deadline = item.deadline;
}

allRightFiles =
this.getStructFiles(item.extends_correct);

/*
  item.answer.extends.forEach((file, j) => {
    studentExtends.push({
      id: file.id,
      content: {
        title: file.file.name,
        size: file.file.size,
        link: file.file.full_link,
      }
    })
  }); */

item.notes.forEach((note, j) => {
  notesFiles = this.getStructFiles(note.extends);

  mistakes.push({
    id: note.local_id,
    idServer: note.id,
    show: false, // открыть окно с ошибкой
    title: note.type_description,
    comment: note.message.replaceAll(/\n\r/g, '<br/>'),
    images: notesFiles.images,
    files: notesFiles.files,
  });

  if (!mistakesAddTypes.includes(note.type)) {
    mistakesAddTypes.push(note.type);
    mistakesByTypes.push({
      id: note.type,
      title: note.type_description,
      count: 1
    })

    if (note.type !== 1) countMistakes++;
  } else {
    for (let k = 0; k < mistakesByTypes.length; k++) {
      if (mistakesByTypes[k].id === note.type) {
        mistakesByTypes[k].count += 1;
        if (note.type !== 1) countMistakes++;
        break;
      }
    }
  }
});

```

```

        if (item.is_checked) {
            studentExtends =
this.getStructFiles(item.answer.extends, 'isHighlightingContent');

            if (item.check_data.feedback.extends) {
                attachedFilesComment =
this.getStructFiles(item.check_data.feedback.extends);
            } else {
                attachedFilesComment = {
                    files: [],
                    images: [],
                }
            }

            studentExtends.images.forEach((img, i) => {
                img.mistake.forEach((item, j) => {
                    let indexMistakesByTypes = undefined;
                    for (let i = 0; i < mistakesByTypes.length; i++) {
                        if (item.typeMistake === mistakesByTypes[i].id) {
                            indexMistakesByTypes = i;
                            break;
                        }
                    }

                    if (indexMistakesByTypes === undefined) {
                        mistakesByTypes.push({
                            id: item.typeMistake,
                            title: item.title,
                            count: 1
                        });
                    } else {
                        mistakesByTypes[indexMistakesByTypes].count += 1;
                    }

                    if (item.typeMistake !== 1) countMistakes++;
                })
            });

            item.criteria.forEach((elem, j) => {
                criteriaScoreReceived += Number(elem.bal);
                criteriaScoreAll += Number(elem.max_bal);

                criteria.push({
                    id: elem.id,
                    prefix: elem.prefix,
                    number: elem.number,
                    text: elem.text,
                    score: {
                        max: elem.max_bal, // максимальный балл
                        received: elem.bal // полученный балл
                    }
                })
            })
        } else {
            studentExtends =
this.getStructFiles(item.answer.extends);

```

```

    }

    answerText =
reverseHtmlEncoding(removeTagsText(item.answer.text));
    answerHTML = item.answer.text.replaceAll(' ',
'&nbsp;&nbsp; ').replace(/\r?\n/g, '<br>');

    this.data.push({
        id: item.id,
        savingStatus: 'none', // none - нет изменений / error -
ошибка / successfully - успешно / inProgress - в процессе
        savingBlockOpened: false, // блок с сохранением
открыть? / только для типов error и successfully
        type: 1, // 1 - блок дз с возможностью ответа / 2 -
блок с тестами / 3 - текстовый блок / 4 - смартфон
        typeAPI: item.type,
        isRevision: item.answer.is_revision, // Отправлено на
доработку?
        isAccessPaste: item.is_access_paste, // разрешить
вставку ответа
        responseRequired: item.is_answer, // требуется ли ответ
        show: true, // развернуть / скрыть блок с ДЗ
        isExtends: item.is_extends, // можно ли прикреплять
файлы // item.is_extends
        isSend: item.answer.is_send, // данные отправлены?
        isChecked: item.is_checked, // данные проверены?
        isStartCheck: ((item.is_start_check === undefined) ||
(item.is_start_check === null)) ? (true) : (item.is_start_check), //
проверка данного пункта начата?
        title: (item.header && (item.header.length > 0)) ?
(item.header) : ('Задание'),
        answer: answerText,
        answerOld: answerText, // для отслеживания вставки
текста
        savedAnswer: answerText, // сохраненный ответ
        answerHTML: answerHTML, // answer преобразованный в
html
        number: numberPoint,
        queryCounter: { // счетчик запросов
            all: 0,
            done: 0,
        },
        deadline: {
            time: (item.deadline === null) ? (null) : (0), //
время оставшееся до дедлайна в секундах (23:00:00)
            timeString: '', // время оставшееся до дедлайна -
текстовое поле (23:00:00)
            timeFull: (item.is_answer) ? (deadlineStr) : (''),
            timeDeadline: (item.is_answer) ? (deadline) : (''),
            // время дедлайна в секундах (15.10.21)
        },
        text: item.text,
        textWithoutTags: item.text.replace(/<(.|\n)*?>/g, ''),
        attachedFiles: allAttachedFiles.files,
        attachedImages: allAttachedFiles.images,
        downloadStudentFiles: studentExtends.files,
        downloadStudentImages: studentExtends.images,
    });

```

```

        // данные, приходящие после проверки пункта дз
        scoreText: (item.is_checked) ? (item.check_data.bal + '
' + this.getWordDeclension(Number(item.check_data.bal), 'балл') + ' из '
+ item.max_bal) : (''),
        textChecked: item.answer.text.replaceAll("\n",
"</br>"),
        activeMistake: undefined, // id открытой ошибки
        mistake: mistakes,
        mistakesByTypes: mistakesByTypes,
        countMistakes: countMistakes,
        criteria: criteria,
        comments: { // Комментарий к ответу
            isShow: (item.is_checked &&
((item.check_data.feedback.text && (item.check_data.feedback.text.length
> 0)) || (item.check_data.feedback.extends &&
(item.check_data.feedback.extends.length > 0))))),
            showAll: false, // показать весь текст
            showButtonUnfolding: false, // показать кнопку
"показать всеЭ
            text: (item.is_checked &&
item.check_data.feedback.text) ? (item.check_data.feedback.text) : (''),
            attachedFiles: (item.is_checked) ?
(attachedFilesComment.files) : ([]),
            attachedImages: (item.is_checked) ?
(attachedFilesComment.images) : ([]),
        },
        rightAnswer: {
            isShow: item.check_data.is_send_correct,
            showAll: false, // показать весь текст
            showButtonUnfolding: false, // показать кнопку
"показать все"
            text: item.correct_text,
            attachedFiles: allRightFiles.files,
            attachedImages: allRightFiles.images,
        }
    });

    break;

case 4: // смартбук
    numberPoint++;
    countPointSubtask++;

    if (item.answer.is_send) {
        countAnswersOnPointSubtask++;
    }

    allAttachedFiles = this.getStructFiles(item.extends);
    deadlineStr = response.lesson.deadline_str2;
    deadline = response.lesson.deadline;

    if (item.is_deadline) {
        deadlineStr = item.deadline_str;
        deadline = item.deadline;
    }

```

```

item.notes.forEach((note, j) => {
  notesFiles = this.getStructFiles(note.extends);

  mistakes.push({
    id: note.local_id,
    idServer: note.id,
    show: false, // открыть окно с ошибкой
    title: note.type_description,
    comment: note.message.replaceAll(/\n\r/g, '<br/>'),
    images: notesFiles.images,
    files: notesFiles.files,
  });

  if (!mistakesAddTypes.includes(note.type)) {
    mistakesAddTypes.push(note.type);
    mistakesByTypes.push({
      id: note.type,
      title: note.type_description,
      count: 1
    })

    if (note.type !== 1) countMistakes++;
  } else {
    for (let k = 0; k < mistakesByTypes.length; k++) {
      if (mistakesByTypes[k].id === note.type) {
        mistakesByTypes[k].count += 1;
        if (note.type !== 1) countMistakes++;
        break;
      }
    }
  }
});

if (item.is_checked) {
  studentExtends =
this.getStructFiles(item.answer.extends, 'isHighlightingContent');

  if (item.check_data.feedback.extends) {
    attachedFilesComment =
this.getStructFiles(item.check_data.feedback.extends);
  } else {
    attachedFilesComment = {
      files: [],
      images: [],
    }
  }

  studentExtends.images.forEach((img, i) => {
    img.mistake.forEach((item, j) => {
      let indexMistakesByTypes = undefined;
      for (let i = 0; i < mistakesByTypes.length; i++) {
        if (item.typeMistake === mistakesByTypes[i].id) {
          indexMistakesByTypes = i;
          break;
        }
      }
    })
  })
}

```

```

        if (indexMistakesByTypes === undefined) {
            mistakesByTypes.push({
                id: item.typeMistake,
                title: item.title,
                count: 1
            });

        } else {
            mistakesByTypes[indexMistakesByTypes].count += 1;
        }

        if (item.typeMistake !== 1) countMistakes++;
    })
});

item.criteria.forEach((elem, j) => {
    criteriaScoreReceived += Number(elem.bal);
    criteriaScoreAll += Number(elem.max_bal);

    criteria.push({
        id: elem.id,
        prefix: elem.prefix,
        number: elem.number,
        text: elem.text,
        score: {
            max: elem.max_bal, // максимальный балл
            received: elem.bal // полученный балл
        }
    })
})
} else {
    studentExtends =
this.getStructFiles(item.answer.extends);
}

    answerText =
reverseHtmlEncoding(removeTagsText(item.answer.text));
    answerHTML = item.answer.text.replaceAll(' ',
'&nbsp;&nbsp; ').replace(/\r?\n/g, '<br>');

    this.data.push({
        id: item.id,
        savingStatus: 'none', // none - нет изменений / error -
ошибка / successfully - успешно / inProgress - в процессе
        savingBlockOpened: false, // блок с сохранением
открыть? / только для типов error и successfully
        type: 4, // 1 - блок дз с возможностью ответа / 2 -
блок с тестами / 3 - текстовый блок / 4 - смартбук
        typeAPI: item.type,
        isRevision: item.answer.is_revision, // Отправлено на
доработку?
        isAccessPaste: item.is_access_paste, // разрешить
вставку ответа
        responseRequired: item.is_answer, // требуется ли ответ
        show: true, // развернуть / скрыть блок с ДЗ
        isExtends: item.is_extends, // можно ли прикреплять
файлы // item.is_extends

```

```

        isSend: item.answer.is_send, // данные отправлены?
        isChecked: item.is_checked, // данные проверены?
        isStartCheck: ((item.is_start_check === undefined) ||
(item.is_start_check === null)) ? (true) : (item.is_start_check), //
проверка данного пункта начата?
        title: (item.header && (item.header.length > 0)) ?
(item.header) : ('Задание'),
        answer: answerText,
        answerOld: answerText, // для отслеживания вставки
текста
        answerHTML: answerHTML, // answer преобразованный в
html
        savedAnswer: answerText, // сохраненный ответ
        number: numberPoint,
        queryCounter: { // счетчик запросов
            all: 0,
            done: 0,
        },
        deadline: {
            time: (item.deadline === null) ? (null) : (0), //
время оставшееся до дедлайна в секундах (23:00:00)
            timeString: '', // время оставшееся до дедлайна -
текстовое поле (23:00:00)
            timeFull: (item.is_answer) ? (deadlineStr) : (''),
            timeDeadline: (item.is_answer) ? (deadline) : (''),
// время дедлайна в секундах (15.10.21)
        },
        text: item.text,
        textWithoutTags: item.text.replace(/<(.|\n)*?>/g, ''),
        attachedFiles: allAttachedFiles.files,
        attachedImages: allAttachedFiles.images,
        downloadStudentFiles: studentExtends.files,
        downloadStudentImages: studentExtends.images,

        // данные, приходящие после проверки пункта дз
        scoreText: (item.is_checked) ? (item.check_data.bal + '
' + this.getWordDeclension(Number(item.check_data.bal), 'балл') + ' из '
+ item.max_bal) : (''),
        textChecked: item.answer.text.replaceAll("\n",
"</br>"),
        activeMistake: undefined, // id открытой ошибки
        mistake: mistakes,
        mistakesByTypes: mistakesByTypes,
        countMistakes: countMistakes,
        criteria: criteria,
        comments: { // Комментарий к ответу
            isShow: (item.is_checked &&
((item.check_data.feedback.text && (item.check_data.feedback.text.length
> 0)) || (item.check_data.feedback.extends &&
(item.check_data.feedback.extends.length > 0))))),
            showAll: false, // показать весь текст
            showButtonUnfolding: false, // показать кнопку
"показать всеЭ
            text: (item.is_checked &&
item.check_data.feedback.text) ? (item.check_data.feedback.text) : (''),
            attachedFiles: (item.is_checked) ?
(attachedFilesComment.files) : ([]),

```

```

        attachedImages: (item.is_checked) ?
(attachedFilesComment.images) : ([]),
    },
    rightAnswer: {
        isShow: false,
        showAll: false, // показать весь текст
        showButtonUnfolding: false, // показать кнопку
"показать все"
        text: '',
        attachedFiles: [],
        attachedImages: [],
    }
    });

    break;
}
});

let th = this;
setTimeout(() => {
    let elemBlockSelected =
document.getElementsByClassName('disabled_select');
    for (let i = 0; i < elemBlockSelected.length; i++) {
        elemBlockSelected[i].onselectstart = returnFalse;
    }

    function returnFalse() {
        return false;
    }

    th.onResize();
}, 1);

if (th.$store.getters.getFontIsLoaded) { // шрифты загружены?
    setTimeout(() => {
        setTimeout(() => {
            th.visibleTaskBlock = true;
        }, 100);

        th.configureVisibleButtonText();
        th.addAnimationSlide();
    }, 250);
} else {
    databus.$on('fontsHasLoaded', (param) => { // шрифты были
загружены
        setTimeout(() => {
            setTimeout(() => {
                th.visibleTaskBlock = true;
            }, 100);

            th.configureVisibleButtonText();
            th.addAnimationSlide();
        }, 250)
    })
}
})
).catch((error) => {

```

```

console.log('error');
console.log(error.response);
console.log(error);

if (!this.dataReceived) this.dataReceived = true;

this.visibleTaskBlock = true;

if (error.response && error.response.data) {
  let response = error.response.data;
  if (response.error === "exam_not_checked") {
    let courseName =
this.$store.getters.getAllCourses[Number(this.$route.params.idCourse)].de
clinationTitle;
    this.isDiagnostics = response.gk_info.is_hw_diagnostic;

    if (this.$route.name === 'gkGameHomework') {
      this.breadcrumbs =
this.getBreadcrumbs(response.level.number, response.lesson.name,
this.isDiagnostics);
    } else if (this.$route.name === 'gkHomework') { // если это
дз / диагностическое задание в уроке
      this.breadcrumbs = [ // хлебные крошки
        {
          id: 1,
          title: 'Курс ' + courseName,
          link: {
            name: 'course',
            params: {idCourse: this.$route.params.idCourse,
idGroup: this.$route.params.idGroup}
          }
        },
        {
          id: 2,
          title: response.lesson.name,
          link: {
            name: 'gkLesson',
            params: {
              idCourse: this.$route.params.idCourse,
              idGroup: this.$route.params.idGroup,
              idLesson: this.$route.params.idLesson
            }
          }
        }
      ],
        {
          id: 3,
          title: (this.isDiagnostics) ? ('Диагностическая
работа') : ('Домашнее задание'),
          link: {
            name: undefined,
            params: {}
          }
        }
      ]
    } else { // если это диагностика

      if (this.$route.name === 'gkGameDiagnostics') {

```

```

        this.breadcrumbs =
this.getBreadcrumbs(response.level.number, response.lesson.name,
this.isDiagnostics);
    } else {
        this.breadcrumbs = [ // хлебные крошки
            {
                id: 1,
                title: 'Курс ' + courseName,
                link: {
                    name: 'course',
                    params: {idCourse: this.$route.params.idCourse,
idGroup: this.$route.params.idGroup}
                }
            },
            {
                id: 3,
                title: (this.isDiagnostics) ? ('Диагностическая
работа') : ('Домашнее задание'),
                link: {
                    name: undefined,
                    params: {}
                }
            }
        ];
    }
}

this.titlePage = response.lesson.name;
this.showDeadline = false;
this.isSend = true;
this.isChecked = false;
} else if (response.error === "deadline_is_expired") {
    this.openModalExpiredPassing();
}
}
}))
},

/** Получить хлебные крошки для ГК и для ПК
 * @param {Array} data - массив объектов формата {title: 'any',
routeName: 'any'}
 * @returns список с хлебными крошками
 * */
breadcrumbsConstructor(data) {
    return data.map((item, i) => {
        return {
            id: i,
            title: item.title,
            link: {
                name: item.routeName,
                params: {}
            }
        }
    })
}

/** Получить список хлебных крошек

```

```

* @param {Number} levelNumber - номер уровня
* @param {String} titleLesson - заголовок урока
* @param {Boolean} isDiagnostics - это диагностическая работа
* @returns {Array} список хлебных крошек
* */
getBreadcrumbs(levelNumber, titleLesson, isDiagnostics) {
  /** Получить наименование родительского маршрута группы
   * @param {String} name - наименование текущего маршрута
   * @returns {String}
   * */
  let getParentRouteName = (name) => {
    switch (name) {
      case 'gkGameLesson':
        return 'gkLevelsMap';
    }

    return null;
  };

  let routeName = getParentRouteName(this.$route.name);

  if (isDiagnostics) {
    return this.breadcrumbsConstructor([
      {title: 'Список уровней', routeName: 'gkLevelsMap'},
      {title: `${getTextNumberWord(levelNumber)} уровень`,
routeName: 'gkLessons'},
      {title: isDiagnostics ? 'Диагностическая работа' : 'Домашнее
задание', routeName: undefined},
    ]);
  }

  return this.breadcrumbsConstructor([
    {title: 'Список уровней', routeName: 'gkLevelsMap'},
    {title: `${getTextNumberWord(levelNumber)} уровень`, routeName:
'gkLessons'},
    {title: titleLesson, routeName: 'gkGameLesson'},
    {title: isDiagnostics ? 'Диагностическая работа' : 'Домашнее
задание', routeName: undefined},
  ]);
},

// время тестирования вышло
timeIsUp(data) {
  console.log('--- timeIsUp ---'); // TODO: исправить / дописать
  return;

  document.activeElement.blur();

  databus.$emit('openGlobalNotifications', {
    show: true,
    time: 4,
    text: 'Время, отведенное на диагностическое задание, вышло',
    type: 'error'
  })

  setTimeout(() => {
    this.sendAnswerOnLesson();
  });
}

```

```

    }, 800);
  },

  // закрыть модальное окно уведомления о не выполненных заданиях
  closeModalInfoSendWork() {
    this.modalEmptyTask.show = false;
  },

