|  |  |  |
| --- | --- | --- |
| Министерство образования и науки Российской Федерации | | |
| федеральное государственное бюджетное образовательное учреждение высшего образования | | |
| «Иркутский государственный университет» | | |
| (ФГБОУ ВО «ИГУ») | | |
| Факультет бизнес-коммуникаций и информатики | | |
|  | | |
|  | |
| **ОТЧЕТ ПО ПРОИЗВОДСТВЕННОЙ ПРАКТИКЕ**  **Б2.В.04(П) Технологическая (проектно-технологическая) практика**  **по направлению 09.03.03 «Прикладная информатика»**  **(прикладной бакалавриат)**  **профиль «Прикладная информатика в дизайне»** | |
|  | |
| Разработка сервиса для подготовки к ЕГЭ | |
|  |  |
|  | Студент 4 курса очной формы обучения, группа 14422-ДБ |
|  | \_\_\_\_\_\_\_\_\_\_ В.Э. Никитин |
|  |  |
|  | Руководитель: к.ф.-м.н., доцент  \_\_\_\_\_\_\_\_\_\_ А.Г. Балахчи |
|  | |
|  | Работа защищена: |
|  | «\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2025 г. |
|  | С оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  |  |

|  |
| --- |
|  |

# Оглавление

[Оглавление 2](#_Toc193630812)

[ВВЕДЕНИЕ 3](#_Toc193630813)

[1 ОБЗОР ПЛАТФОРМ ДЛЯ ОНЛАЙН-ОБУЧЕНИЯ И ПОДГОТОВКИ К ЕГЭ, ВЫЯВЛЕНИЕ ТРЕБОВАНИЙ К ФУНКЦИОНАЛУ И ВЫБОР СРЕДСТВ РАЗРАБОТКИ ПРИЛОЖЕНИЯ 5](#_Toc193630814)

[1.1 Обзор существующих решений на рынке онлайн-обучения 5](#_Toc193630815)

[1.2 Минимальные требования к функционалу платформы 9](#_Toc193630816)

[1.3 Инструменты разработки 9](#_Toc193630817)

[Выводы по главе 13](#_Toc193630818)

[2 РАЗРАБОТКА МАКЕТА ДИЗАЙНА ПЛАТФОРМЫ, РАЗВЕРТЫВАНИЕ ЛОКАЛЬНОГО ВЕБ-СЕРВЕРА, НАЧАЛО РАЗРАБОТКИ ПРОТОТИПА САЙТА 15](#_Toc193630819)

[2.1 Разработка макета дизайна платформы 15](#_Toc193630820)

[2.2 Развертывание локального веб-сервера 22](#_Toc193630821)

[2.3 Разработка прототипа приложения 25](#_Toc193630822)

[Выводы по главе 43](#_Toc193630823)

[ЗАКЛЮЧЕНИЕ 45](#_Toc193630824)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 46](#_Toc193630825)

[ПРИЛОЖЕНИЯ 47](#_Toc193630826)

[Приложение А 47](#_Toc193630827)

[Ежедневные записи студента по практике 47](#_Toc193630828)

[Приложение Б 48](#_Toc193630829)

[Ссылки на разработанные материалы 48](#_Toc193630830)

# ВВЕДЕНИЕ

В условиях активной цифровизации образовательного процесса и возрастающей конкуренции в сфере дополнительного образования, особенно в подготовке к ЕГЭ, важным становится применение современных информационных технологий. Современные образовательные платформы, являющиеся системами управления обучением (LMS), позволяют значительно повысить качество и эффективность обучения, автоматизируя рутинные процессы, такие как проверка домашних заданий и проведение тестирований. Разработка специализированного веб-приложения для подготовки школьников к ЕГЭ, объединяющего в себе функционал планирования занятий, загрузки домашних заданий, проведения тестов и онлайн-вебинаров, является перспективным направлением, способствующим оптимизации образовательного процесса и повышению успеваемости учащихся.

**Целью** производственной практики является освоение современных инструментов разработки веб-приложений и создание базового прототипа онлайн-платформы для подготовки школьников ИНК Академии к ЕГЭ по информатике. Практика направлена на изучение технологий Python FastAPI, Vue.js и MySQL, разработку макета дизайна для пользователей, развертывание локального веб-сервера и реализацию ключевых функциональных модулей системы.