  // проверить поля на заполненность и отправить данные на сервер, при
  // условии что все ок
  checkAndSendAnswer() {
    let tasksWithoutAnswer = [];
    let countTestBlock = 0;
    let countPracticeBlock = 0;

    this.data.forEach((item, i) => {
      switch (item.type) {
        case 1: // блок дз с возможностью ответа

          if (item.responseRequired && (item.answer.length === 0) &&
            (item.downloadStudentImages.length === 0) &&
            (item.downloadStudentFiles.length === 0)) {
            countPracticeBlock++;

            tasksWithoutAnswer.push({
              type: item.type,
              title: item.title
            })
          }

          break;

        case 2: // блок с тестами

          if (!this.isDiagnostics) {

            for (let i = 0; i < item.tests.length; i++) {
              if (!item.tests[i].isChecked) {
                countTestBlock++;

                tasksWithoutAnswer.push({
                  type: item.type,
                  title: item.title
                });
              }

              break;
            }
          }

          break;
      }
    });

    if (tasksWithoutAnswer.length > 0) {

```

```

this.modalEmptyTask.show = true;
tasksWithoutAnswer.sort((prev, next) => prev.type - next.type);
let textNotification = 'Вы не выполнили ';

if (countPracticeBlock > 0) {
  if (countPracticeBlock > 1) {
    textNotification += 'задания в блоках '
  } else {
    textNotification += 'задание в блоке '
  }

  tasksWithoutAnswer.forEach((item, i) => {
    if ((item.type === 1) || (item.type === 4)) {
      textNotification += '«' + item.title + '»';

      if ((tasksWithoutAnswer.length - 1) !== i) {
        textNotification += ', ';
      }
    }
  })
}

if ((countTestBlock > 0) && (countPracticeBlock > 0)) {
  textNotification += ', а также ';
}

if (countTestBlock > 0) {

  if (countTestBlock > 1) {
    textNotification += 'задания в блоках '
  } else {
    textNotification += 'задание в блоке '
  }

  tasksWithoutAnswer.forEach((item, i) => {
    if (item.type === 2) {
      textNotification += '«' + item.title + '»';

      if ((tasksWithoutAnswer.length - 1) !== i) {
        textNotification += ', ';
      }
    }
  })
}

let textWork = (this.isDiagnostics) ? ('экзамен') : ('работу');
textNotification += '. ';
textNotification += 'Уверены что желаете отправить ' + textWork +
' на проверку?';

this.modalEmptyTask.text = textNotification;

} else {
  setTimeout(() => {
    this.sendAnswerOnLesson();
  }, 100);
}

```

```

    }
  },

  // отправить на сервер информацию о том что практика / экзамен
  // завершен
  sendAnswerOnLesson(allQuestionSaved) {

    if (this.modalEmptyTask.show) {
      this.modalEmptyTask.show = false;
    }

    if (allQuestionSaved || this.allQuestionSaved()) {
      let data = {
        module: 'hw_close_exam',
        group_id: this.idGroup,
        lesson_id: this.$route.params.idLesson,
        // time // для экзамена / прошло времени с прошлой пинги, не
        // обязателен, по умолчанию = 0, передавать надо только в случае если таймер
        // не строгий
      };

      console.log('data');
      console.log(data);

      send
      .post('lesson.php', data)
      .then((response) => {
        console.log(response)
        console.log('отправлено на проверку');

        this.getData();
      })
      .catch((error) => {
        console.log('error');
        console.log(error.response);
        console.log(error);

        this.getData();

        let response = error.response.data;
        if (response.error === "deadline_is_expired") {
          this.openModalExpiredPassing();
        }
      });
    } else {
      this.showModalWaitingQuery = true;

      let timeDuration = 3000;
      let sumTime = 0;

      this.resendFailedQuery();

      intervalSendAnswer = setInterval(() => {
        sumTime += timeDuration;

        if (this.allQuestionSaved()) {
          this.showModalWaitingQuery = false;
        }
      }, timeDuration);
    }
  }
}

```

```

        clearInterval(intervalSendAnswer);
        this.sendAnswerOnLesson(true);
    } else if (sumTime <= 10000) {
        this.resendFailedQuery();
    } else { // если в течении 10сек не получен ответ от сервера
        вывести ошибку
        this.notification.show = true;
        this.notification.text = 'Не удалось отправить ответ,
        перезагрузите страницу и попробуйте снова. Если ошибка повторится,
        обратитесь в <a href="https://vk.com/im?sel=381434758"
        target="_blank">тех. поддержку</a>';
        this.notification.type = 'error';
        this.notification.time = 6;

        this.showModalWaitingQuery = false;
        clearInterval(intervalSendAnswer);
    }
    }, timeDuration)
    },
    },

    // проверить, все ли вопросы были сохранены
    allQuestionSaved() {
        let tasks = this.data.filter((item) => item.typeAPI === 3);

        for (let item of tasks) {
            if (!((item.savingStatus === 'successfully') ||
            (item.savingStatus === 'none')))) {
                return false;
            }
        }

        return true
    },

    // отправить ответы на вопросы повторно (отправка ответов которых не
    была успешна)
    resendFailedQuery() {
        for (let [i, item] of this.data.entries()) {
            if ((item.typeAPI === 3) && (item.savingStatus === 'error')) {
                this.savePartHomework(i);
            }
        }
    },

    // поворот изображения в слайдере
    sliderRotateImage(item) {

this.data[this.activePoint].downloadStudentImages[item.index].rotateDegree = item.degree;
        this.recalculationImageSize(this.activePoint, item.index);

        let data = {
            module: 'hw_change_rotate_extends_query_points',
            group_id: this.idGroup,
            lesson_id: this.$route.params.idLesson,
            point_id: this.data[this.activePoint].id,

```

```

        extend_id:
this.data[this.activePoint].downloadStudentImages[item.index].id,
        rotate: item.degree
    };

    send
        .post('lesson.php', data)
        .then((response) => {
            console.log('угол изменен')
        })
        .catch((error) => {
            console.log('error');
            console.log(error.response);
            console.log(error);

            let response = error.response.data;
            if (response.error === "deadline_is_expired") {
                this.openModalExpiredPassing();
            }
        });
},

// задать вопрос
goToChat(id) {
    this.notification.show = true;
    this.notification.text = 'Функционал находится в разработке';
    this.notification.type = 'error';
    this.notification.time = 2;
},

// удалить изображение
deleteSlide(item) {
    // this.activePoint
    this.deleteDownloadImage(this.activePoint, item.index);
    this.closeSlider();
},

// открыть модальное окно слайдера выделения по фото
openSliderHighlightingImage(item, i, j) {
    this.showHighlightingImage = true;
    this.contentHighlightingImage = item;
    this.activeHighlightingImage = j;
},

// закрыть модальное окно слайдера выделения по фото
closeSliderHighlightingImage() {
    this.showHighlightingImage = false;
    this.activeHighlightingImage = 0;
    this.contentHighlightingImage = [];
},

// вернуть склонение слова
getWordDeclension(count, text) {
    if ((count >= 5) && (count <= 20)) {
        return text + 'ов'

    } else if ((count % 10) === 1) {

```

```

        return text + ''

    } else if (((count % 10) >= 2) && ((count % 10) <= 4)) {
        return text + 'a'

    } else {
        return text + 'ов'

    }
},

// при изменении пути пересчитать размеры повернутых изображений
onChangeRouter() {
    this.data.forEach((item, i) => {
        if ((item.type === 1) || (item.type === 2) || (item.type === 4))
{
            this.data[i].show = true;
        }
    });

    setTimeout(() => {
        this.recalculationImagesSize();

        setTimeout(() => {
            this.data.forEach((item, i) => {
                if ((item.type === 1) || (item.type === 2) || (item.type ===
4)) {
                    this.data[i].show = false;
                }
            });
        })
    }, 1);
},

// привести файлы и картинки к требуемому формату
getStructFiles(item, type) {
    let images = [];
    let files = [];
    item.map(item => {
        if (item.file.type && (item.file.type.indexOf('image/') !== -1))
{
            if ((type !== 'isHighlightingContent') || (item.rotate ===
null) || (item.rotate === undefined)) {
                let orientation = 'landscape';
                let rotate = 0;

                if (item.rotate) {
                    if (item.orientation) {
                        orientation = 'landscape';
                    } else {
                        orientation = 'portrait';
                    }

                    rotate = item.rotate;
                }

                images.push({

```

```

        id: item.id,
        content: item.file.full_link,
        rotateDegree: rotate, // градус поворота изображения
        orientation: orientation, // portrait - false / landscape
- true, // portrait / landscape
    })
  } else {
    let orientation;

    if (item.orientation) {
      orientation = 'landscape';
    } else {
      orientation = 'portrait';
    }

    let mistakes = [];
    item.notes.forEach((note, i) => {

      let notesFiles = this.getStructFiles(note.extends);

      let left = 0,
          top = 0,
          width = 0,
          height = 0,
          figureId = -mistakes.length;

      if (note.figures[0]) {
        left = note.figures[0].x;
        top = note.figures[0].y;

        if (note.figures[0].w < 0) {
          left -= Math.abs(note.figures[0].w);
        }

        if (note.figures[0].h < 0) {
          top -= Math.abs(note.figures[0].h);
        }

        width = Math.abs(note.figures[0].w);
        height = Math.abs(note.figures[0].h);
        figureId = note.figures[0].id;
      }

      let classMistake = '';

      if (top < 50) {
        if ((Number(height) > 15) && ((Number(height) + top) >
65)) {
          classMistake += 'mistake__yPosition_default'
        } else {
          classMistake += 'mistake__yPosition_bottom'
        }
      } else {
        classMistake += 'mistake__yPosition_top'
      }

      let center = (Number(width) / 2) + left;

```

```

        if ((center > 40) && (center < 60)) {
            classMistake += ' mistake__xPosition_center'
        } else if (center <= 40) {
            classMistake += ' mistake__xPosition_left'
        } else {
            classMistake += ' mistake__xPosition_right'
        }

        mistakes.push({
            id: note.id,
            idError: note.local_id,
            highlightBlockId: figureId,
            highlightBlockPercent: {
                width: width, // ширина блока в % от родительского
                height: height, // высота блока в % от родительского
                left: left, // расстояние влево в % от родительского
                top: top // расстояние сверху в % от родительского
            },
            highlightBlockCount: {
                x: 0, // ширина блока
                y: 0, // высота блока
                w: 0, // расстояние влево
                h: 0 // расстояние сверху
            },
            classMistake: classMistake,
            show: false, // открыть окно с ошибкой
            edit: false, // активно ли окно с редактированием
            title: note.type_description,
            typeMistake: note.type, // тип ошибки
            comment: note.message,
            commentHTML: note.message.replaceAll('\n', '</br>'),
            images: notesFiles.images,
            files: notesFiles.files,
        })
    });

    images.push({
        id: item.id,
        content: item.file.full_link,
        rotateDegree: item.rotate, // градус поворота изображения
        orientation: orientation, // portrait - false / landscape
        - true, // portrait / landscape
        mistake: mistakes,
    })
}

} else {
    files.push({
        id: item.id,
        img: this.$store.getters.path + '/img/bx-file-blank.svg',
        title: item.file.name,
        size: this.getSizeFile(item.file.size),
    })
}

```

```

        link: item.file.full_link
    });
}
});

return {images: images, files: files}
},

// по изменению размеров окна браузера
onResize() {
    if (this.screenWidth !== window.innerWidth) {
        this.screenWidth = window.innerWidth;

        if (this.screenWidth <= 1160) {
            this.isMobileDevise = true;

            // для планшетов и мобильных устр. при изменении ширины экрана
            пересчитать отступы модальных окон ошибок
            this.data.forEach((item, i) => {
                if (((item.type === 1) || (item.type === 4)) &&
                    (item.activeMistake !== undefined)) {
                    this.setPositionErrorModalBlock(i, item.activeMistake)
                }
            })

        } else {
            this.isMobileDevise = false;
        }

        let oldInnerWidth = window.innerWidth;
        let oldInnerHeight = window.innerHeight;

        setTimeout(() => {
            let currentInnerWidth = window.innerWidth;
            let currentInnerHeight = window.innerHeight;

            if ((oldInnerWidth === currentInnerWidth) && (oldInnerHeight
            === currentInnerHeight)) {
                this.recalculationImagesSize();
                this.addAnimationSlide();
                this.pageHasBeenResized = true;
            }
        }, 100)
    }
},

// пересчитать размер изображения
recalculationImageSize(i, j) {
    let image = this.data[i].downloadStudentImages[j];
    let elemParent = document.getElementById('block_image_' + i + '_' +
j);
    let elemImage = document.getElementById('image_background_' + i +
'_' + j);
    let elemParentMobile =
document.getElementById('block_image_mobile_' + i + '_' + j);

```

```

    let elemImageMobile =
document.getElementById('image_background_mobile_' + i + '_' + j);

    if (elemImage && (window.innerWidth > 600)) { // для десктопных
устр. и для планшетов
        let elemParentWidth = elemParent.offsetWidth;
        let elemParentHeight = elemParent.offsetHeight;

        if ((image.orientation === 'landscape') && ((image.rotateDegree
=== 0) || (image.rotateDegree === 180))) {
            elemImage.style.width = '100%';
            elemImage.style.height = '100%';

        } else if (image.orientation === 'landscape') {
            elemImage.style.width = elemParentHeight + 'px';
            elemImage.style.height = elemParentWidth + 'px';

        } else if ((image.orientation === 'portrait') &&
((image.rotateDegree === 0) || (image.rotateDegree === 180))) {
            elemImage.style.width = '100%';
            elemImage.style.height = '100%';

        } else if (image.orientation === 'portrait') {
            elemImage.style.height = elemParentWidth + 'px';
            elemImage.style.width = elemParentHeight + 'px';
        }
    } else if (elemImageMobile && (window.innerWidth <= 600)) { // для
мобильных устройств
        let elemParentWidth = elemParentMobile.offsetWidth;
        let elemParentHeight = elemParentMobile.offsetHeight;

        if ((image.orientation === 'landscape') && ((image.rotateDegree
=== 0) || (image.rotateDegree === 180))) {
            elemImageMobile.style.width = '100%';
            elemImageMobile.style.height = '100%';

        } else if (image.orientation === 'landscape') {
            elemImageMobile.style.width = elemParentHeight + 'px';
            elemImageMobile.style.height = elemParentWidth + 'px';

        } else if ((image.orientation === 'portrait') &&
((image.rotateDegree === 0) || (image.rotateDegree === 180))) {
            elemImageMobile.style.width = '100%';
            elemImageMobile.style.height = '100%';

        } else if (image.orientation === 'portrait') {
            elemImageMobile.style.height = elemParentWidth + 'px';
            elemImageMobile.style.width = elemParentHeight + 'px';
        }
    }
},

// пересчитать размеры изображений
recalculationImagesSize() {
    let countIteration = 0;

    let recalculation = () => {

```

```

    this.data.forEach((item, i) => {
      if ((item.type === 1) || (item.type === 4)) {
        item.downloadStudentImages.forEach((image, j) => {
          let elemParent = document.getElementById('block_image_' + i
+ '_' + j);
          let elemImage = document.getElementById('image_background_'
+ i + '_' + j);
          let elemParentMobile =
document.getElementById('block_image_mobile_' + i + '_' + j);
          let elemImageMobile =
document.getElementById('image_background_mobile_' + i + '_' + j);

          if (!elemParent || !elemImage) { // не загрузился блок
            if (countIteration > 60) return false;

            setTimeout(() => {
              countIteration++;
              recalculation();
            }, 100);
          } else if (elemImage && (window.innerWidth > 600)) {
            let elemParentWidth = elemParent.offsetWidth;
            let elemParentHeight = elemParent.offsetHeight;

            if ((image.orientation === 'landscape') &&
((image.rotateDegree === 0) || (image.rotateDegree === 180))) {
              elemImage.style.width = '100%';
              elemImage.style.height = '100%';

            } else if (image.orientation === 'landscape') {
              elemImage.style.width = elemParentHeight + 'px';
              elemImage.style.height = elemParentWidth + 'px';

            } else if ((image.orientation === 'portrait') &&
((image.rotateDegree === 0) || (image.rotateDegree === 180))) {
              elemImage.style.width = '100%';
              elemImage.style.height = '100%';

            } else if (image.orientation === 'portrait') {
              elemImage.style.height = elemParentWidth + 'px';
              elemImage.style.width = elemParentHeight + 'px';
            }
          } else if (elemImageMobile && (window.innerWidth <= 600)) {
            let elemParentWidth = elemParentMobile.offsetWidth;
            let elemParentHeight = elemParentMobile.offsetHeight;

            if ((image.orientation === 'landscape') &&
((image.rotateDegree === 0) || (image.rotateDegree === 180))) {
              elemImageMobile.style.width = '100%';
              elemImageMobile.style.height = '100%';

            } else if (image.orientation === 'landscape') {

              elemImageMobile.style.width = elemParentHeight + 'px';
              elemImageMobile.style.height = elemParentWidth + 'px';
            }
          }
        });
      }
    });

```

```

        } else if ((image.orientation === 'portrait') &&
((image.rotateDegree === 0) || (image.rotateDegree === 180))) {

            elemImageMobile.style.width = '100%';
            elemImageMobile.style.height = '100%';

        } else if (image.orientation === 'portrait') {

            elemImageMobile.style.height = elemParentWidth + 'px';
            elemImageMobile.style.width = elemParentHeight + 'px';
        }
    }
    })
}
});

recalculation();
},

// клик по блоку с исправленным feedback'ом
openMistakeBlock(event, i) {
    // console.log(event);
    if (event.target.className.indexOf('essay_error') !== -1) {
        let index = event.target.id.indexOf('essay_error_');
        let idEl = Number(event.target.id.slice(index + 12)); // получаем
id ошибки

        let mistake = this.data[i].mistake;

        for (let j = 0; j < mistake.length; j++) {
            if (mistake[j].id === idEl) {
                if (!mistake[j].show) {
                    this.showMistakeModal(mistake[j].show, i, j);
                    this.scrollMistakeBlock(i, mistake[j].id)
                }

                break;
            }
        }
        // console.log('idEl = ' + idEl);
    }
},

// проскролить блок ошибок до элемента
scrollMistakeBlock(i, id) {
    // плавная прокрутка
    /* let blockMistakes = document.getElementById('mistakes_block_'
+ i);
    let blockMistake = document.getElementById('mistake_' + i + '_' +
id);

    let scrollCount = blockMistakes.scrollTop;
    let offsetTop = blockMistake.offsetTop - 20;
    let offsetClear = offsetTop - scrollCount;

    let timeAnimation = 500;

```

```

    let fps = 60;
    let countFrames = (timeAnimation/1000)*fps; // кол-во кадров
анимации

    let step = offsetClear / countFrames;
    let stepNumber = 0;

    console.log('interval')
    console.log('scrollCount = ' + scrollCount)
    console.log('step = ' + step)

    let interval = setInterval(() => {
        blockMistakes.scrollTo(0, (scrollCount + step))
        stepNumber++;

        scrollCount = blockMistakes.scrollTop;
        offsetTop = blockMistake.offsetTop - 20;
        offsetClear = offsetTop - scrollCount;
        step = offsetClear / countFrames;

        if (stepNumber === countFrames) {
            clearInterval(interval)
        }
    }, (timeAnimation/countFrames));    */

    // не плавная прокрутка
    /*
        setTimeout(() => {
            blockMistakes.scrollTo(0, blockMistake.offsetTop - 20)
        }, 500)    */
    },

    // переключить видимость отображения модального окна с ошибками
visibleMistakeModal(show, i, j) {
    if (show) {
        this.hideMistakeModal(show, i, j);
    } else {
        this.showMistakeModal(show, i, j);
    }
},

    // скрыть модальное окно с ошибкой для планшетов и мобильных
устройств
hideMistakeModalTablet(event, id, i, j) {
    if (this.isMobileDevise) {
        this.data.forEach((item, m) => {
            if (this.data[m].activeMistake !== undefined) {
                let path = event.path || (event.composedPath &&
event.composedPath());
                let checkClickByFiles = false;

                for (let i = 0; i < path.length; i++) {

                    if ((path[i].nodeName === 'SPAN') &&
(path[i].className.indexOf('essay_error') !== -1)) {
                        checkClickByFiles = true;

```

```

        break;
    }
}

    if (!checkClickByFiles) {
        this.hideMistakeModal(false, m, j)
    }

    }
    });
}
},

// скрыть модальное окно с ошибкой
hideMistakeModal(show, i, j) {
    this.data[i].mistake.forEach((item, k) => {
        this.data[i].activeMistake = undefined;
        this.data[i].mistake[k].show = false;

        let el = document.getElementById('essay_error_' + item.id);
        if (el.className.indexOf('essay_error__active') !== -1) {
            el.classList.remove('essay_error__active');
        }
    });
},

// установить позицию модальному окну ошибок при разрешениях < 1160px
setPositionErrorModalBlock(i, id) {
    let el = document.getElementById('essay_error_' + id);
    let errorTop = el.offsetTop;
    let errorLeft = el.offsetLeft;
    let widthError = el.offsetWidth;
    let heightError = el.offsetHeight;
    let elemModalError = document.getElementById('mistake_tablet_' + i
+ '_' + id);
    let elemModalErrorDefault =
document.getElementById('mistake_tablet_default_' + i);
    let elemModalErrorWidth = elemModalErrorDefault.offsetWidth;
    elemModalError.style.top = (errorTop + heightError + 4) + 'px';

    let answerBlock = document.getElementById('answer_test_' + i);
    let answerBlockPaddingLeft =
Number(window.getComputedStyle(answerBlock,
null).getPropertyValue('padding-left').slice(0, -2));
    let widthAnswerBlock = answerBlock.offsetWidth;
    let errorRight = widthAnswerBlock - errorLeft - widthError;

    if ((errorLeft + widthError) > widthAnswerBlock) {
        elemModalError.style.left = answerBlockPaddingLeft + 'px';
        elemModalError.style.right = 'auto';
    } else if ((errorRight + widthError) > elemModalErrorWidth) {
        elemModalError.style.left = errorLeft + 'px';
        elemModalError.style.right = 'auto';
    } else if ((errorLeft + widthError) < elemModalErrorWidth) {
        elemModalError.style.left = answerBlockPaddingLeft + 'px';
        elemModalError.style.right = 'auto';
    }
}

```

```

    } else {
      elemModalError.style.right = errorRight + 'px';
      elemModalError.style.left = 'auto';
    }
  },

  // показать модальное окно с ошибкой
  showMistakeModal(show, i, j) {
    this.data[i].mistake.forEach((item, k) => {
      if (j === k) {
        this.data[i].mistake[k].show = true;
        this.data[i].activeMistake = this.data[i].mistake[k].id;

        let el = document.getElementById('essay_error_' + item.id);
        if (el) {
          el.classList.add('essay_error__active');

          if (this.isMobileDevise) {
            this.setPositionErrorModalBlock(i, item.id)
          }
        }
      }
    });
  },

  // сохранить данный пункт (блок) дз
  savePartHomework(i, isDraft = false) {
    let notificationSavedDraft = () => {
      this.notification.show = true;
      this.notification.text = 'Успешно! Данный ответ был сохранен как черновик';
      this.notification.type = 'successfully';
      this.notification.time = 5;
    }

    if (this.data[i].savingStatus === 'inProgress' &&
    this.data[i].savedAnswer === this.data[i].answer) {
      if (isDraft) setTimeout(notificationSavedDraft, 1000);
      return;
    }

    this.setNewStatusItem(i, 'inProgress');
    this.data[i].savedAnswer = this.data[i].answer;
    this.currentSavingIndex = i;
    this.data[i].queryCounter.all++;

    let data = {
      module: 'hw_set_answer_query_points',
      group_id: this.idGroup,

```

```

        lesson_id: this.$route.params.idLesson,
        point_id: this.data[i].id,
        text: this.data[i].answer,
    };

    send
    .post('lesson.php', data)
    .then((response) => {
        this.data[i].queryCounter.done++;
        if (isDraft) notificationSavedDraft();
        this.setNewStatusItem(i, 'successfully');
    })
    .catch((error) => {
        console.log('error');
        console.log(error.response);
        console.log(error);
        this.setNewStatusItem(i, 'error');

        let response = error.response.data;
        if (response.error === "deadline_is_expired") {
            this.openModalExpiredPassing();
        } else {
            this.notification.show = true;
            this.notification.text = 'Не удалось отправить ответ,
перезагрузите страницу и попробуйте снова. Если ошибка повторится,
обратитесь в <a href="https://vk.com/im?sel=381434758"
target="_blank">тех. поддержку</a>';
            this.notification.type = 'error';
            this.notification.time = 10;
        }
    });
},

// отправить данный пункт (блок) дз на сервер
sendPartHomework(i) {
    let stringIsEmpty = (string) => {
        const symbolsReplace = [' ', '\n', '/', '-', '_', '|'];

        for (let i = 0; i < string.length; i++) {
            if (!symbolsReplace.includes(string[i])) return false
        }

        return true
    };

    if (
        (this.data[i].downloadStudentFiles.length === 0) &&
        (this.data[i].downloadStudentImages.length === 0) &&
        stringIsEmpty(this.data[i].answer)
    ) {
        this.notification.show = true;
        this.notification.text = 'Вы ввели некорректный ответ в поле
ввода';
        this.notification.type = 'error';
        this.notification.time = 4;

        return false;
    }
}

```

```

    }

    if ((this.$route.name === "ffHomework") &&
        (getMoscowTime().timeInSeconds >= this.timeDeadline)) {
        this.userIsBlocked = true;

        this.notification.show = true;
        this.notification.text = 'Ошибка! Время, в которое можно
отправить работу, вышло';
        this.notification.type = 'error';
        this.notification.time = 4;

        return false;
    }

    // проверяет наличие пунктов, отправленных на доработку
    let checkRevisionPoint = () => {
        for (let k = 0; k < this.data.length; k++) {
            if (this.data[k].isRevision) {
                return true;
            }
        }

        return false;
    };

    if (this.data[i].queryCounter.all ===
this.data[i].queryCounter.done) {
        let data = {
            module: 'hw_send_answer_query_points',
            group_id: this.idGroup,
            lesson_id: this.$route.params.idLesson,
            point_id: this.data[i].id,
            text: this.data[i].answer,
        };

        send
        .post('lesson.php', data)
        .then((response) => {

            this.notification.show = true;
            this.notification.text = 'Ответ отправлен на проверку';
            this.notification.type = 'successfully';
            this.notification.time = 4;

            this.data[i].isSend = true;
            if (this.data[i].type === 4) this.data[i].isChecked = true;

            let text = document.getElementById('answer_textarea_' +
this.data[i].id).value;
            this.data[i].answerHTML = text.replaceAll(' ',
'&nbsp;&nbsp;&nbsp;').replace(/\r?\n/g, '</br>');

            if (this.data[i].isRevision) this.data[i].isRevision = false;
            if (this.isRevision) this.isRevision = checkRevisionPoint();
        })
        .catch((error) => {

```

```

        console.log('error');
        console.log(error.response);
        console.log(error);

        let response = error.response.data;
        if (response.error === "deadline_is_expired") {
            this.openModalExpiredPassing();
        } else if (response.error === "answer_is_send") {
            this.notification.show = true;
            this.notification.text = 'Успешно! Ответ был изменен';
            this.notification.type = 'successfully';
            this.notification.time = 4;

            this.data[i].isSend = true;

            let text = document.getElementById('answer_textarea_' +
this.data[i].id).value;
            this.data[i].answerHTML = text.replaceAll(' ',
'&nbsp;&nbsp; ').replace(/\r?\n/g, '</br>');
        }
    });
    } else {
        this.showModalWaitingQuery = true;
        let sumTime = 0;
        intervalSendAnswer = setInterval(() => {
            sumTime += 1000;

            if (this.data[i].queryCounter.all ===
this.data[i].queryCounter.done) {
                this.showModalWaitingQuery = false;
                clearInterval(intervalSendAnswer);
                this.sendPartHomework(i);
            } else if (sumTime >= 10000) { // если в течении 10сек не
получен ответ от сервера вывести ошибку
                this.notification.show = true;
                this.notification.text = 'Не удалось отправить ответ,
перезагрузите страницу и попробуйте снова. Если ошибка повторится,
обратитесь в <a href="https://vk.com/im?sel=381434758"
target="_blank">тех. поддержку</a>';
                this.notification.type = 'error';
                this.notification.time = 6;

                this.showModalWaitingQuery = false;
                clearInterval(intervalSendAnswer);
            }
        }, 1000)
    }
},

closeModalNotification() {
    this.notification.show = false;
},

// прикрепить файлы к блоку задания
addFile(obj) {
    console.log('addFile')
    console.log(obj)

```

```

let index = this.download.activeElemIndex;
this.data[index].queryCounter.all++;

let data = {
  module: 'hw_add_extends_query_points',
  group_id: this.idGroup,
  lesson_id: this.$route.params.idLesson,
  point_id: this.data[index].id,
  file_id: obj.id,
  file_hash: obj.hash,
  file_name: obj.data.name,
};

console.log('hw_add_extends_query_points');
console.log(data);

send
  .post('lesson.php', data)
  .then((response) => {
    this.data[index].queryCounter.done++;

    if (obj.data.type.indexOf('image/') !== -1) {
      this.data[index].downloadStudentImages.push({
        id: response.extend_id,
        content: `${obj.content}`, // `${obj.content}`
        rotateDegree: 0, // градус поворота изображения
        orientation: (response.orientation) ? ('landscape') :
('portrait'), // portrait - false / landscape - true
      });

    } else {
      this.data[index].downloadStudentFiles.push({
        id: response.extend_id,
        img: this.$store.getters.path + '/img/bx-file-blank.svg',
        title: obj.data.name,
        size: this.getSizeFile(obj.data.size),
        link: obj.link
      })
    }
  })
  .catch((error) => {
    console.log('error');
    console.log(error.response);
    console.log(error);

    let response = error.response.data;

    switch (response.error) {
      case 'deadline_is_expired':
        this.openModalExpiredPassing();
        break;

      case 'file_is_set':
        this.notification.show = true;
    }
  })

```

```

        this.notification.text = 'Данный файл уже добавлен к
данному заданию!';
        this.notification.type = 'error';
        this.notification.time = 5;
        this.data[index].queryCounter.done++;
        break;
    }
    });
},

// перейти на стр. теста
goToTest(test) {

    if (this.isDiagnostics) {
        let nameTestPage = '';

        switch (this.$route.name) {
            case 'gkGameHomework':
                nameTestPage = 'gkGameTest';
                break;

            case 'gkHomeworkPractice':
                nameTestPage = 'gkTestPractice';
                break;

            case 'gkHomeworkPitStop':
                nameTestPage = 'gkTestPitStop';
                break;

            case 'gkDiagnosticsPractice':
                nameTestPage = 'diagnosticsTestPractice';
                break;

            case 'gkHomework':
                nameTestPage = 'gkTest';
                break;

            case 'gkGameDiagnostics':
                nameTestPage = 'diagnosticsGameTest';
                break;

            case 'gkDiagnostics':
                nameTestPage = 'diagnosticsTest';
                break;

            case 'pgkHomework':
                nameTestPage = 'pgkTest';
                break;

            case 'pgkDiagnostics':
                nameTestPage = 'pgkDiagnosticsTest';
                break;

            case 'theoryHomework':
                nameTestPage = 'theoryTest';
                break;
        }
    }
}

```

```

console.log("this.$route.name", !test.isChecked, this.isChecked);
if (!test.isChecked || this.isChecked) { // если тест не был
пройден
    this.$router.push({
        name: nameTestPage,
        params: {
            idTest: test.id
        }
    });
} else { // если тест уже был пройден
    this.showModalExplanation = true;
}
} else {
switch (this.$route.name) {
    case 'gkGameHomework':
        this.$router.push({
            name: 'gkGameTest', params: {
                idTest: test.id
            }
        });
        break;

    case 'gkHomeworkPractice':
        this.$router.push({
            name: 'gkTestPractice', params: {
                idTest: test.id
            }
        });
        break;

    case 'gkHomeworkPitStop':
        this.$router.push({
            name: 'gkTestPitStop', params: {
                idTest: test.id
            }
        });
        break;

    case 'gkDiagnosticsPractice':
        this.$router.push({
            name: 'diagnosticsTestPractice', params: {
                idTest: test.id
            }
        });
        break;

    case 'gkHomework':
        this.$router.push({
            name: 'gkTest', params: {
                idTest: test.id
            }
        });
        break;

    case 'theoryHomework':

```

```

        this.$router.push({
            name: 'theoryTest', params: {
                idTest: test.id
            }
        });
        break;

    case 'gkGameDiagnostics':
        this.$router.push({
            name: 'diagnosticsGameTest', params: {
                idTest: test.id
            }
        });
        break;

    case 'gkDiagnostics':
        this.$router.push({
            name: 'gkTest', params: {
                idTest: test.id
            }
        });
        break;

    case 'pgkHomework':
    case 'pgkDiagnostics':
        this.$router.push({
            name: 'pgkTest', params: {
                idTest: test.id
            }
        });
        break;

    case 'ffHomework':
        this.$router.push({
            name: 'ffTest', params: {
                idTest: test.id
            }
        });
        break;
    }
}
},

// закрыть модальное окно с пояснением к закрытому экзамену
closeModalExplanation() {
    this.showModalExplanation = false;
},

// открыть модальное окно загрузки файлов
openModalAddFiles(i) {
    this.download.showDownloadFilesModal = true;
    this.download.activeElemIndex = i;
},

closeModalAddFiles() {
    this.download.showDownloadFilesModal = false;
    // this.download.activeElemIndex = undefined;

```

```

},

// удалить загруженный (прикрепленный) файл к ответу дз
deleteDownloadFile(i, j) {
  this.data[i].queryCounter.all++;
  let data = {
    module: 'hw_remove_extends_query_points',
    group_id: this.idGroup,
    lesson_id: this.$route.params.idLesson,
    point_id: this.data[i].id,
    extend_id: this.data[i].downloadStudentFiles[j].id,
  };

  send
    .post('lesson.php', data)
    .then((response) => {
      this.data[i].queryCounter.done++;
      this.data[i].downloadStudentFiles.splice(j, 1);
    })
    .catch((error) => {
      console.log('error');
      console.log(error.response);
      console.log(error);

      let response = error.response.data;
      if (response.error === "deadline_is_expired") {
        this.openModalExpiredPassing();
      }
    });
},

// удалить загруженное (прикрепленное) изображение к ответу дз
deleteDownloadImage(i, j) {
  this.data[i].queryCounter.all++;
  let data = {
    module: 'hw_remove_extends_query_points',
    group_id: this.idGroup,
    lesson_id: this.$route.params.idLesson,
    point_id: this.data[i].id,
    extend_id: this.data[i].downloadStudentImages[j].id,
  };

  send
    .post('lesson.php', data)
    .then((response) => {
      console.log('Файл успешно удален!');
      this.data[i].queryCounter.done++;
      this.data[i].downloadStudentImages.splice(j, 1);
    })
    .catch((error) => {
      console.log('error');
      console.log(error.response);
      console.log(error);

      let response = error.response.data;
      if (response.error === "deadline_is_expired") {
        this.openModalExpiredPassing();
      }
    });
},

```

```

    }
  });
},

  // переключить отображение контента блоков дз (сворачивание /
  разворачивание пунктов дз)
  switchVisibleMainContent(item) {
    item.show = !item.show;

    if (this.pageHasBeenResized) {
      setTimeout(() => {
        this.addAnimationSlide();
        this.pageHasBeenResized = false;
      }, this.durationSlideDown);
    }
  },

  // Показать / скрыть весь текст
  visibilityAllText(blockName, i) {
    let th = this;
    switch (blockName) {
      case 'important_information':
        th.data[i].showAll = !th.data[i].showAll;
        break;

      case 'answer_comment':
        th.data[i].comments.showAll = !th.data[i].comments.showAll;
        break;

      case 'answer_right':
        th.data[i].rightAnswer.showAll =
!th.data[i].rightAnswer.showAll;
        break;
    }
  },

  // проверка вставки ответа
  checkPastText(e, item, i) {
    this.setNewStatusItem(i, 'none');

    if (!item.isAccessPaste && (e.inputType === 'insertFromPaste')) {
      item.answer = item.answerOld;
    } else {
      item.answerOld = item.answer;
    }
  },

  // скачать файл
  downloadFile(item) {
    let link = document.createElement('a');
    link.setAttribute('href', item.link);
    link.setAttribute('target', '_blank');
    // link.setAttribute('download', item.title);
    link.click();
  },