**Задачи работы:**

1. проанализировать существующие платформы для онлайн-обучения;
2. выявить минимальные требования к реализуемому функционалу;
3. ознакомиться с инструментами разработки;
4. разработать макет дизайна платформы;
5. развернуть локальный веб-сервер;
6. приступить к разработке прототипа, заложив основу для дальнейшего совершенствования функционала.

**Объектом исследования** являются информационные системы и веб-приложения, предназначенные для подготовки школьников к ЕГЭ.

**Предметом исследования** выступают методы и технологии разработки онлайн-платформ, а также принципы проектирования и развертывания веб-сервисов с базой данных.

**Практическая значимость** данной работы заключается в приобретении навыков работы с современными инструментами веб-разработки, что позволяет создать функциональный прототип образовательной платформы. Полученные результаты способствуют оптимизации процесса подготовки школьников к ЕГЭ и могут быть использованы для дальнейшего совершенствования платформы, что сделает её полноценным решением в рамках LMS.

# 1 ОБЗОР ПЛАТФОРМ ДЛЯ ОНЛАЙН-ОБУЧЕНИЯ И ПОДГОТОВКИ К ЕГЭ, ВЫЯВЛЕНИЕ ТРЕБОВАНИЙ К ФУНКЦИОНАЛУ И ВЫБОР СРЕДСТВ РАЗРАБОТКИ ПРИЛОЖЕНИЯ

## Обзор существующих решений на рынке онлайн-обучения

В условиях активного развития цифровых технологий в образовании системы управления обучением (Learning Management Systems, LMS) играют важную роль в организации учебного процесса. LMS представляют собой интегрированные программные решения, предназначенные для автоматизации и оптимизации образовательного процесса. Такие системы обеспечивают возможность дистанционного обучения, проведения тестирований, планирования и управления учебными мероприятиями, а также контроля за успеваемостью учащихся. Использование LMS позволяет образовательным учреждениям адаптироваться к современным требованиям, повышая эффективность обучения и расширяя доступ к образовательным ресурсам.

На рынке онлайн-образования представлено множество решений, каждое из которых обладает своими преимуществами и недостатками.

Одним из популярных сервисов, ориентированных на подготовку школьников к ЕГЭ, является **«Умскул»**. Платформа предлагает продуманную структуру для организации занятий, к которым прикрепляются видеолекции и рабочие тетради с полезной информацией для подготовки к уроку, систему поощрения школьников для их лучшей успеваемости и яркий современный дизайн с интуитивно понятной навигацией. Среди полезных функций —тестирование, где к каждому вопросу из текста есть возможность прикрепить подсказку и теорию (рис 1.). Из минусов можно выделить отсутствие аналитики своей успеваемости (только через систему достижений) и то, что платформа полностью платная, бесплатных материалов для ознакомления с платформой не найти, только попросив доступ у поддержки сайта.