  // настроить отображение кнопок "Показать все"

```

```

configureVisibleButtonText() {
    let el, elNone, heightEl, heightElNone;

    this.data.forEach((item, i) => {
        if (((item.type === 1) || (item.type === 4)) && item.isChecked) {
            // если это дз с возможностью ответа и оно проверено

            if (item.comments.isShow) {
                // блок с комментариями репетитора к ответу
                el = document.getElementById('answer_comment_' + item.id);
                elNone = document.getElementById('answer_comment_' + item.id
+ '___none');

                heightEl = el.offsetHeight;
                heightElNone = elNone.offsetHeight;

                if (!((heightEl >= (heightElNone - 1)) && (heightEl <=
(heightElNone + 1)))) {
                    this.data[i].comments.showButtonUnfolding = true;
                } else {
                    this.data[i].comments.showButtonUnfolding = false;
                }
            }

            if (item.rightAnswer.isShow) {
                // блок с правильным ответом
                el = document.getElementById('answer_right_' + item.id);
                elNone = document.getElementById('answer_right_' + item.id +
'___none');

                heightEl = el.offsetHeight;
                heightElNone = elNone.offsetHeight;

                if (!((heightEl >= (heightElNone - 1)) && (heightEl <=
(heightElNone + 1)))) {
                    this.data[i].rightAnswer.showButtonUnfolding = true;
                } else {
                    this.data[i].rightAnswer.showButtonUnfolding = false;
                }
            }

            } else if (item.type === 3) { // если это текстовый блок
                el = document.getElementById('important_information_' +
item.id);
                elNone = document.getElementById('important_information_' +
item.id + '___none');

                if (elNone && el) {
                    heightEl = el.offsetHeight;
                    heightElNone = elNone.offsetHeight;

                    if (!((heightEl >= (heightElNone - 1)) && (heightEl <=
(heightElNone + 1)))) {
                        this.data[i].showButtonUnfolding = true;
                    } else {
                        this.data[i].showButtonUnfolding = false;
                    }
                }
            }
        }
    });
}

```

```

    }
  }
})
},

// анимация плавного разворачивания текста
addAnimationSlide() {
  let el, elNone;

  this.data.forEach((item, i) => {
    if (((item.type === 1) || (item.type === 4)) && item.isChecked) {
// если это дз с возможностью ответа и оно проверено

      // блок с комментариями репетитора к ответу
      el = document.getElementById('answer_comment_' + item.id);
      elNone = document.getElementById('answer_comment_' + item.id +
'__none');

      if (el) {
        el.style.maxHeight = elNone.offsetHeight + 'px'
      }

      // блок с правильным ответом
      el = document.getElementById('answer_right_' + item.id);
      elNone = document.getElementById('answer_right_' + item.id +
'__none');

      if (el) {
        el.style.maxHeight = elNone.offsetHeight + 'px'
      }

      this.data[i].show = false;
    } else if ((item.type === 1) || (item.type === 4)) {
      this.data[i].show = false;
    } else if (item.type === 3) { // текстовый блок
      el = document.getElementById('important_information_' +
item.id);
      elNone = document.getElementById('important_information_' +
item.id + '__none');

      if (el) {
        el.style.maxHeight = elNone.offsetHeight + 'px'
      }
    }
  })

  setTimeout(() => {
    this.durationSlideDown = 1000;
  })
},

// возвращает первые 3 изображения, если их > 3-х
getFirstImage(img, count) {
  if (img.length > count) {
    let mass = [];

    for (let i = 0; i < count; i++) {

```

```

        mass.push(img[i])
    }

    return mass
} else {
    return img
}
},

setActiveBlock(i) {
    this.indexActiveImageSlider = i;
},

// открыть галерею изображений для мобильных устройств блок важной
информации
scalePreloadImgMobileInfo(numSlide) {
    this.sliderShow = true;
    this.activeImg = numSlide;

    this.sliderImages = [];

    setTimeout(() => {

this.getMassImagesImportantInfo(this.indexActiveImageSlider).forEach((ite
m, i) => {
        this.sliderImages.push({
            id: i,
            img: item
        });
    })
    }, 1);
},

// открытие галереи изображений на мобильных устройствах
scalePreloadImgMobileRightAnswer(numSlide) {
    this.sliderShow = true;
    this.activeImg = numSlide;

    this.sliderImages = [];

    setTimeout(() => {

this.getMassImagesImportantInfoRightAnswer(this.indexActiveImageSlider).f
orEach((item, i) => {
        this.sliderImages.push({
            id: i,
            img: item
        });
    })
    }, 1);
},

// открыть галерею изображений для мобильных устройств блок с Task
scalePreloadImgMobileTask(numSlide) {
    this.sliderShow = true;
    this.activeImg = numSlide;

```

```

        this.sliderImages = [];

        setTimeout(() => {

this.getMassImagesTask(this.indexActiveImageSlider).forEach((item, i) =>
{
    this.sliderImages.push({
        id: i,
        img: item
    });
})
}, 1);

},

    // открыть галерею изображений для мобильных устройств блок с
комментариями
    scalePreloadImgMobileComment(numSlide) {
        this.sliderShow = true;
        this.activeImg = numSlide;

        this.sliderImages = [];

        setTimeout(() => {

this.getMassImagesComment(this.indexActiveImageSlider).forEach((item, i)
=> {
    this.sliderImages.push({
        id: i,
        img: item
    });
})
}, 1);
},

    // открыть галерею изображений для мобильных устройств блок ответа
пользователя
    scalePreloadDownloadImgMobileTask(numSlide) {
        setTimeout(() => {
            if (!this.isClickOnDeleteBlock) {
                this.sliderShow = true;
                this.activeImg = numSlide;

                this.sliderImages = [];

                setTimeout(() => {

this.getMassDownloadImagesTask(this.indexActiveImageSlider).forEach((item
, i) => {
    this.sliderImages.push({
        id: i,
        img: item.content,
        rotateDegree: item.rotateDegree,
        orientation: item.orientation
    });
})
}, 1);

```

```

    }

    }, 1);
},

// открыть галерею изображений (галерея с ошибками) для мобильных
устройств блок ответа пользователя
openSliderHighlightingImageMobile(numSlide) {
    setTimeout(() => {
        this.showHighlightingImage = true;
        this.activeHighlightingImage = numSlide;
        this.contentHighlightingImage =
this.data[this.indexActiveImageSlider].downloadStudentImages;
    }, 1);
},

setActiveBlockDownloadImages(event, i) {
    this.indexActiveImageSlider = i;
    this.activePoint = i;

    let path = event.path || (event.composedPath &&
event.composedPath());
    let checkEndFunction = false;

    for (let i = 0; i < path.length; i++) {

        if ((String(path[i].nodeName) === 'DIV') && path[i].className &&
(path[i].className.indexOf('fileLoad_image__close_uniqueClass') !== -1))
        {
            checkEndFunction = true;
            break;
        }

    }

    if (!checkEndFunction) { // открыть предпросмотр фото
        if (!this.data[i].isSend && !this.data[i].isChecked) {
            this.sliderType = 'editImages';
        }

        this.isClickOnDeleteBlock = false;
    } else {
        this.isClickOnDeleteBlock = true;
    }
},

// предпросмотр загруженных пользователем изображений
scalePreloadDownloadImg(event, images, i, j) {

    let path = event.path || (event.composedPath &&
event.composedPath());
    let checkEndFunction = false;

    for (let i = 0; i < path.length; i++) {

```

```

        if ((String(path[i].nodeName) === 'DIV') && path[i].className &&
(path[i].className.indexOf('fileLoad_image__close_uniqueClass') !== -1))
    {
        checkEndFunction = true;
        break;
    }

}

if (!checkEndFunction) { // открыть предпросмотр фото
    if (!this.data[i].isSend && !this.data[i].isChecked) {
        this.sliderType = 'editImages';
    }

    this.sliderShow = true;
    this.activeImg = j;
    this.activePoint = i;

    this.sliderImages = [];

    images.forEach((item, i) => {
        this.sliderImages.push({
            id: i,
            img: item.content,
            rotateDegree: item.rotateDegree,
            orientation: item.orientation
        });
    })
}
},

// предпросмотр предзагруженных изображений
scalePreloadImg($event, images, j) {
    this.sliderShow = true;
    this.activeImg = j;

    this.sliderImages = [];

    images.forEach((item, i) => {
        this.sliderImages.push({
            id: i,
            img: item.content
        });
    })
},

// закрыть галерею просмотра фото
closeSlider() {
    this.sliderShow = false;
    this.activeImg = 0;
    this.sliderImages = [];

    if (this.sliderType !== undefined) {
        this.sliderType = undefined;
    }

    if (this.indexActiveImageSlider !== undefined) {

```

```

    this.indexActiveImageSlider = undefined;
  }

  if (this.activePoint !== undefined) {
    this.activePoint = undefined;
  }
},

// преобразует время из формата секунд в формат (чч:мм:сс)
getTime(time) {
  let hours = Math.floor(time / 60 / 60);
  let minutes = Math.floor(time / 60) - (hours * 60);
  let seconds = time - ((hours * 60 * 60) + (minutes * 60));

  if (hours < 10) {
    hours = '0' + hours;
  }

  if (minutes < 10) {
    minutes = '0' + minutes;
  }

  if (seconds < 10) {
    seconds = '0' + seconds;
  }

  return hours + ':' + minutes + ':' + seconds
},

// вернуть размер элемента
getSizeFile(size) {
  let returnSize = Number((size / 1024).toFixed(1));
  let returnText = '';

  if (returnSize >= 1024) {
    returnSize = Number((returnSize / 1024).toFixed(1));

    if (returnSize >= 1024) {
      returnSize = Number((returnSize / 1024).toFixed(1));
      returnText = returnSize + ' Гб'
    } else {
      returnText = returnSize + ' Мб'
    }

  } else {
    returnText = returnSize + ' Кб'
  }

  return returnText
},

// вернуть массив изображений для пунктов дз
getMassImagesTask(i) {
  let images = [];

  this.data[i].attachedImages.forEach((item, j) => {

```

```

        images.push(item.content);
    });

    return images
},

// вернуть массив изображений для пункта дз с комментарием
getMassImagesComment(i) {
    let images = [];

    this.data[i].comments.attachedImages.forEach((item, j) => {
        images.push(item.content);
    });

    return images
},

// вернуть массив загруженных изображений для пунктов дз
getMassDownloadImagesTask(i) {
    let images = [];

    this.data[i].downloadStudentImages.forEach((item, j) => {
        images.push({
            content: item.content,
            rotateDegree: item.rotateDegree,
            orientation: item.orientation
        });
    });

    return images
},

// вернуть массив изображений для пунктов дз
getMassImagesImportantInfo(index) {
    let images = [];

    this.data[index].attachedImages.forEach((item, j) => {
        images.push(item.content);
    });

    return images
},

// вернуть массив изображений для пунктов правильного ответа дз
getMassImagesImportantInfoRightAnswer(index) {
    let images = [];

    this.data[index].rightAnswer.attachedImages.forEach((item, j) => {
        images.push(item.content);
    });

    return images
},

// закрыть модальное окно "Работа заблокирована (дедлайн прошел)"
closeModalExpiredPassing() {
    this.showModalExpiredPassing = false;

```

```

        /*setTimeout(() => {
            this.$router.push({name: 'megarace'});
        }, 500)*/
    },

    // открыть модальное окно "Работа заблокирована (дедлайн прошел)"
    openModalExpiredPassing() {
        this.showModalExpiredPassing = true;
    },

    // загрузить прикрепленный пользователем файл
    downloadUserFile(file, isSend) {
        if (isSend) {
            this.downloadFile(file);
        }
    },

    // переключить отображение блока сохранения работы
    visibleMobileSavingBlock(i) {
        if (window.innerWidth <= 600) {
            this.data[i].savingBlockOpened = !this.data[i].savingBlockOpened;
        }
    },

    // установить новый статус (сохранено, ошибка)
    setNewStatusItem(i, newStatus) {
        if (newStatus === 'inProgress') {
            this.data[i].savingStatus = newStatus;
        } else {
            if (newStatus === 'error') {
                if (!this.showModalWaitingQuery) {
                    if (this.showModalErrorSaving) {
                        this.notification.show = true;
                        this.notification.text = 'Сохранить ответ не удалось,
пробуйте еще раз или обратитесь к <a
href="https://vk.com/im?sel=381434758" target="_blank">техническому
специалисту</a>';
                        this.notification.type = 'error';
                        this.notification.time = 6;
                    } else {
                        this.showModalErrorSaving = true;
                    }
                }
            } else if (this.showModalErrorSaving) {
                this.showModalErrorSaving = false;
            }

            let timeDuration = 1; // время для анимации созхранения

            setTimeout(() => {
                this.data[i].savingStatus = newStatus;
            }, timeDuration * 1000)
        }
    },

    // закрыть модальное окно с информацией об ошибке при сохранении

```

```

closeModalErrorSaving() {
  this.showModalErrorSaving = false;
},

// повторить отправку запроса на сохранения ответа
repeatSendingQuerySaving(store) {
  this.savePartHomework(store.index);
},

// открыть модальное окно "доработать ДЗ"
openRevisionModal(index) {
  this.revisionModal.show = true;
  this.revisionModal.index = index;
},

// открыть модальное окно "доработать ДЗ"
closeRevisionModal() {
  this.revisionModal.show = false;
  this.revisionModal.index = undefined;
},

// доработать пункт
revisionPoint() {
  // удаление тегов в тексте
  let removeTagsText = (textWithTags) => {
    let text = textWithTags.replaceAll("</br>",
"\n").replaceAll("<br>", "\n");

    return text.replace(/<(.|\n)*?>/g, '');
  };

  let data = {
    module: 'start_revision_work',
    lesson_id: this.$route.params.idLesson,
    point_id: this.data[this.revisionModal.index].id
  };

  console.log('hw_get_points');
  console.log(data)

  send
    .post('lesson.php', data)
    .then((response) => {
      this.data[this.revisionModal.index].answer =
removeTagsText(this.data[this.revisionModal.index].answer);
      this.data[this.revisionModal.index].isSend = false;
      this.data[this.revisionModal.index].isChecked = false;
      this.data[this.revisionModal.index].scoreText = '';
      this.data[this.revisionModal.index].textChecked = '';
      this.data[this.revisionModal.index].activeMistake = undefined;
      this.data[this.revisionModal.index].mistake = [];
      this.data[this.revisionModal.index].mistakesByTypes = [];
      this.data[this.revisionModal.index].countMistakes = undefined;
      this.data[this.revisionModal.index].criteria = [];
      this.data[this.revisionModal.index].comments.isShow = false;
      this.data[this.revisionModal.index].comments.showAll = false;
    });
}

```

```

this.data[this.revisionModal.index].comments.showButtonUnfolding = false;
this.data[this.revisionModal.index].comments.text = '';
this.data[this.revisionModal.index].comments.attachedFiles =
[];
this.data[this.revisionModal.index].comments.attachedImages =
[];
this.data[this.revisionModal.index].rightAnswer.isShow = false;
this.data[this.revisionModal.index].rightAnswer.showAll =
false;

this.data[this.revisionModal.index].rightAnswer.showButtonUnfolding =
false;
this.data[this.revisionModal.index].rightAnswer.text = '';
this.data[this.revisionModal.index].rightAnswer.attachedFiles =
[];
this.data[this.revisionModal.index].rightAnswer.attachedImages
= [];

this.closeRevisionModal();
})
.catch((error) => {
  console.log('error');
  console.log(error.response);
  console.log(error);
});
},

// Изменить ответ (до проверки репетитором)
editAnswer(index) {
  this.data[index].isSend = false;
},
},

computed: {},

directives: {
  clickOutside: {
    bind(el, binding) {
      el.addEventListener('click', e => e.stopPropagation());
      document.body.addEventListener('click', binding.value);
    },
    unbind(el, binding) {
      document.body.removeEventListener('click', binding.value);
    }
  }
},

components: {
  Breadcrumbs,
  SlideUpDown,
  DefaultSlider,
  ModalNotification,
  AddFileWithPasteFunction,
  VueAwesomeSwiper,
  Swiper,
  SwiperSlide,

```

```
    HighlightingImageComponent,  
    ExamIsCheckedBlock,  
    WorkDeadlineTimer,  
    modalInfoBlock,  
    LoadingComponent,  
    WaitingLoading,  
    InformationBlock,  
    FilesComponent,  
  }  
}  
</script>
```

Приложение В
(рекомендуемое)
Исходной код серверной части

Листинг В.1 – Модуль получения списка уроков

```

$api->module('get_lessons_data_gk', ['group_id:int', 'level_id:int'],
function ($data) use ($api, $db, $timestamp, $vk_push, $ac) {
    $group = \Platform\Group::create($data['group_id']);
    $course = $group->get_course();
    if ($course->get_year() < 2024 || $course->get_id() == 6189) {
        $api->answer = $api->run_module('groups', 'get_lessons_data', [
            'group_id' => $data['group_id']
        ]);
        return;
    }
    $product_id = $course->get_productId();

    /** @var \Platform\User $user */
    $user = \Library\System::getUser(true);
    $api->answer['deb'] = $product_id;
    if ($product_id === 1) {
        /** @var \Pay\Buy\GK $buy */
        $buy = \Pay\Buy\GK::searchForUserAndCourse($user, $course, true);
    } else if ($product_id === 8) {
        /** @var \Pay\Buy\PGK $buy */
        $buy = \Pay\Buy\PGK::searchForUserAndCourse($user, $course, true);
    } else if ($product_id === 9) {
        /** @var \Pay\Buy\MRT $buy */
        $buy = \Pay\Buy\MRT::searchForUserAndCourse($user, $course, true);
    } else {
        throw new \Library\Exception('Product is invalid');
    }

    $api->answer['group_id'] = (int)$data['group_id'];

    if (!$buy) {
        $api->field_invalid('group_id')->validate();
    }

    if (!$buy->isSetGroup($data['group_id'])) {
        $pt = \Platform\PaydayTeacher::findPaidFromGroup($group, $user);
        if ($pt->empty())
            $api->field_invalid('group_id')->validate();
        $api->answer['is_current_group'] = false;
    } else {
        $api->answer['is_current_group'] = true;
    }

    $level = \Platform\GK\LessonLevel::create($data['level_id']);
    $level_in_group =
    \Platform\GK\LessonLevelInGroup::createExtend($group, $level);

    $level = \Platform\LessonLevel::create($level->get_id());
    $level_game = \Platform\LevelGame::createFrom($group, $level);

    $user_game = new \Platform\UserGame($user, $course, $level_game);

    $is_time_pitstop = false;
    $is_time_boss = false;
    $is_need_passing_pit_stop = false;

```

```

$sis_need_boss_fight = false;

// Если на уровне есть геймификация, тогда высчитываем значение
переменных битвы и питстопа
if ($level->is_game) {
    if ($level_in_group->time_pit_stop_start && $level_in_group-
>time_pit_stop_end) {
        if ($timestamp >= $level_in_group->time_pit_stop_start-
>unixtime() && $timestamp < $level_in_group->time_pit_stop_end-
>unixtime()) {
            $sis_time_pitstop = true;
        }
    }

    if ($sis_time_pitstop && $user_game->getScore() < $level_game-
>passing_score)
        $sis_need_passing_pit_stop = true;

    if ($level_in_group->time_boss_start && $level_in_group-
>time_boss_end) {
        if ($timestamp >= $level_in_group->time_boss_start->unixtime()
&& $timestamp < $level_in_group->time_boss_end->unixtime()) {
            $sis_time_boss = true;
        }
    }

    if ($sis_time_boss && $user_game->getScore() < $level_game-
>passing_score)
        $sis_need_boss_fight = true;
}

$api->answer['lessons'] = gk_get_list_lessons_2024(
    $course->get_id(),
    $group->get_id(),
    $level->id,
    $product_id
);

$api->answer['course_name'] = $course->get_name();
$api->answer['course_type'] = $course->get_type();
$api->answer['course_description'] = $course->get_description();
$api->answer['course_id'] = $course->get_id();
$api->answer['price'] = $buy->get_price()->get_renewalPrice()-
>get_amount();

$api->answer['level'] = [
    'number' => $level_in_group->number,
    'name' => $level->name,
    'is_game' => (bool)$level->is_game,
    'step' => $level_in_group->getCurrentStep(),
    'is_pit_stop' => false,
    'is_boss' => false,
    'time_pit_stop' => [
        'start' => $level_in_group->time_pit_stop_start ?
$level_in_group->time_pit_stop_start->unixtime() : null,
        'start_str' => $level_in_group->time_pit_stop_start ?
$level_in_group->time_pit_stop_start->str_3() : null,

```

```

        'end' => $level_in_group->time_pit_stop_end ? $level_in_group-
>time_pit_stop_end->unixtime() : null,
        'end_str' => $level_in_group->time_pit_stop_end ?
$level_in_group->time_pit_stop_end->str_3() : null
    ],
    'pit_stop' => [],
    'time_boss' => [
        'start' => $level_in_group->time_boss_start ? $level_in_group-
>time_boss_start->unixtime() : null,
        'start_str' => $level_in_group->time_boss_start ?
$level_in_group->time_boss_start->str_3() : null,
        'end' => $level_in_group->time_boss_end ? $level_in_group-
>time_boss_end->unixtime() : null,
        'end_str' => $level_in_group->time_boss_end ? $level_in_group-
>time_boss_end->str_3() : null
    ],
    'boss_fight' => [],
    'score_for_skip_boss' => $level_game->passing_score,
    'score' => $user_game->getScore()
];

if ($is_time_pitstop) {
    $api->answer['level']['is_pit_stop'] = true;
    if ($is_need_passing_pit_stop)
        $api->answer['level']['pit_stop']['banner'] = [
            'img' => f_struct(4970036),
            'header' => 'Приближается время пит-стопа. Необходимо
сдать долги!',
            'text' => [
                'Что такое «Пит-стоп»? Это дополнительное время за
которое есть возможность сдать все долги по учебе и заработать
недостающие баллы.',
                'В случае успешного прохождения пит-стопа, можно
избежать битвы с боссом - финального тестирования.'
            ]
        ];
    else
        $api->answer['level']['pit_stop']['banner'] = [
            'img' => f_struct(4970038),
            'header' => 'Ты отлично справляешься, самое время немного
перевести дух!',
            'text' => [
                'Во время пит-стопа:',
                'Не приходят ТА и интенсивы, а домашние задания
становятся менее объемными.',
                'Можно отдохнуть и набраться сил для дальнейшей
учебы.',
                'Можно потратить накопленные баллы в маркете.'
            ]
        ];

    $api->answer['level']['pit_stop']['is_need_passing'] =
$is_need_passing_pit_stop;
} else if ($is_time_boss) {
    $api->answer['level']['is_boss'] = true;
    $api->answer['level']['boss_fight']['is_need_passing'] =
$is_need_boss_fight;
}

```

```

        $api->answer['level']['boss_fight']['is_access_start_boss_fight']
= $user_game->isAccessStartBossFight($group);
        $api->answer['level']['boss_fight']['is_maximum_count_try'] =
($user_game->bossFightCompleteTryCount() == $user_game-
>bossFightAllTryCount());
    }

    $bonus_list = $user_game->getBonusList();
    /** @var \Platform\Bonus $bonus */
    foreach ($bonus_list as $bonus) {
        $api->answer['level_progress']['bonus'][] = [
            'id' => $bonus->get_id(),
            'name_bonus' => $bonus->name,
            'score' => $bonus->score,
            'image' => $bonus->img->struct(),
            'modal_bonus' => [
                'show_modal' => (bool)$bonus->is_show_modal,
                'image' => $bonus->img_modal ? $bonus->img_modal->struct()
: null,
                'description_text' => $bonus->getDescription(),
                'comment' => $bonus->comment
            ]
        ];
    }

    $user_game->recalculateScore();

});

```

Листинг В.2 – Модуль получения информации об одном уроке

```

$api->module('get_lesson_data', ['lesson_id'], function ($data) use
($api, $db, $timestamp, $vk_push, $c, $ac) {
    $course_type = $ac->is_lesson($data['lesson_id']);

    $is_drain = false;

    if (!$course_type) {
        if (DRAIN_START_TIME <= $timestamp && $timestamp < DRAIN_END_TIME
&& gk_check_drain(null, $data['lesson_id']))
            $is_drain = true;
        else
            $ac->error();
    }

    $info = $db->query("SELECT mci.id                as 'course_id',
                           mci.name                as 'course_name',
                           mci.type                as 'course_type',
                           mci_parent.id           as 'parent_course_id',
                           mci_parent.name         as 'parent_course_name',
                           g.id                    as 'group_id',
                           g.number                as 'group_number',
                           lg.id,
                           lg.name,
                           lg.number,
                           lg.type,
                           lg.time_lesson,
                           lg.time_start,

```

```

lg.time_end          as 'deadline',
lg.max_bal,
lg.score_passing,
lg.video_record_link,
lg.kineskope,
lg.is_broadcast,
lg.time_end_broadcast,
lg.video_record_name,
lg.thumbnail_id,
lg.level,
lg.is_hw,
lu.user_id           as 'is_set_lu',
lu.is_start,
lu.is_sent,
lu.is_deadline,
lu.is_response,
lu.time_sent,
lu.is_copy,
lu.bal,
lu.time_start        as 'u_time_start',
lu.is_revision,
lu.is_passed,
lu.time_response,
lu.time_duration,
lu.time_deadline,
lu.desired_bal,
is_hw.is_set_hw,
lg.timer_mode,
lg.is_hw_diagnostic,
lg.video_record_type,

f_video.id           as 'video_id',
f_video.local_name   as 'video_link',
f_video.typefile     as 'video_type',
f_video.size         as 'video_size',
f_video.dir          as 'video_dir',

preview.id           as 'preview_id',
preview.local_name   as 'preview_link',
preview.typefile     as 'preview_type',
preview.size         as 'preview_size',
preview.dir          as 'preview_dir',

lf.score             as 'feedback_score',
lf.text              as 'feedback_text',
lf.id                as 'feedback_id',
lf.is_send           as 'feedback_is_send',
lf.is_anon           as 'feedback_is_anon',
lf.is_answer_teacher as
'feedback_is_answer_teacher',
lf.teacher_answer as
'feedback_teacher_answer',

lf.time_answer as 'feedback_time_answer',
lf.time_send as 'feedback_time_send',
ui_teacher.id as 'feedback_teacher_id',
ui_teacher.name as 'feedback_teacher_name',
f_ava_teacher.id      as 'fb_t_ava_id',

```

```

f_ava_teacher.local_name as
'fb_t_ava_link',
f_ava_teacher.typefile as
'fb_t_ava_type',
f_ava_teacher.size as
'fb_t_ava_size',
f_ava_teacher.dir as 'fb_t_ava_dir',
reserve.time_end as 'reserve_deadline',
pt.id as 'is_pay',
t.user_id as 'real_teacher_id'
FROM lesson_groups lg
LEFT JOIN LATERAL ( SELECT
MAX(lrt.time_end) as 'time_end', lrt.lesson_id
FROM
lesson_reserve_time lrt
INNER JOIN
lesson_reserve_time_user lrtu
on lrt.id = lrtu.reserve_time_id and lrtu.user_id = ?i
WHERE lrt.lesson_id
= lg.id
GROUP BY
lrt.lesson_id
LIMIT 1) reserve on
reserve.lesson_id = lg.id
INNER JOIN `groups` g on g.id =
lg.group_id and g.is_delete = 0
INNER JOIN main_course_info mci on mci.id
= g.course_id and mci.is_delete = 0
LEFT JOIN main_course_info mci_parent on
mci.parent_id = mci_parent.id
LEFT JOIN lesson_user lu on lg.id =
lu.lesson_id and lu.user_id = ?i and lu.is_delete = 0
LEFT JOIN LATERAL (SELECT
COALESCE(hit.id, hiq.id, hitt.id) as 'is_set_hw', lg.id
FROM (SELECT 1) c
LEFT JOIN
homework_item_text hitt on hitt.lesson_id = lg.id and hitt.is_delete = 0
LEFT JOIN
homework_item_test hit on hit.lesson_id = lg.id and hit.is_delete = 0
LEFT JOIN
homework_item_quest hiq on hiq.lesson_id = lg.id and hiq.is_delete = 0
LIMIT 1) is_hw on
is_hw.id = lg.id
LEFT JOIN files f_video on
lg.video_record = f_video.id
LEFT JOIN files preview on lg.preview_id
= preview.id
LEFT JOIN lesson_feedback lf on lg.id =
lf.lesson_id and lf.user_id = ?i
LEFT JOIN users_info ui_teacher on
ui_teacher.id = lf.teacher_id
LEFT JOIN files f_ava_teacher on
f_ava_teacher.id = ui_teacher.avatar_id
LEFT JOIN payday_teacher pt on
pt.course_id = g.course_id and pt.user_id = ?i and pt.lesson_number =
lg.number

```

```

                                LEFT JOIN teachers t on t.group_id = g.id
and t.is_real = 1
                                WHERE lg.id = ?i
                                and lg.is_delete = 0
                                ORDER BY lg.number
                                LIMIT 1;", [$SESSION['id'], $SESSION['id'],
$SESSION['id'], $SESSION['id'], $data['lesson_id']])>fetch();

    if (empty($info['id']))
        $api->field_invalid('lesson_id')>validate();
    if ((!$info['time_lesson'] || $info['time_lesson'] > $timestamp) &&
$info['course_id'] != 6189)
        $api->error('is_not_start');

    if ($info['course_id'] == 6189 && !$info['is_pay']) {
        $db->exec("INSERT INTO payday_teacher (teacher_id, user_id,
course_id, group_id, pay, real_pay, time, lesson_number, time_payday)
                                VALUE (?i, ?i, ?i, ?i, 1, 1, ?i, ?i, null)",
[$info['real_teacher_id'], $SESSION['id'], $info['course_id'],
        $info['group_id'], $timestamp, $info['number']]);
    }

    if ($info['type'] == 3 && !$info['is_start'])
        $api->error('exam_is_not_start');

    $info['is_hw'] = (bool)$info['is_hw'];
    $info['is_set_hw'] = (bool)$info['is_set_hw'];
    if ($info['is_hw'] != $info['is_set_hw']) {
        $info['is_hw'] = $info['is_set_hw'];
        $db->exec("UPDATE lesson_groups SET is_hw = ?i WHERE id = ?i LIMIT
1",
                [$info['is_set_hw'], $data['lesson_id']]);
    }

    $api->answer['course'] = [
        'id' => (int)$info['course_id'],
        'name' => $info['course_name'],
        'type' => (int)$info['course_type'],
        'parent' => $info['parent_course_id'] ? [
            'id' => (int)$info['parent_course_id'],
            'name' => $info['parent_course_name']
        ] : []
    ];

    $api->answer['group'] = [
        'id' => (int)$info['group_id'],
        'number' => (int)$info['group_number']
    ];

    $api->answer['is_drain'] = $is_drain;

    $is_set_feedback = true;
    $is_open_hw = $info['time_start'] && $info['time_start'] <=
$timestamp;

    $is_complete_lesson = $info['video_id'] || $info['video_record_link']
|| $info['kineskope'] || $is_open_hw ||

```



```

ORDER BY lgv.number, t_code_time
ASC;", [$data['lesson_id'], $info['video_record_type']]);

while ($row = $extra_video_records_q->fetch()) {
    if (empty($extra_video_records[$row['id']]))
        $extra_video_records[$row['id']] = [
            'id' => (int)$row['id'],
            'file' => $row['file_id'] ? f_struct($row['file_id']) :
null,
            'preview' => $row['preview_id'] ?
f_struct($row['preview_id']) : null,
            'name' => $row['name'],
            'link' => $row['link'],
            'is_broadcast' => (bool)$row['is_broadcast'],
            'time_end_broadcast' => $row['time_end_broadcast'] ?
(int)$row['time_end_broadcast'] : null,
            'number' => (int)$row['number'],
            'times_code' => []
        ];

    if ($row['t_code_id'])
        $extra_video_records[$row['id']]['times_code'][] = [
            'id' => (int)$row['t_code_id'],
            'time_code' => (int)$row['t_code_time'],
            'description' => $row['t_code_description']
        ];
}
$extra_video_records = array_values($extra_video_records);

$api->answer['lesson'] = [
    'name' => $info['name'],
    'number' => (int)$info['number'],
    'type' => (int)$info['type'],
    'time_start' => $info['time_lesson'] ? (int)$info['time_lesson'] :
null,
    'time_start_str' => $info['time_lesson'] ?
timestamp_to_text($info['time_lesson'], 3) : null,
    'time_open_hw' => $info['time_start'] ? (int)$info['time_start'] :
null,
    'time_open_hw_str' => $info['time_start'] ?
timestamp_to_text($info['time_start']) : null,
    'is_open_hw' => $is_open_hw,
    'deadline' => $info['deadline'] ? (int)$info['deadline'] : null,
    'deadline_str' => $info['deadline'] ?
timestamp_to_text($info['deadline']) : null,
    'is_deadline_expired' => $info['deadline'] && $info['deadline'] <=
$timestamp,
    'score_passing' => $info['score_passing'] ?
(int)$info['score_passing'] : null,
    'max_bal' => $info['max_bal'] ? (int)$info['max_bal'] : null,
    'passed_data' => [
        'is_start' => (bool)$info['is_start'],
        'time_start' => $info['is_start'] ? (int)$info['u_time_start']
: null,
        'time_start_str' => $info['is_start'] ?
timestamp_to_text($info['u_time_start'], 3) : null,
        'is_sent' => (bool)$info['is_sent'],

```

```

        'time_sent' => $info['is_sent'] ? (int)$info['time_sent'] :
null,
        'time_sent_str' => $info['is_sent'] ?
timestamp_to_text($info['time_sent'], 3) : null,
        'is_response' => (bool)$info['is_response'],
        'time_response' => $info['is_response'] ?
(int)$info['time_response'] : null,
        'time_response_str' => $info['is_response'] ?
timestamp_to_text($info['time_response'], 3) : null,
        'is_copy' => (bool)$info['is_copy'],
        'bal' => (int)$info['bal'],
        'time_duration' => (int)$info['time_duration'],
        'is_revision' => (bool)$info['is_revision'],
        'is_deadline' => (bool)$info['is_deadline']
    ],
    'is_timer' => (bool)($info['timer_mode'] & 1),
    'is_strict' => (bool)($info['timer_mode'] & 2),
    'desired_bal' => is_null($info['desired_bal']) ? null :
(int)$info['desired_bal'],
    'materials' => &$materials,
    'feedback' => [
        'score' => (int)$info['feedback_score'],
        'text' => $info['feedback_text'],
        'tags' => &$feedback_tags,
        'is_send' => (bool)$info['feedback_is_send'],
        'is_anon' => (bool)$info['feedback_is_anon'],
        'is_answer' => (bool)$info['feedback_is_answer_teacher'],
        'text_answer' => $info['feedback_is_answer_teacher'] &&
$info['feedback_teacher_answer'] ? $info['feedback_teacher_answer'] : '',
        'time_answer' => $info['feedback_time_answer'] &&
$info['feedback_is_answer_teacher'] ? (int)$info['feedback_time_answer']
: null,
        'time_answer_str' => $info['feedback_is_answer_teacher'] ?
timestamp_to_text($info['feedback_time_answer'], 4) : null,
        'time_send' => $info['feedback_time_send'] &&
$info['feedback_is_send'] ? (int)$info['feedback_time_send'] : null,
        'time_send_str' => $info['feedback_is_send'] ?
timestamp_to_text($info['feedback_time_send'], 4) : null,
        'teacher' => $info['feedback_teacher_id'] &&
$info['feedback_is_answer_teacher'] ? [
            'id' => (int)$info['feedback_teacher_id'],
            'name' => $info['feedback_teacher_name'],
            'avatar' => file_struct($info['fb_t_ava_id'],
$info['fb_t_ava_size'], $info['fb_t_ava_link'], $info['fb_t_ava_type'],
$info['fb_t_ava_dir'])
        ] : []
    ],
    'is_show_feedback_modal' => $is_complete_lesson &&
!$is_set_feedback,
    'video_record' => file_struct($info['video_id'],
$info['video_size'], $info['video_link'], $info['video_type'],
$info['video_dir'], $info['video_record_name']),
    'video_preview' => file_struct($info['preview_id'],
$info['preview_size'], $info['preview_link'], $info['preview_type'],
$info['preview_dir']),
    'video_record_link' => $info['video_record_link'] ?: null,
    'kinescope' => $info['kinescope'] ?: null,

```

```

        'is_broadcast' => $info['video_record_link'] ?
(bool)$info['is_broadcast'] : false,
        'time_end_broadcast' => $info['time_end_broadcast'] &&
$info['video_record_link'] ? (int)$info['time_end_broadcast'] : null,
        'is_hw_diagnostic' => (bool)$info['is_hw_diagnostic'],
        'extra_video_records' => $info['video_record_type'] == 1 ?
$extra_video_records : [],
        'extra_audio_records' => $info['video_record_type'] == 2 ?
$extra_video_records : [],
        'video_record_type' => (int)$info['video_record_type'],
        'thumbnail' => f_struct($info['thumbnail_id']),
        'level' => (int)$info['level'],
        'is_hw' => $info['is_hw'] && $info['is_set_hw'] && !$is_drain
];

$get_ball_array_from_bin = function ($bin) {
    $arr = [1, 2, 3, 4, 5];
    foreach ($arr as $b => $v)
        if ((2 ** $b & $bin) === 0)
            unset($arr[$b]);
    return array_values($arr);
};