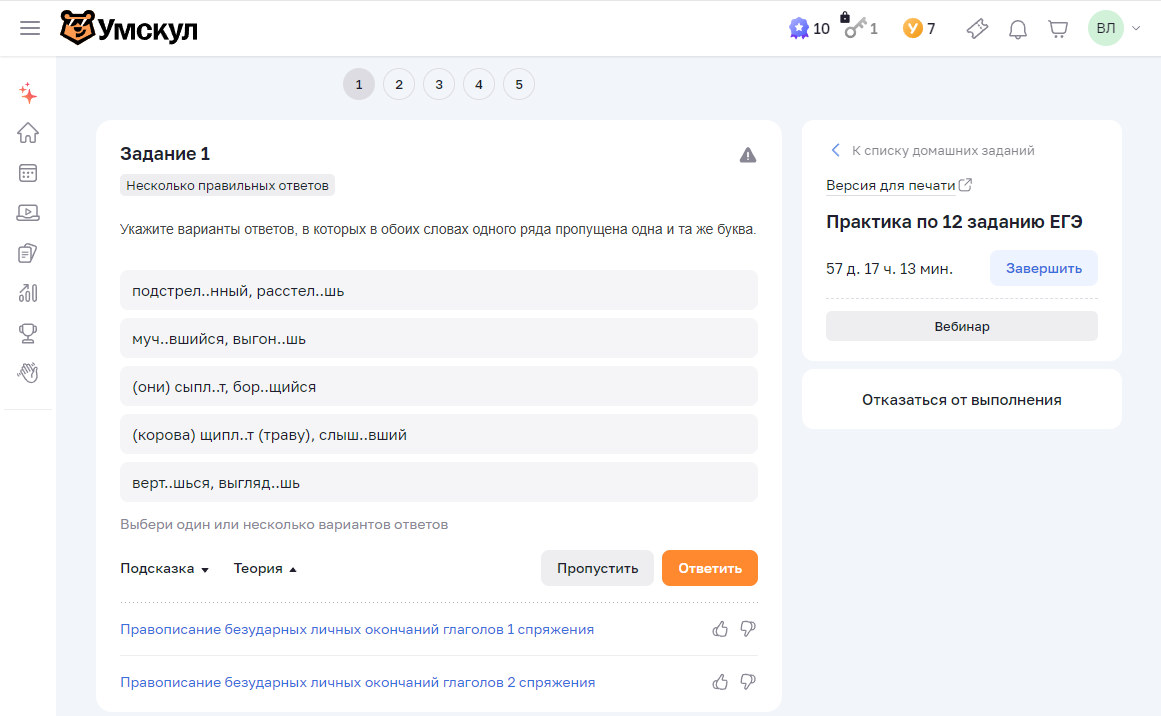


Рисунок 1 – Интерфейс тестирования в «Умскул»

Другой популярной платформой является «Фоксфорд», представляющий собой современную онлайн-школу, которая предлагает широкий спектр курсов для учеников различных возрастных категорий. Среди её преимуществ выделяются возможность тренировки отдельных заданий КИМа ЕГЭ с большим количеством сопутствующих материалов для подготовки, возможность опробовать функционал платформы и получить доступ к пробным урокам курсов зарегистрировавшись на ней. Кроме того, система позволяет учащимся отслеживать свою успеваемость и предлагает удобный формат интерактивных вебинаров, где учебные материалы и задания интегрированы в интерфейс занятия. Это обеспечивает комфортное взаимодействие с преподавателем без необходимости переключаться между вкладками браузера (рис2).

Из минусов “Фоксфорда” можно отметить перегруженный и местами неинтуитивный интерфейс, что затрудняет навигацию для новых пользователей.

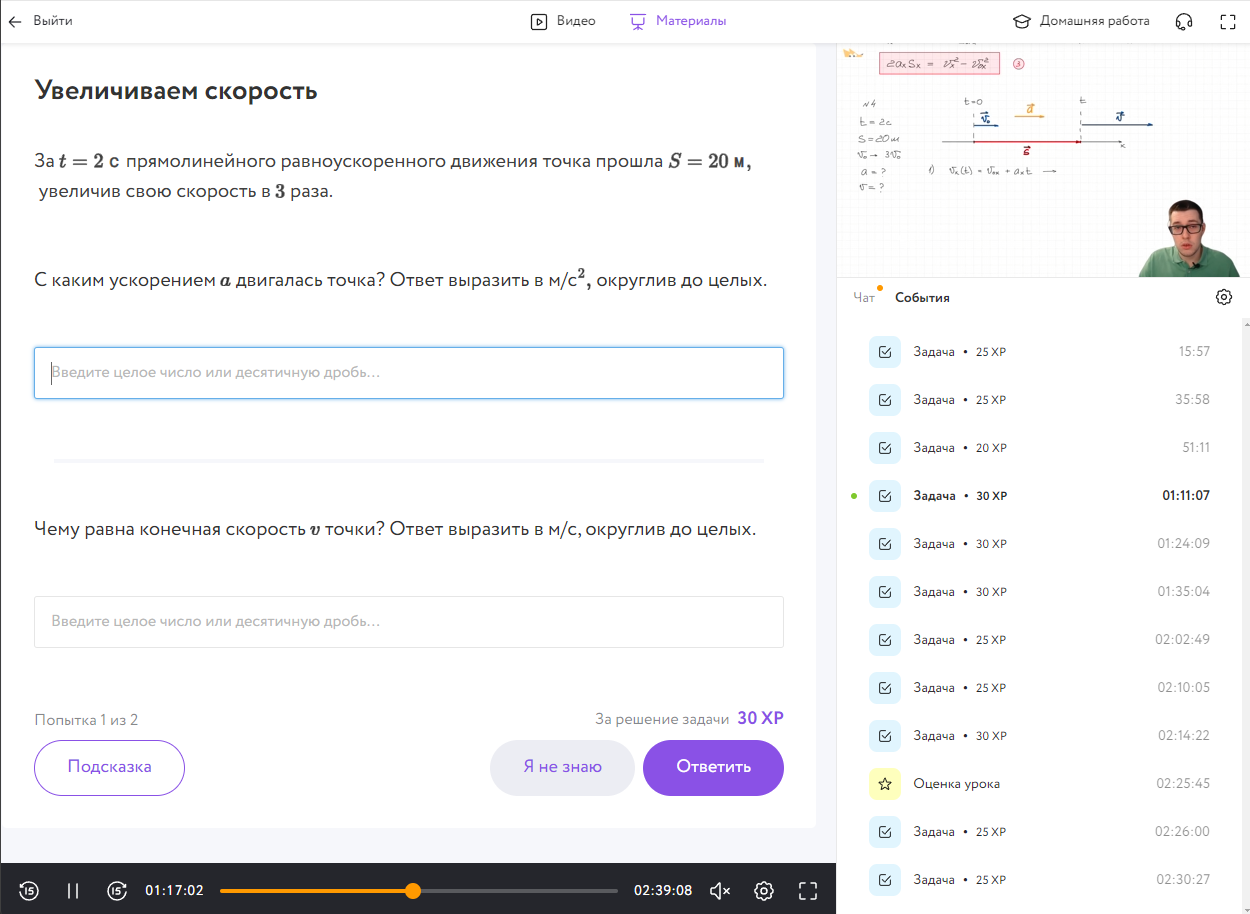


Рисунок 2 – Интерфейс вебинара на «Фоксфорд»

Платформа Moodle представляет собой одну из наиболее распространенных систем управления обучением (Learning Management System, LMS) с открытым исходным кодом. Данная платформа широко используется образовательными учреждениями по всему миру благодаря своей гибкости и возможности адаптации под различные учебные процессы. Moodle поддерживает создание онлайн-курсов, управление учебными материалами, проведение тестирования, отслеживание успеваемости учащихся и взаимодействие между преподавателями и студентами через встроенные форумы и чаты. Несмотря на свои преимущества, Moodle имеет серьезный недостаток в виде его устаревшего интерфейса, который осложняет его использование в контексте подготовки школьников к ЕГЭ и может затруднить восприятие учащимися. Интерфейс платформы (рис 4), хотя и функциональный, но требует доработки для повышения удобства взаимодействия.