$feedback_q = $db->query("SELECT lft.id,
                                lft.text,
                                lft.bin_bal_opened,
                                icon_f.id                as
'icon_id',
                                icon_f.local_name          as
'icon_link',
                                icon_f.typefile            as
'icon_type',
                                icon_f.size                as
'icon_size',
                                icon_f.dir                 as
'icon_dir',

                                active_icon_f.id           as
'active_icon_id',
                                active_icon_f.local_name   as
'active_icon_link',
                                active_icon_f.typefile     as
'active_icon_type',
                                active_icon_f.size         as
'active_icon_size',
                                active_icon_f.dir           as
'active_icon_dir',

                                lfts.tag_id                as
'is_select'
                                FROM lesson_feedback_tags lft
                                LEFT JOIN files icon_f on
lft.icon_id = icon_f.id
                                LEFT JOIN files active_icon_f on
lft.active_icon_id = active_icon_f.id

```

```

                                LEFT JOIN
lesson_feedback_tags_select lfts on lft.id = lfts.tag_id and
lfts.feedback_id = ?i
                                WHERE lft.is_show = 1 and lft.id is not
null;", [$info['feedback_id']]);
    while ($row = $feedback_q->fetch()) {
        $feedback_tags[] = [
            'id' => (int)$row['id'],
            'text' => $row['text'],
            'icon' => file_struct($row['icon_id'], $row['icon_size'],
$row['icon_link'], $row['icon_type'], $row['icon_dir']),
            'icon_active' => file_struct($row['active_icon_id'],
$row['active_icon_size'], $row['active_icon_link'],
$row['active_icon_type'], $row['active_icon_dir']),
            'is_select' => (bool)$row['is_select'],
            'from_bal' =>
$get_ball_array_from_bin((int)$row['bin_bal_opened'])
        ];
    }
    $materials_q = $db->query("SELECT lm.id          as 'material_id',
                                lm.type           as 'material_type',
                                lm.description     as
'material_description',
                                lm.title          as 'material_title',
                                lm.url            as 'material_url',
                                lm.rotate,
                                f_m.id           as 'fm_id',
                                f_m.local_name   as 'fm_link',
                                f_m.typefile     as 'fm_type',
                                f_m.size         as 'fm_size',
                                f_m.dir          as 'fm_dir'
                                FROM lesson_groups lg
                                LEFT JOIN lesson_materials lm on
lg.id = lm.lesson_id and lm.type in (1, 2, 3)
                                LEFT JOIN files f_m on lm.file_id
= f_m.id
                                WHERE lg.id = ?i
                                AND lg.is_delete = 0;",
[$data['lesson_id']]);

    while ($row = $materials_q->fetch()) {
        if ($row['material_type'] == 1) {
            $materials['links'][] = [
                'id' => (int)$row['material_id'],
                'description' => $row['material_description'],
                'title' => $row['material_title'],
                'url' => $row['material_url'],
                'preview' => file_struct($row['fm_id'], $row['fm_size'],
$row['fm_link'], $row['fm_type'], $row['fm_dir'])
            ];
        } else if ($row['material_type'] == 2) {
            $materials['files'][] = [
                'id' => (int)$row['material_id'],
                'rotate' => (int)$row['rotate'],
                'file' => file_struct($row['fm_id'], $row['fm_size'],
$row['fm_link'], $row['fm_type'], $row['fm_dir'], $row['material_title'])
            ];
        }
    }
}

```

```

    } else if ($row['material_type'] == 3) {
        $materials['plan'][] = [
            'id' => (int)$row['material_id'],
            'rotate' => (int)$row['rotate'],
            'file' => file_struct($row['fm_id'], $row['fm_size'],
            $row['fm_link'], $row['fm_type'], $row['fm_dir'], $row['material_title'])
        ];
    }
}

if (!$is_drain) {
    if (!$info['is_hw'] && !$info['is_set_hw']) {
        if ($original_deadline != $info['deadline'] &&
        $info['course_type'] == 11)
            $info['time_deadline'] = $info['deadline'];
        if (!$info['is_set_lu']) {
            $db->nulling()->exec("INSERT INTO lesson_user (lesson_id,
            user_id, time_start, is_passed, is_sent, is_response, is_deadline,
            time_sent, time_deadline) VALUE (?i, ?i, ?i, 1, 1, 1, 1, ?i, ?i)",
            [$data['lesson_id'], $_SESSION['id'], $timestamp,
            $timestamp, $info['time_deadline']]);
        } else if (!$info['is_sent']) {
            $db->nulling()->exec("UPDATE lesson_user SET
                is_passed = 1,
                is_sent = 1,
                is_response = 1,
                is_deadline = 1,
                time_sent = ?i,
                time_deadline = ?i WHERE lesson_id = ?i and user_id
                = ?i and is_delete = 0",
                [$timestamp, $info['time_deadline'],
                $data['lesson_id'], $_SESSION['id']]);
        }
    }
} else {
    if (!$info['is_set_lu']) {
        $db->nulling()->exec("INSERT INTO lesson_user (lesson_id,
        user_id, time_start) VALUE (?i, ?i, ?i)",
        [$data['lesson_id'], $_SESSION['id'], $timestamp]);
    }
}
});

```

Листинг В.3 – Модуль получения информации домашнего задания к уроку

```

$lesson = \Platform\Lesson::create($data['lesson_id']);
$group = $lesson->group;
$course = $group->get_course();
if ($course->get_year() >= 2024 && $course_type === 1) {
    $level_in_group = $lesson->lesson_level;
    $api->answer['level'] = [];
    if ($level_in_group)
        $api->answer['level'] = [
            'id' => $level_in_group->lesson_level->id,
            'number' => $level_in_group->number
        ];
}

```

```

if ($course_type === 1) {
    $info = $db->query("SELECT g.id,
        g.number,
        mci.id          as 'course_id',
        mci.name        as 'course_name',
        mci.title       as 'course_title',
        mpi.id          as 'product_id',
        lg.number       as 'lesson_number',
        lg.name         as 'lesson_name',
        lg.time_start   as 'lesson_start_time',
        lg.time_end,
        ubc.is_blocked,
        lg.is_hw_diagnostic,
        lg.type,
        lg.timer_time,
        lg.timer_mode,
        lgt.max_bal,
        lg.score_passing,
        lg.level,
        lus.bal          as 'test_score',
        lus.is_start,
        lus.user_id      as 'is_mlu',
        lus.time_deadline,
        lus.is_response,
        lus.time_start,
        lus.is_sent,
        lus.is_passed,
        lus.is_deadline,
        lus.is_copy,
        lus.desired_bal,
        lus.time_duration,
        lus.is_revision,
        reserve.time_end as 'reserve_deadline',
        pt.tariff_bin_id as 'tariff_bin_id',
        pt.id as 'payday_teacher_id'
    FROM `groups` g
        INNER JOIN main_course_info mci on
g.course_id = mci.id and mci.type = 1 and mci.is_delete = 0
        INNER JOIN lesson_groups lg on g.id =
lg.group_id and lg.id = ?i and lg.is_delete = 0
        INNER JOIN main_product_info mpi on
mci.product_id = mpi.id
        LEFT JOIN LATERAL ( SELECT
MAX(lrt.time_end) as 'time_end', lrt.lesson_id
        FROM
        lesson_reserve_time lrt
        INNER JOIN
        lesson_reserve_time_user lrtu
on lrt.id = lrtu.reserve_time_id and lrtu.user_id = ?i
        WHERE lrt.lesson_id =
lg.id
        GROUP BY
lrt.lesson_id
        LIMIT 1) reserve on
reserve.lesson_id = lg.id

```

```

INNER JOIN users_buy_course ubc on mci.id
= ubc.course_id and ubc.user_id = ?i and ubc.is_delete = 0
LEFT JOIN payday_teacher pt
on mci.id = pt.course_id and
pt.user_id = ubc.user_id and pt.lesson_number = lg.number
LEFT JOIN lesson_user lus on lg.id =
lus.lesson_id and lus.user_id = ubc.user_id and lus.is_delete = 0
LEFT JOIN lesson_groups_tariff lgt on
lg.id = lgt.lesson_id and lgt.tariff_bin_id = pt.tariff_bin_id
WHERE g.is_delete = 0
LIMIT 1", [$data['lesson_id'], $_SESSION['id'],
$_SESSION['id']]));
$info = $info->fetch();
}
else {
$info = $db->query("SELECT g.id,
g.number,
mci.id          as 'course_id',
mci.name        as 'course_name',
mci.title       as 'course_title',
lg.number       as 'lesson_number',
lg.name         as 'lesson_name',
lg.time_start   as 'lesson_start_time',
lg.time_end,
ubc.is_blocked,
lg.is_hw_diagnostic,
lg.type,
lg.timer_time,
lg.timer_mode,
0              as 'max_bal',
lg.score_passing,
lg.level,
lus.bal        as 'test_score',
lus.is_start,
lus.user_id    as 'is_mlu',
lus.time_deadline,
lus.is_response,
lus.time_start,
lus.is_sent,
lus.is_passed,
lus.is_deadline,
lus.is_copy,
lus.desired_bal,
lus.time_duration,
lus.is_revision,
reserve.time_end as 'reserve_deadline',
lgt.tariff_bin_id as 'tariff_bin_id'
FROM `groups` g
INNER JOIN main_course_info mci on g.course_id = mci.id and
mci.type = 1 and mci.is_delete = 0
INNER JOIN lesson_groups lg on lg.id = ?i and lg.is_delete = 0
LEFT JOIN LATERAL ( SELECT
MAX(lrt.time_end) as 'time_end', lrt.lesson_id
FROM
lesson_reserve_time lrt
INNER JOIN
lesson_reserve_time_user lrtu

```

```

on lrt.id = lrtu.reserve_time_id and lrtu.user_id = ?i
                                                    WHERE lrt.lesson_id =
lg.id
                                                    GROUP BY
lrt.lesson_id
                                                    LIMIT 1) reserve on
reserve.lesson_id = lg.id
    INNER JOIN users_buy_course ubc on mci.id = ubc.course_id and
ubc.user_id = ?i and ubc.is_delete = 0
    LEFT JOIN lesson_user lus on lg.id = lus.lesson_id and lus.user_id =
ubc.user_id and lus.is_delete = 0
    LEFT JOIN lesson_groups_tariff lgt on lg.id = lgt.lesson_id
WHERE g.is_delete = 0
LIMIT 1", [$data['lesson_id'], $_SESSION['id'], $_SESSION['id']])-
>fetch();
    }

    if (empty($info['id']))
        $api->field_invalid('group_id')->field_invalid('lesson_id')-
>validate();

    if ($course_type == 1 && $info['product_id'] == 9 )
        $tariff_bin_ids = 16;
    else
        $tariff_bin_ids = (int)$info['tariff_bin_id'];

    if ($info['lesson_start_time'] > $timestamp)
        $api->error('lesson_not_public');

    if ($info['is_hw_diagnostic'] && !$info['is_start'])
        $api->error('is_not_start');

    if ($info['type'] == 3 || $info['is_hw_diagnostic'])
        $info['result_score'] =
BAL_EGE_TO_RESULT[$info['course_id']][$info['test_score']] ?? 0;

    if (!$info['is_start']) {
        if (!$info['is_mlu'])
            $db->exec("INSERT IGNORE INTO lesson_user (lesson_id,
user_id, time_start) VALUE (?i, ?i, ?i)", [$data['lesson_id'],
$_SESSION['id'], $timestamp]);
        else
            $db->exec("UPDATE lesson_user SET time_start = ?i,
is_start = 1 WHERE lesson_id = ?i and user_id = ?i and is_delete = 0",
[$timestamp, $data['lesson_id'], $_SESSION['id']]);
            $info['time_start'] = $timestamp;
    }

    if ($info['time_deadline'])
        $info['time_end'] = $info['time_deadline'];
    if ($info['reserve_deadline'] && $info['reserve_deadline'] >
$info['time_end'])
        $info['time_end'] = (int)$info['reserve_deadline'];

    $is_active_timer = (bool)($info['timer_mode'] & 1);
    $timer_is_strict = (bool)($info['timer_mode'] & 2);

```

```

if ($is_active_timer && !$info['is_start'] && $timer_is_strict) {
    $api->error('is_not_start');
}

$api->answer['course'] = [
    'id' => (int)$info['course_id'],
    'name' => $info['course_name'],
    'title' => $info['course_title']
];

$api->answer['lesson'] = [
    'number' => (int)$info['lesson_number'],
    'name' => $info['lesson_name'],
    'deadline' => (int)$info['time_end'],
    'deadline_str' => $info['time_end'] ?
timestamp_to_text($info['time_end']) : '',
    'deadline_str2' => $info['time_end'] ? date('d.m.Y H:i',
$info['time_end']) : '',
    'level' => (int)$info['level']
];

$api->answer['group'] = [
    'number' => (int)$info['number']
];

$api->answer['is_blocked'] = (bool)$info['is_blocked'];
$api->answer['timestamp'] = $timestamp;

if ($is_active_timer) {
    $time_end_timer = $info['timer_time'] + $info['time_start'];
    $timer_time_now = $timer_is_strict ? $timestamp :
($info['time_start'] + $info['time_duration']);
    $left_time = $time_end_timer > $timer_time_now ?
$time_end_timer - $timer_time_now : 0;
    if ($left_time == 0 && !$info['is_sent']) {
        $info['is_sent'] = true;
        close_exam($data['lesson_id'], 1);
    }
}

//          if (!$info['max_bal'] || true)
//          $info['max_bal'] =
(int)getMaxBalForLesson($data['lesson_id'], $data['group_id'], true);
$info['max_bal'] = (int)$info['max_bal'];

$info['score_passing'] = (int)$info['score_passing'];

$api->answer['gk_info'] = [
    'type' => (int)$info['type'],
    'time_start' => $info['lesson_start_time'] ?
(int)$info['lesson_start_time'] : null,
    'time_start_str' => $info['lesson_start_time'] ?
timestamp_to_text($info['lesson_start_time']) : null,
    'deadline' => $info['time_end'] ? (int)$info['time_end'] :
null,
    'deadline_str' => $info['time_end'] ?
timestamp_to_text($info['time_end']) : '',

```

```

        'isExpired' => $info['time_end'] ? $timestamp >=
$info['time_end'] : null,
        'group_id' => (int)$data['group_id'],
        'timer' => [
            'time' => (int)$info['timer_time'],
            'is_active' => $is_active_timer,
            'is_strict' => $timer_is_strict,
            'time_left' => $is_active_timer ? $left_time : null,
            'timestamp_stop' => $is_active_timer ? $time_end_timer :
null
        ],
        'pass' => [
            'is_sent' => (bool)$info['is_sent'],
            'is_passed' => (bool)$info['is_passed'],
            'is_response' => (bool)$info['is_response'],
            'is_deadline' => (bool)$info['is_deadline'],
            'time_start' => (int)$info['time_start'],
            'is_revision' => (bool)$info['is_revision']
        ],
        'score_passing' => &$info['score_passing'],
        'max_bal' => &$info['max_bal'],
        'written_off' => (bool)$info['is_copy'],
        'is_hw_diagnostic' => (bool)$info['is_hw_diagnostic']
    ];

    if (!$info['score_passing'])
        $is_set_passed_score = false;

    if (($info['type'] == 3 || $info['is_hw_diagnostic']) &&
$info['is_sent']) {
        if (!$info['is_response'])
            $api->error('exam_not_checked');
        if ($timestamp <= $info['time_end'] &&
!$info['time_deadline']) {
            $is_check_not_all = (bool)$db->query("SELECT 1
                                                    FROM lesson_groups lg
                                                    INNER JOIN
payday_teacher pt on lg.group_id = pt.group_id and lg.number =
pt.lesson_number and pt.is_delete = 0
                                                    LEFT JOIN
lesson_user lu on lg.id = lu.lesson_id and lu.user_id = pt.user_id and
pt.is_delete = 0
                                                    WHERE lg.id = ?i
                                                    and (lu.user_id is
null or lu.is_response = 0)
                                                    LIMIT 1;",
[$data['lesson_id']])->fetchColumn(0);
            if ($is_check_not_all)
                $api->error('exam_not_checked');
        }
    }

    if ($info['is_response']) {
        if ($info['test_score'] > $info['max_bal'])
            $info['max_bal'] = getMaxBalForLesson($data['lesson_id'],
$data['group_id'], true);
        if ($info['type'] == 3 || $info['is_hw_diagnostic'])

```

```

        $api->answer['exam_score'] = [
            'test_bal' => (double)$info['test_score'],
            'result_bal' => (int)$info['result_score'],
            'desired_bal' => (int)$info['desired_bal']
        ];
    else
        $api->answer['exam_score'] = [
            'test_bal' => (int)$info['test_score']
        ];
    }

while ($row = $list_q_text_test->fetch()) {
    if ($row['text_id']) { // Заполняем инфу о тексте
        $id = "1-$row[text_id]";
        if (empty($hw[$id]))
            $hw[$id] = [
                'id' => (int)$row['text_id'],
                'type' => 1,
                'number' => (int)$row['text_number'],
                'text' => $row['text_text'],
                'header' => $row['text_header'],
                'extends' => []
            ];

            if ($row['text_ex_id'] &&
empty($hw[$id]['extends'][$row['text_ex_id']]))
                $hw[$id]['extends'][$row['text_ex_id']] = [
                    'id' => (int)$row['text_ex_id'],
                    'max_count_opening' =>
(int)$row['text_f_max_count_opening'],
                    'count_opening' => (int)$row['text_f_count_opening'],
                    'rotate' => (int)$row['text_f_rotate'],
                    'file' => file_struct($row['text_f_id'],
$row['text_f_size'], $row['text_f_link'], $row['text_f_type'],
$row['text_f_dir'], $row['text_f_name'])
                ];
        }

        if ($row['test_id']) { // Заполняем инфу о тестовых
            $id = "2-$row[test_id]";
            if (empty($hw[$id]))
                $hw[$id] = [
                    'id' => (int)$row['test_id'],
                    'type' => 2,
                    'number' => (int)$row['test_number'],
                    'text' => $row['test_text'],
                    'header' => $row['test_header'],
                    'is_deadline' => (bool)$row['test_is_deadline'],
                    'deadline' => $row['test_is_deadline'] ?
(int)$row['test_deadline'] : null,
                    'deadline_str' => $row['test_is_deadline'] ? date('d.m.Y
H:i', $row['test_deadline']) : '',
                    'tests' => [],
                    'extends' => []
                ];
        }
    }
}

```