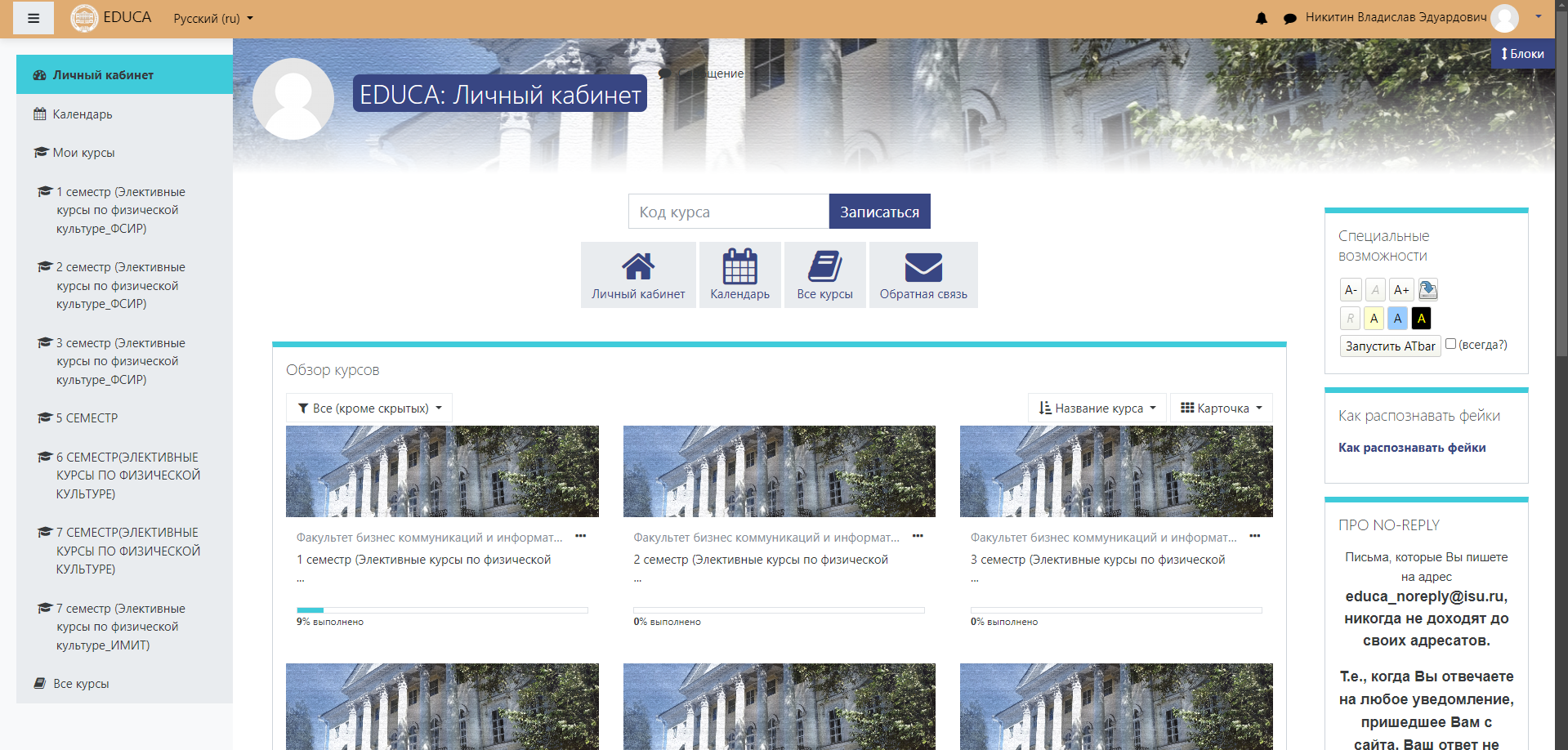


Рисунок 4 – Интерфейс EDUCA, основанный на платформе Moodle

​Платформа «Решу ЕГЭ» представляет собой бесплатный онлайн-ресурс, предназначенный для подготовки школьников к Единому государственному экзамену (рис. 3).