```

        if ($row['test_ex_id'] &&
empty($hw[$id]['extends'][$row['test_ex_id']]))
        $hw[$id]['extends'][$row['test_ex_id']] = [
            'id' => (int)$row['test_ex_id'],
            'max_count_opening' =>
(int)$row['test_f_max_count_opening'],
            'count_opening' => (int)$row['test_f_count_opening'],
            'rotate' => (int)$row['test_f_rotate'],
            'file' => file_struct($row['test_f_id'],
$row['test_f_size'], $row['test_f_link'], $row['test_f_type'],
$row['test_f_dir'], $row['test_f_name'])
        ];

        if ($row['test_test_id'] &&
empty($hw[$id]['tests'][$row['test_test_id']])) {
            $score_passing = $row['test_test_score_passing'] ??
$row['local_score_passing'];
            $is_passed_test = $row['test_test_is_passed'] ? $score_passing
<= $row['test_test_score'] : null;
            if (!is_null($is_passed_test) && $is_passed_test !=
$row['test_test_is_passed_user']) {
                $db->exec("UPDATE reports_tests SET is_passed = ?i WHERE
id = ?i", [$is_passed_test, $row['test_test_is_passed']]);
            }

            $hw[$id]['tests'][$row['test_test_id']] = [
                'id' => (int)$row['test_test_id'],
                'name' => $row['test_test_name'],
                'middle_time' => (int)$row['test_test_middle_time_dur'],
                'is_passed' => (bool)$row['test_test_is_passed'],
                'score' => $row['test_test_is_passed'] ?
(int)round($row['test_test_score']) : null,
                'score_max' => $row['real_test_test_score_max'] ?
(int)$row['real_test_test_score_max'] : null,
                'score_passing' => !is_null($score_passing) ?
(int)$score_passing : null,
                'is_passing' => $is_passed_test
            ];

            if (!$is_set_passed_score && $score_passing)
                $info['score_passing'] += $score_passing;
        }
    }

    getAllQueryPoints($_SESSION['id'], $data['lesson_id'], $info['time_end'],
$hw, $null, false, $tariff_bin_ids);

    if (!$is_set_passed_score)
        foreach ($hw as $quest)
            if (($quest['type'] === 3 || $quest['type'] === 4) &&
$quest['score_passing'])
                $info['score_passing'] += $quest['score_passing'];

    $hw = array_values($hw);

```

Листинг В.4 – Модуль создания нового пункта домашнего задания

```

$api->module('md_hw_remove_point', ['lesson_id:int', 'group_id:int',
'type:int', 'point_id:int'], function ($data) use ($api, $db, $ac) {
    $ac->teacher_lesson($data['lesson_id'])->check();
    $course_type = $ac->lesson_to_courseType($data['lesson_id']);

    if ($data['type'] < 1 || $data['type'] > 4)
        $api->field_invalid('type')->validate();

    $lesson = \Platform\Lesson::create($data['lesson_id']);
    $lesson_level = null;
    if ($lesson->getLessonLevel())
        $lesson_level = $lesson->getLessonLevel()->lesson_level;

    $info = $db->query("SELECT g.id,
                           hit_tx.id as 'hit_tx_id',
                           hit_ts.id as 'hit_ts_id',
                           hiq.id as 'hiq_id'
                        FROM `groups` g
                        INNER JOIN lesson_groups lg on g.id =
lg.group_id and lg.is_delete = 0 and lg.id = ?i
                        LEFT JOIN homework_item_text hit_tx on
lg.id = hit_tx.lesson_id and hit_tx.id = ?i and hit_tx.is_delete = 0
                        LEFT JOIN homework_item_test hit_ts on
lg.id = hit_ts.lesson_id and hit_ts.id = ?i and hit_ts.is_delete = 0
                        LEFT JOIN homework_item_quest hiq on
lg.id = hiq.lesson_id and hiq.id = ?i and hiq.is_delete = 0
                        INNER JOIN teachers t on g.id =
t.group_id and t.user_id = ?i
                        WHERE g.id = ?i
                        and g.is_delete = 0 LIMIT 1",
[$data['lesson_id'], $data['point_id'], $data['point_id'],
$data['point_id'], $_SESSION['id'], $data['group_id']])->fetch();

    if (empty($info['id']))
        $api->field_invalid('group_id')->field_invalid('lesson_id')->validate();

    if ($data['type'] == 1 and $info['hit_tx_id'])
        $db->query("UPDATE homework_item_text SET is_delete = 1 WHERE id =
?i", [$info['hit_tx_id']]);
    else if ($data['type'] == 2 and $info['hit_ts_id']) {
        $db->query("UPDATE homework_item_test SET is_delete = 1 WHERE id =
?i", [$info['hit_ts_id']]);
        if ($lesson->getGroup()->isFlagman() && $lesson_level) {
            $lesson_level->max_score = $lesson_level->max_score - 4;
        }
    } else if (($data['type'] == 3 || $data['type'] == 4) and
$info['hiq_id']) {
        $db->query("UPDATE homework_item_quest SET is_delete = 1 WHERE id
= ?i", [$info['hiq_id']]);

        $db->exec("UPDATE lesson_user lu
                  SET lu.is_revision = (SELECT BIT_OR(hrq.is_revision)
                                         FROM lesson_groups lg
                                         INNER JOIN
homework_item_quest hiq on lg.id = hiq.lesson_id and hiq.is_delete = 0

```

```

                                INNER JOIN
homework_response_quest hrq
                                on hiq.id =
hrq.item_id and hrq.is_delete = 0 and hrq.user_id = lu.user_id
                                WHERE lu.lesson_id = lg.id
                                and lg.is_delete = 0
                                GROUP BY lg.id
                                LIMIT 1)
WHERE lu.lesson_id = ?i
and lu.is_delete = 0;", [$data['lesson_id']]);

if ($lesson->getGroup()->isFlagman() && $lesson_level) {
    if ($data['type'] == 3)
        $lesson_level->max_score = $lesson_level->max_score - 3;
    elseif ($data['type'] == 4)
        $lesson_level->max_score = $lesson_level->max_score - 3;
}

} else
    $api->field_invalid('type')->field_invalid('point_id')->
    validate();
});

```

Листинг В.5 – Модуль добавления критерия для задания

```

$api->module('md_hw_add_criteria_in_quest', ['lesson_id:int',
'group_id:int', 'point_id:int'], function ($data) use ($api, $db, $ac) {
    $ac->access(['teacher'], 'set')->check();
    $course_type = $ac->lesson_to_courseType($data['lesson_id']);

    $info = $db->query("SELECT g.id,
                        g.is_flagman,
                        hiq.id as 'hiq_id',
                        hiq.parent_id
                        FROM `groups` g
                        INNER JOIN lesson_groups lg on g.id =
lg.group_id and lg.is_delete = 0 and lg.id = ?i
                        INNER JOIN homework_item_quest hiq on
lg.id = hiq.lesson_id and hiq.id = ?i and hiq.is_delete = 0
                        INNER JOIN teachers t on g.id =
t.group_id and t.user_id = ?i
                        WHERE g.id = ?i
                        and g.is_delete = 0 LIMIT 1",
[$data['lesson_id'], $data['point_id'], $_SESSION['id'],
$data['group_id']])->fetch();
    if (empty($info['id']))
        $api->field_invalid('group_id')->field_invalid('lesson_id')->
        field_invalid('point_id')->validate();
    if (!$info['is_flagman'] && $info['parent_id'])
        $api->error('is_not_flagman');

    $db->query("INSERT INTO homework_item_quest_criteria (item_id, number,
bal, text)
                (SELECT ?i, COALESCE(MAX(hiqc.number), 0) + 1, 0, ''
                FROM (SELECT 1) e
                LEFT JOIN homework_item_quest_criteria hiqc on
hiqc.item_id = ?i
                LIMIT 1)", [$data['point_id'], $data['point_id']]);

```

```

    $api->answer['criteria_id'] = (int)$db->lastInsertId();
    $api->answer['criteria_prefix'] = 'K';
    $api->answer['criteria_number'] = (int)$db->query("SELECT number FROM
homework_item_quest_criteria WHERE id = ?i LIMIT 1", [$api-
>answer['criteria_id']])->fetchColumn(0);

    if ($course_type === 1)
        $db->exec("INSERT INTO homework_item_quest_criteria (item_id,
number, bal, text, parent_id)
                SELECT hiq.id, ?i, 0, '', ?i FROM homework_item_quest
hiq WHERE hiq.parent_id = ?i", [$api->answer['criteria_number'], $api-
>answer['criteria_id'], $data['point_id']]);
});

```

Листинг В.6 – Модуль добавления теста в домашнее задание

```

$api->module('md_hw_add_test_in_point', ['lesson_id:int', 'group_id:int',
'point_id:int', 'test_id:int', '?score_passing:int'], function ($data)
use ($api, $db, $ac) {
    $ac->access(['teacher'], 'set')->check();
    $course_type = $ac->lesson_to_courseType($data['lesson_id']);

    if (!isset($data['score_passing']))
        $data['score_passing'] = null;

    if ($course_type === 11) {
        $info = $db->query("SELECT g.id,
                                g.is_flagman,
                                hit_ts.parent_id,
                                hitt.id_test as 'is_set_test',
                                tests.max_bal,
                                tests.score_passing
                                FROM `groups` g
                                INNER JOIN lesson_groups lg on g.id =
lg.group_id and lg.is_delete = 0 and lg.id = ?i
                                INNER JOIN homework_item_test hit_ts
on lg.id = hit_ts.lesson_id and hit_ts.id = ?i and hit_ts.is_delete = 0
                                INNER JOIN teachers t on g.id =
t.group_id and t.user_id = ?i
                                INNER JOIN main_course_info mci on
g.course_id = mci.id
                                INNER JOIN tests on (tests.course_id
= g.course_id or tests.course_id = mci.parent_id) and tests.is_delete = 0
and tests.id = ?i
                                LEFT JOIN homework_item_test_tests
hitt on hit_ts.id = hitt.id_item and hitt.id_test = tests.id
                                WHERE g.id = ?i
                                and g.is_delete = 0 LIMIT 1",
[$data['lesson_id'], $data['point_id'], $_SESSION['id'],
$data['test_id'], $data['group_id']])->fetch();
    } else {
        $info = $db->query("SELECT g.id,
                                g.is_flagman,
                                hit_ts.parent_id,
                                hitt.id_test as 'is_set_test',
                                tests.max_bal,
                                tests.score_passing
                                FROM `groups` g

```

```

                                INNER JOIN lesson_groups lg on g.id =
lg.group_id and lg.is_delete = 0 and lg.id = ?i
                                INNER JOIN homework_item_test hit_ts
on lg.id = hit_ts.lesson_id and hit_ts.id = ?i and hit_ts.is_delete = 0
                                INNER JOIN teachers t on g.id =
t.group_id and t.user_id = ?i
                                INNER JOIN tests on tests.course_id =
g.course_id and tests.is_delete = 0 and tests.id = ?i
                                LEFT JOIN homework_item_test_tests
hitt on hit_ts.id = hitt.id_item and hitt.id_test = tests.id
                                WHERE g.id = ?i
                                and g.is_delete = 0 LIMIT 1",
[$data['lesson_id'], $data['point_id'], $_SESSION['id'],
$data['test_id'], $data['group_id']])->fetch();
    }

    if (empty($info['id']))
        $api->field_invalid('group_id')->field_invalid('lesson_id')-
>field_invalid('point_id')->field_invalid('test_id')->validate();
    if ($info['is_set_test'])
        $api->error('is_set_test');
    if (!$info['is_flagman'] && $info['parent_id'])
        $api->error('is_not_flagman');

    $db->nulling()->query("INSERT INTO homework_item_test_tests (id_item,
id_test, score_passing) VALUE (?i, ?i, ?i)", [$data['point_id'],
$data['test_id'], $data['score_passing']]);

    if ($course_type === 1)
        $db->exec("INSERT INTO homework_item_test_tests (id_item, id_test,
score_passing) SELECT id, ?i, ?i FROM homework_item_test WHERE is_delete
= 0 and parent_id = ?i", [$data['test_id'], $data['score_passing'],
$data['point_id']]);

    $api->answer['local_score_passing'] =
(int)(!is_null($data['score_passing']) ? $data['score_passing'] :
$info['score_passing']);
    $api->answer['max_bal'] = (int)$info['max_bal'];
    $api->answer['score_passing'] = (int)$info['score_passing'];
});

```

Листинг В.7 – Модуль отправки оценки репетитора на ответ ученика

```

$api->module('md_send_feedback_answer_hw', ['group_id:int',
'lesson_id:int', 'point_id:int', 'user_id:int', 'is_send:bool'], function
($data) use ($api, $db, $timestamp, $ac) {
    $ac->set()->teacher($data['group_id']->check());
    if ($ac->lesson_to_courseType($data['lesson_id']) === 10)
        $ac->check_lesson_fe($data['lesson_id']);
    if (!$ac->is_lesson($data['lesson_id'], $data['user_id']))
        $api->field_invalid('group_id')->field_invalid('lesson_id')-
>field_invalid('point_id')->field_invalid('user_id')->validate();

    $info = $db->query("SELECT hrq.id as 'answer_id',
                                hrq.is_teacher_feedback as
'is_send_feedback'
                                FROM `groups` g

```

```

                                INNER JOIN lesson_groups lg on g.id =
lg.group_id and lg.is_delete = 0 and lg.id = ?i
                                INNER JOIN homework_item_quest hiq on
lg.id = hiq.lesson_id and hiq.is_delete = 0 and hiq.id = ?i
                                INNER JOIN homework_response_quest hrq on
hiq.id = hrq.item_id and hrq.user_id = ?i and hrq.is_delete = 0
                                WHERE g.id = ?i
                                    and g.is_delete = 0
                                    LIMIT 1", [$data['lesson_id'], $data['point_id'],
$data['user_id'], $data['group_id']])->fetch();
    if (empty($info['answer_id']))
        $api->field_invalid('group_id')->field_invalid('lesson_id')-
>field_invalid('point_id')->field_invalid('user_id')->validate();