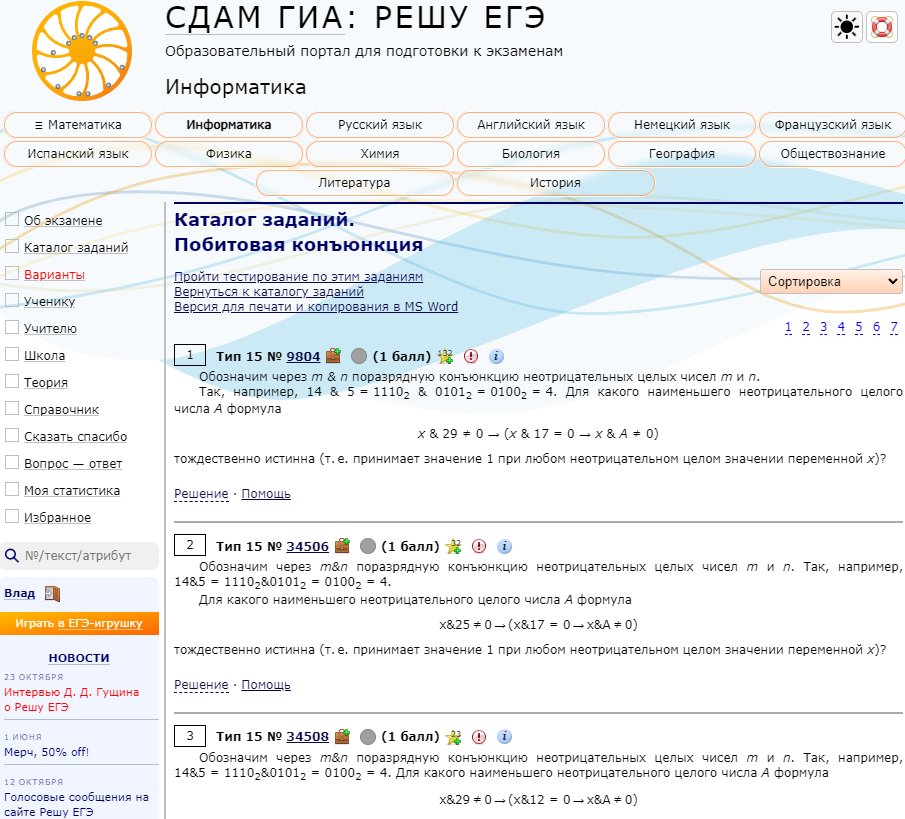


Рисунок 3 – Интерфейс сайта «Решу ЕГЭ»

Его главное преимущество — простая и легковесная структура, обеспечивающая стабильную работу даже при слабом интернет-соединении. Интерфейс платформы в разделе заданий интуитивно понятен, что позволяет пользователям легко ориентироваться. Однако в целом интерфейс остального сайта выглядит устаревшим. Кроме того, ограниченная интерактивность и отсутствие мультимедийных элементов может снижать вовлеченность учащихся.

## Минимальные требования к функционалу платформы

На основе анализа существующих решений на рынке онлайн-образования, для создания эффективной системы подготовки школьников к ЕГЭ по информатике необходимо определить минимальный набор функциональных требований, обеспечивающих автоматизацию и оптимизацию образовательного процесса в рамках LMS. Прежде всего, платформа должна поддерживать две ключевые роли пользователей: преподавателя и школьника. Преподаватели должны иметь возможность планировать занятия, загружать учебные материалы, домашние задания и тесты, а также проводить вебинары. Школьники, в свою очередь, должны получать доступ к просмотру занятий и материалов к ним, сдаче домашних заданий, прохождению тестов и просмотру вебинаров.

Для проведения вебинаров планируется интеграция с сервисом VK Видео. Система тестирования должна предусматривать автоматическую проверку результатов.

Не менее значимым является наличие личного кабинета для пользователей, где будет отображаться успеваемость, статус выполнения домашних заданий и тестов, а также предоставляться аналитическая информация об образовательном процессе. Дизайн платформы должен быть адаптирован под аудиторию школьников и соответствовать корпоративному стилю ИНК Академии.

## Инструменты разработки

Проект построен по принципам REST архитектуры [1], что подразумевает четкое разделение на клиентскую и серверную части, взаимодействующие посредством стандартизированных API. Фронтенд и бэкенд реализованы как самостоятельные, но тесно взаимосвязанные компоненты. Взаимодействие между ними осуществляется через REST API, что обеспечивает модульность, масштабируемость и независимую разработку каждой части системы. Фронтенд представляет собой динамичный, одностраничный адаптивный интерфейс, разработанный для максимального удобства пользователей, а бэкенд отвечает за обработку логики платформы, выполнение вычислительных операций и интеграцию с внешними сервисами. Для хранения данных используется масштабируемая база данных MySQL, способная эффективно обрабатывать большие объемы информации и поддерживать необходимую производительность при росте нагрузки. Такая архитектурная модель позволяет легко интегрировать внешние сервисы, например, для проведения вебинаров, что значительно расширяет функциональность системы.

Про выбор фреймворка для реализации клиентской части проекта было проведено сравнение Vue.js, React.js и Angular.

Vue.js [2] отличается своей интуитивностью и легкостью освоения, что позволяет быстро приступить к разработке. Его лаконичный синтаксис и естественная поддержка компонентной архитектуры делают создание адаптивных и динамичных одностраничных приложений простым и эффективным. При этом Vue.js демонстрирует высокую производительность, что особенно важно для приложений с интенсивным взаимодействием с пользователем. В сравнении с ним React.js [3], хотя и обладает широкой поддержкой сообщества и богатой экосистемой сторонних библиотек, требует более детальной настройки: разработчикам приходится самостоятельно подбирать инструменты для маршрутизации и управления состоянием, что может привести к увеличению времени разработки и усложнению архитектуры проекта.

Angular [3], являясь полнофункциональным фреймворком, включает в себя множество встроенных решений и строгую структуру, однако его сложность и высокий порог входа зачастую оказываются избыточными для проектов небольшого и среднего масштаба. Таким образом, с учетом требований к быстроте разработки, гибкости и производительности, выбор Vue.js обоснован как оптимальное решение для данного проекта.

FastAPI [4] получил предпочтение благодаря своей асинхронной природе и высокой производительности, что критично для обработки множества одновременных запросов в режиме реального времени. Используя современный синтаксис Python с аннотациями типов, FastAPI позволяет разработчикам не только создавать быстрые и масштабируемые API, но и автоматически генерировать документацию (Swagger UI и ReDoc) без дополнительных усилий. Это существенно ускоряет процесс разработки, снижает вероятность ошибок при интеграции и упрощает поддержку кода. Кроме того, встроенный механизм dependency injection помогает организовать код в модульном виде, что делает проект легко расширяемым.

Django [4], хотя и обладает широким набором встроенных инструментов, таких как система аутентификации, административная панель и ORM, традиционно ориентирован на монолитные решения. Это может ограничивать гибкость при создании специализированных API, поскольку Django по умолчанию работает в синхронном режиме. Хотя существуют возможности для реализации асинхронных обработчиков в последних версиях, они не так хорошо интегрированы в общий рабочий процесс фреймворка, как это реализовано во FastAPI.

Flask [4] представляет собой легковесное и минималистичное решение, которое даёт разработчику большую свободу выбора архитектурных подходов и дополнительных библиотек. Однако базовая версия Flask не поддерживает асинхронность, что требует дополнительной настройки или использования расширений, например, таких как Quart для асинхронного исполнения. Это добавляет время на конфигурацию и может увеличить время разработки, особенно если требуется масштабируемость и высокая производительность в режиме реального времени.

Для сравнения СУБД для разрабатываемой платформы, были выбраны MySQL и PostgreSQL [5]. Веб-приложение не предполагает реализации сложной бизнес-логики или обработки тяжёлых запросов, поэтому MySQL оказывается более предпочтительным выбором. MySQL широко известна своей простотой настройки, высокой скоростью выполнения стандартных операций создания, чтения, обновления и удаления (CRUD), а также низкими накладными расходами при типичных сценариях веб-разработки. Она идеально подходит для проектов, где объем данных и сложность запросов не требуют использования расширенных возможностей, предоставляемых PostgreSQL.

PostgreSQL, напротив, предлагает богатый функционал, включая поддержку сложных запросов, расширенную работу с транзакциями, полнотекстовый поиск и строгую приверженность стандартам SQL. Эти возможности делают PostgreSQL привлекательной для крупных и сложных систем, где критически важна гибкость и точность обработки данных. Однако для небольшого или среднего веб-приложения, которое не будет нагружено сложными вычислениями или высокими объемами данных, дополнительные возможности PostgreSQL могут оказаться избыточными, а их настройка — сложнее и требует большего времени на администрирование.

Учитывая, что веб-приложение не является чрезмерно сложным, выбор MySQL обоснован её простотой, производительностью и удобством управления, что позволяет сосредоточиться на разработке функционала, не тратя лишние ресурсы на конфигурацию и поддержку СУБД. Это делает MySQL оптимальным решением для проекта, где важны быстрота и надежность стандартных операций без необходимости использования сложных функций, предоставляемых PostgreSQL.