    if ($info['is_send_feedback'] != $data['is_send'])
        $db->query("UPDATE homework_response_quest SET is_teacher_feedback
= ?i, time_teacher_feedback = ?i WHERE id = ?i", [$data['is_send'],
$timestamp, $info['answer_id']]);
});

```

Приложение Г
(рекомендуемое)
Тестирование приложения

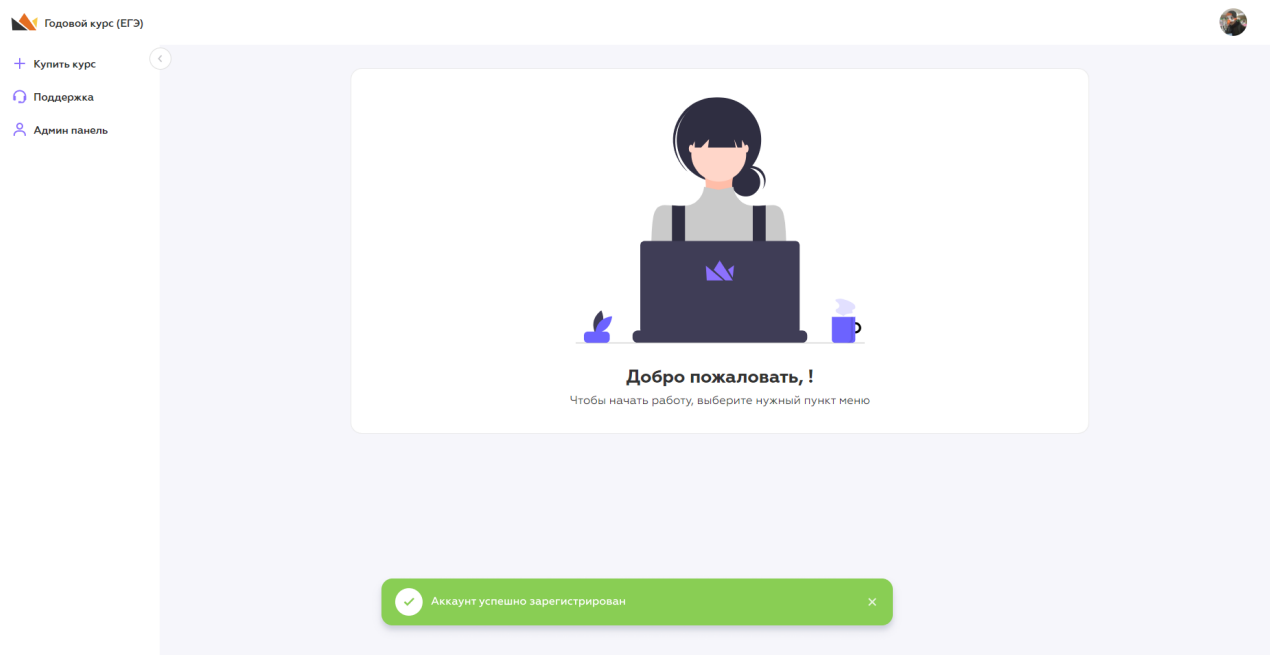


Рисунок Г.1 – Аккаунт зарегистрирован

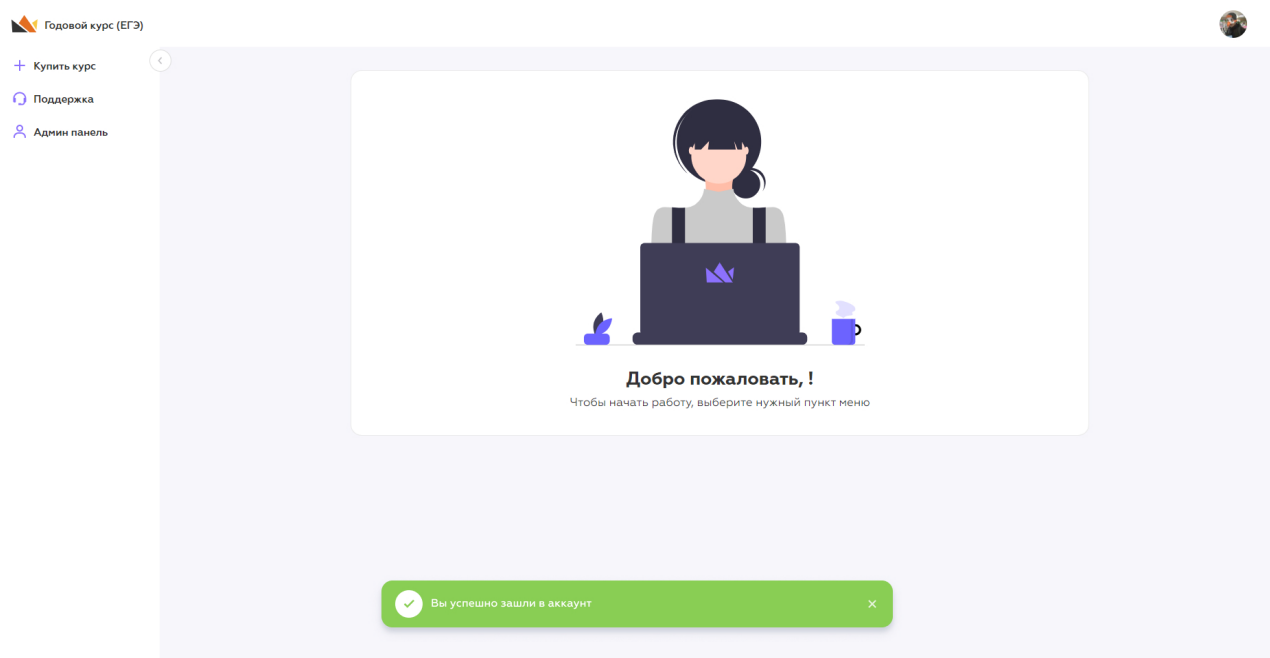


Рисунок Г.2 – Вход в аккаунт

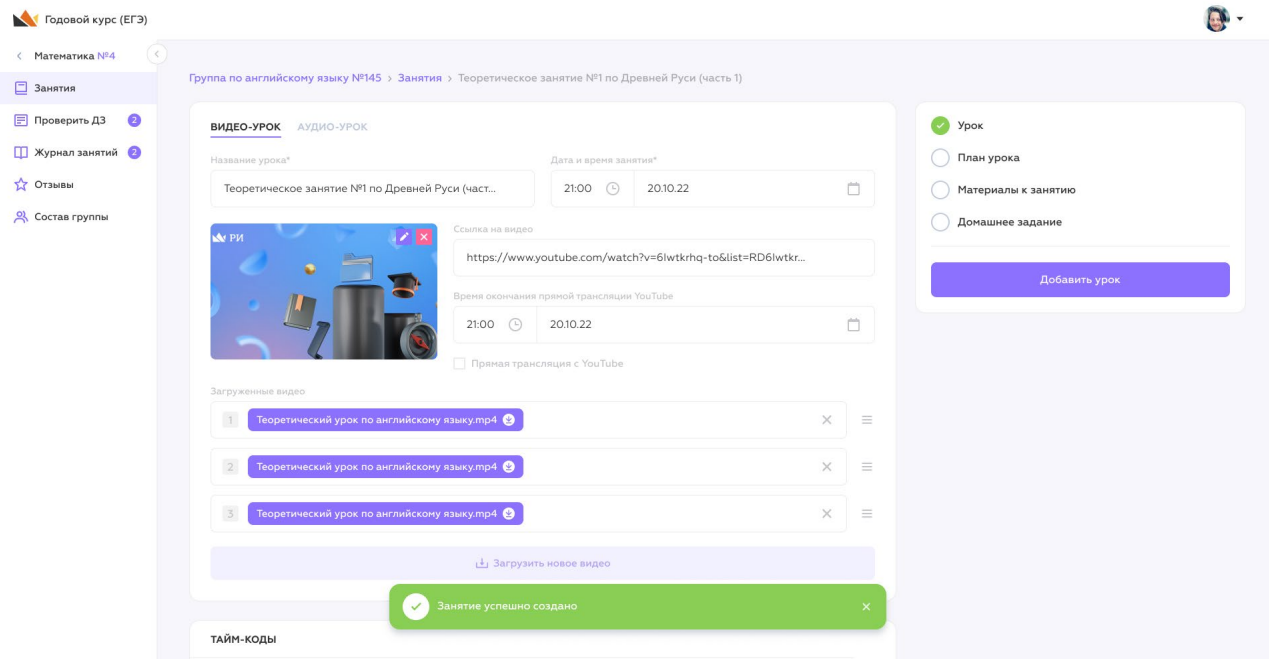


Рисунок Г.3 – Занятие создано

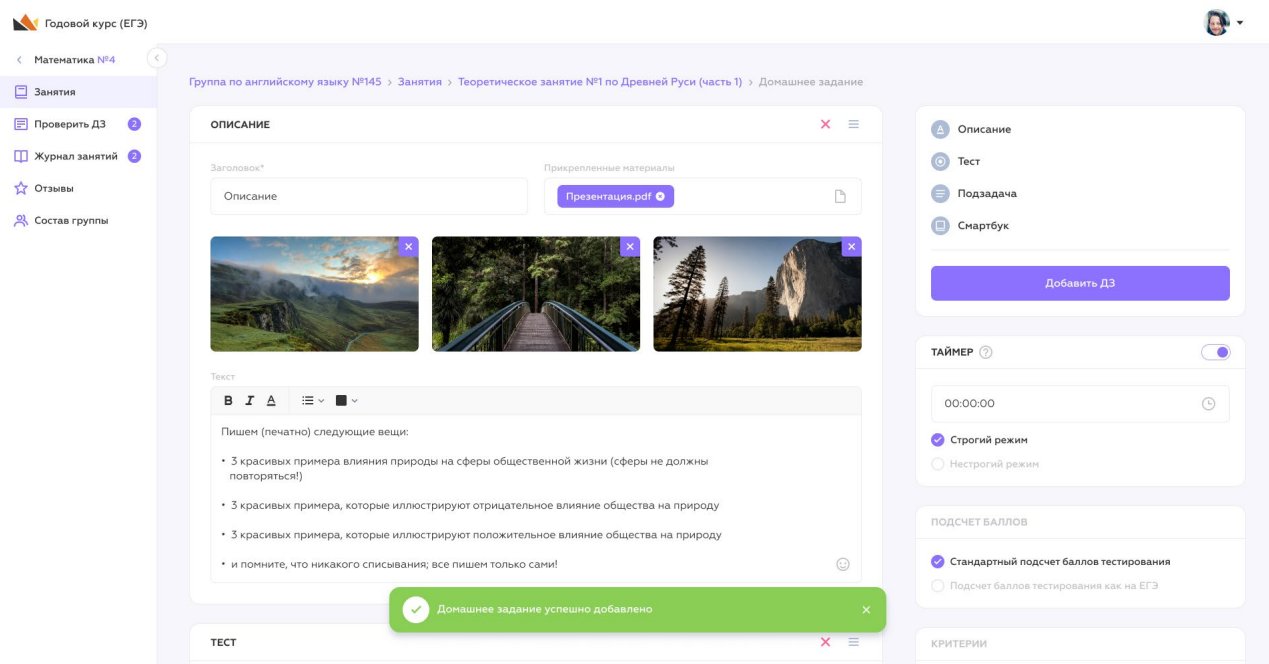


Рисунок Г.4 – ДЗ прикреплено

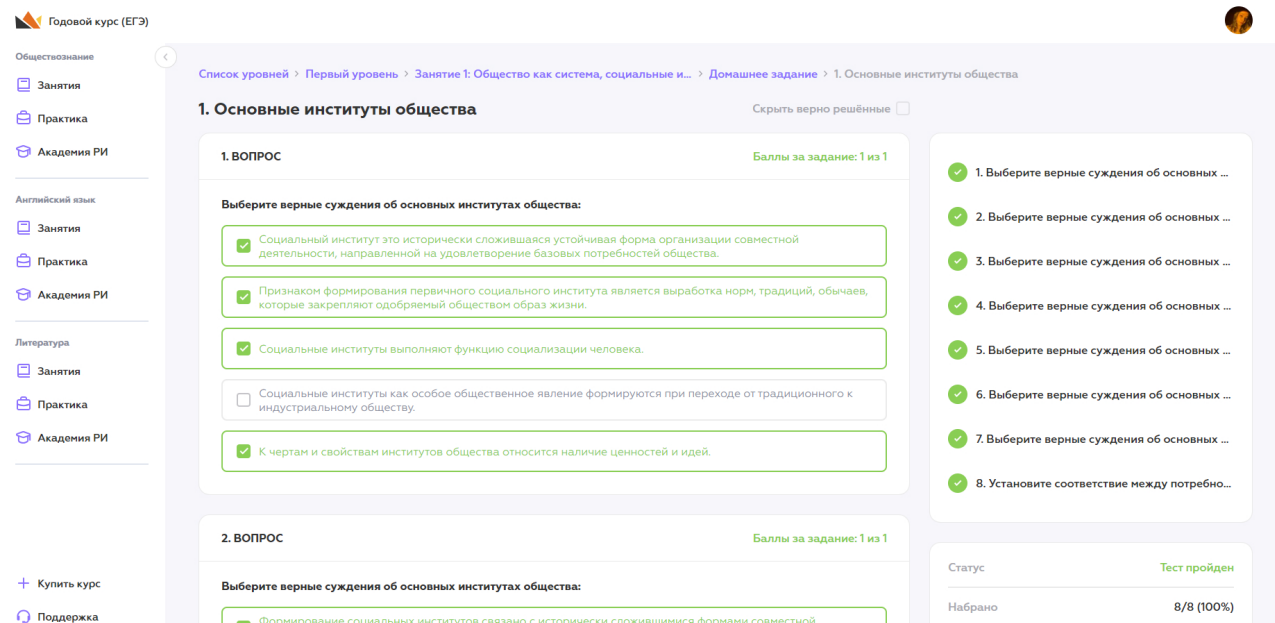


Рисунок Г.5 – ДЗ выполнено

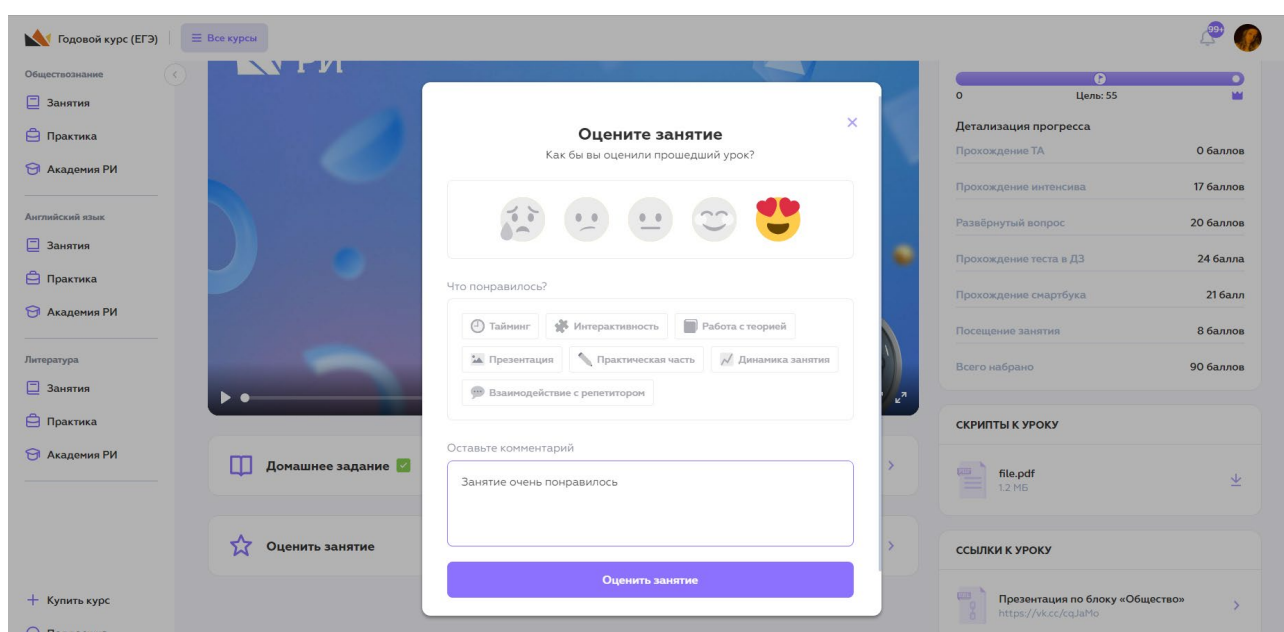


Рисунок Г.6 – Оценка качества занятия

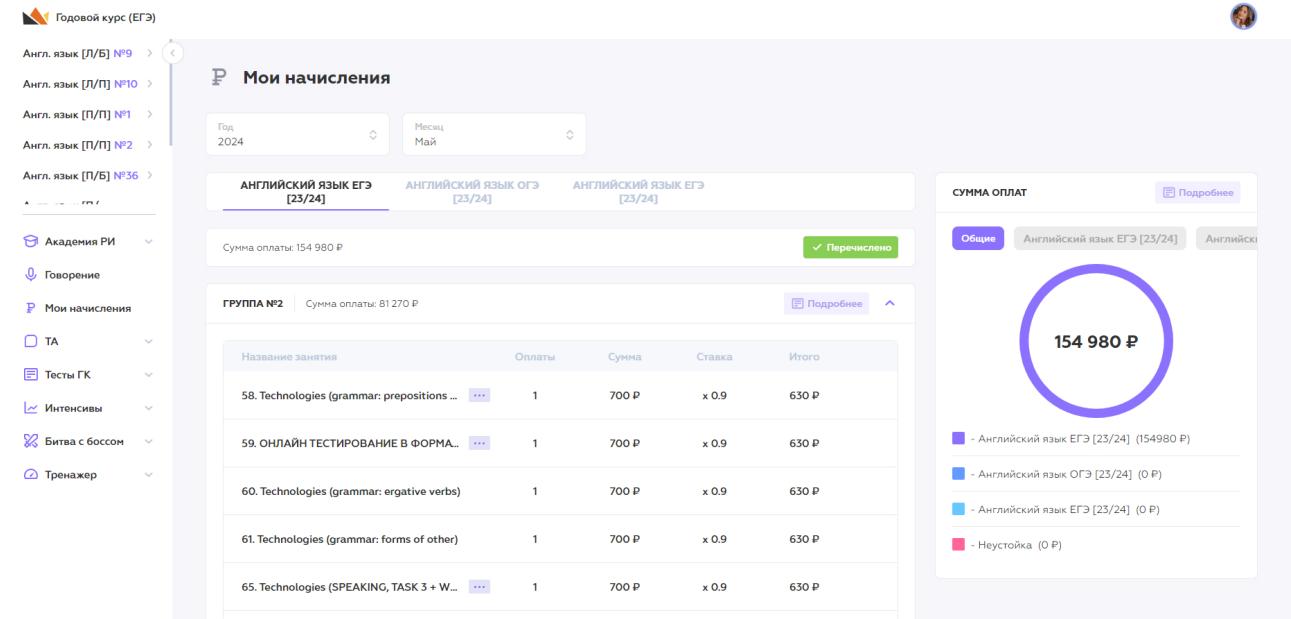


Рисунок Г.7 – Просмотр заработной платы

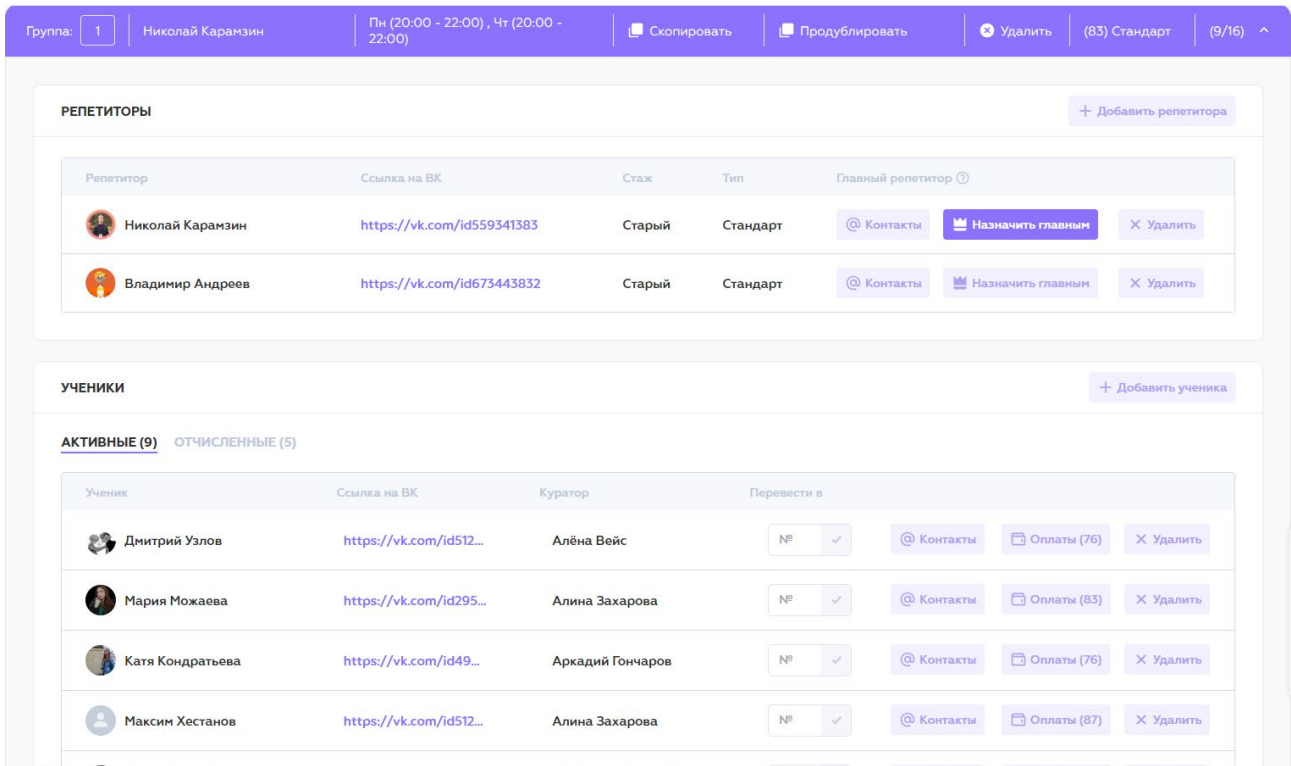


Рисунок Г.8 – Создана новая учебная группа

Никита Аникин

Репетитор

<https://vk.com/id210720286>

^

Контроль качества

[ГК](#) [ОИ](#) [ПГК](#)

Маркет

[Управление товарами](#)

Менеджер

[Марафон](#) [Мега-забег](#)

Огненное итоговое

[Репетитор](#) [Редактор тестов \(22/23\)](#) [Редактор тестов \(23/24\)](#)

Просмотр

[Отчётов всего центра](#) [Прочтения рассылок в беседах](#)

Редактор интенсивов ГК

[История](#) [Обществознание](#) [Литература](#) [Русский язык](#) [Английский язык](#) [Математика](#)

[Информатика](#) [Химия](#) [Физика](#) [Полигон](#) [Биология](#)

Рисунок Г.9 – Права изменены