При выборе ORM-инструмента проводилось сравнение SQLAlchemy, Django ORM и Peewee [6]. SQLAlchemy обеспечивает высокую производительность и гибкость, позволяя строить сложные запросы и эффективно масштабировать решение. Его возможности по тонкой настройке и расширенной поддержке сложной логики взаимодействия с базой данных делают его предпочтительным выбором для проектов, требующих высокой масштабируемости. Django ORM, хотя и удобен благодаря своей интеграции в экосистему Django, имеет ограничения при построении сложных запросов и зачастую не справляется с высокими нагрузками, характерными для масштабируемых решений. Peewee, несмотря на свою простоту и низкий порог входа, оказывается недостаточно производительным для реализации сложных и ресурсоемких операций с данными. С учетом того, что проект использует MySQL, выбор SQLAlchemy обоснован необходимостью обеспечить достаточную гибкость и производительность при работе с базой данных.

## Выводы по главе

По итогам главы было выполнено 3 из 6 поставленных задач.

Первая задача заключалась в анализе существующих платформ для онлайн-обучения, особенно ориентированных на подготовку школьников к ЕГЭ. В ходе исследования были рассмотрены различные системы управления обучением (LMS) и специализированные образовательные решения. Это позволило выявить ключевые требования к функционалу разрабатываемой платформы, такие как удобство использования, наличие системы тестирования, организация вебинаров и возможность мониторинга успеваемости учащихся.

Вторая задача заключалась в определении минимального набора функциональных требований и выделении необходимых возможностей, чтобы платформа соответствовала ожиданиям пользователей. Были сформулированы требования к ролям пользователей, возможностям планирования занятий, загрузке учебных материалов, проведению вебинаров и тестированию, а также к системе аналитики успеваемости. Это позволило создать чёткое представление о том, какой функционал необходимо реализовать для оптимизации процесса подготовки школьников к ЕГЭ.

Третья задача состояла в выборе инструментов разработки и обосновании архитектурного решения проекта. Для реализации клиентской части проведён сравнительный анализ Vue.js, React.js и Angular, в результате которого Vue.js был признан оптимальным благодаря своей гибкости, простоте освоения и высокой производительности, что особенно важно для создания динамичного одностраничного интерфейса. Аналогичным образом, анализ серверной части показал, что FastAPI превосходит Django и Flask по скорости, асинхронности и удобству автоматической генерации документации, что критично для обработки множества запросов в режиме реального времени. Кроме того, проведено сравнение систем управления базами данных, в частности MySQL и PostgreSQL. Для данного веб-приложения, не предполагающего реализации сложной бизнес-логики и тяжёлой обработки запросов, MySQL оказалась более предпочтительным выбором благодаря простоте настройки, высокой скорости выполнения стандартных операций и низким накладным расходам. Выбор ORM-инструмента пал на SQLAlchemy, который обеспечивает необходимую гибкость, высокую производительность и возможность построения сложных запросов, что в совокупности с MySQL позволяет оптимально реализовать работу с базой данных.

Таким образом, выполненные задачи позволили определить ключевые требования к функционалу образовательной платформы, а также обосновать выбор современных инструментов разработки. Полученные результаты создают прочную основу для дальнейшего развития системы как полноценного решения в рамках LMS, обеспечивая баланс между производительностью, гибкостью и удобством эксплуатации, что особенно важно для эффективной подготовки школьников к ЕГЭ.

# 2 РАЗРАБОТКА МАКЕТА ДИЗАЙНА ПЛАТФОРМЫ, РАЗВЕРТЫВАНИЕ ЛОКАЛЬНОГО ВЕБ-СЕРВЕРА, НАЧАЛО РАЗРАБОТКИ ПРОТОТИПА САЙТА

* 1. Разработка макета дизайна платформы

Разработка макета дизайна платформы стала ключевым этапом в формировании современного и удобного веб-приложения для подготовки школьников к ЕГЭ, которое соответствует корпоративному стилю ИНК Академии. Использованы фирменные цвета, шрифты и логотип [7]. На данном этапе были проработаны макеты всех основных страниц, обеспечивающих интуитивно понятное и эффективное взаимодействие как для учащихся, так и для преподавателей. В основу визуальной концепции легли принципы минимализма, адаптивного дизайна и единообразия, что позволило создать решение, удобное для работы на различных устройствах и отвечающее современным требованиям к пользовательскому опыту.

Стартовой страницей является страница «План на день» (рис. 5), где отображаются запланированные занятия, домашние задания и ближайшие уроки.

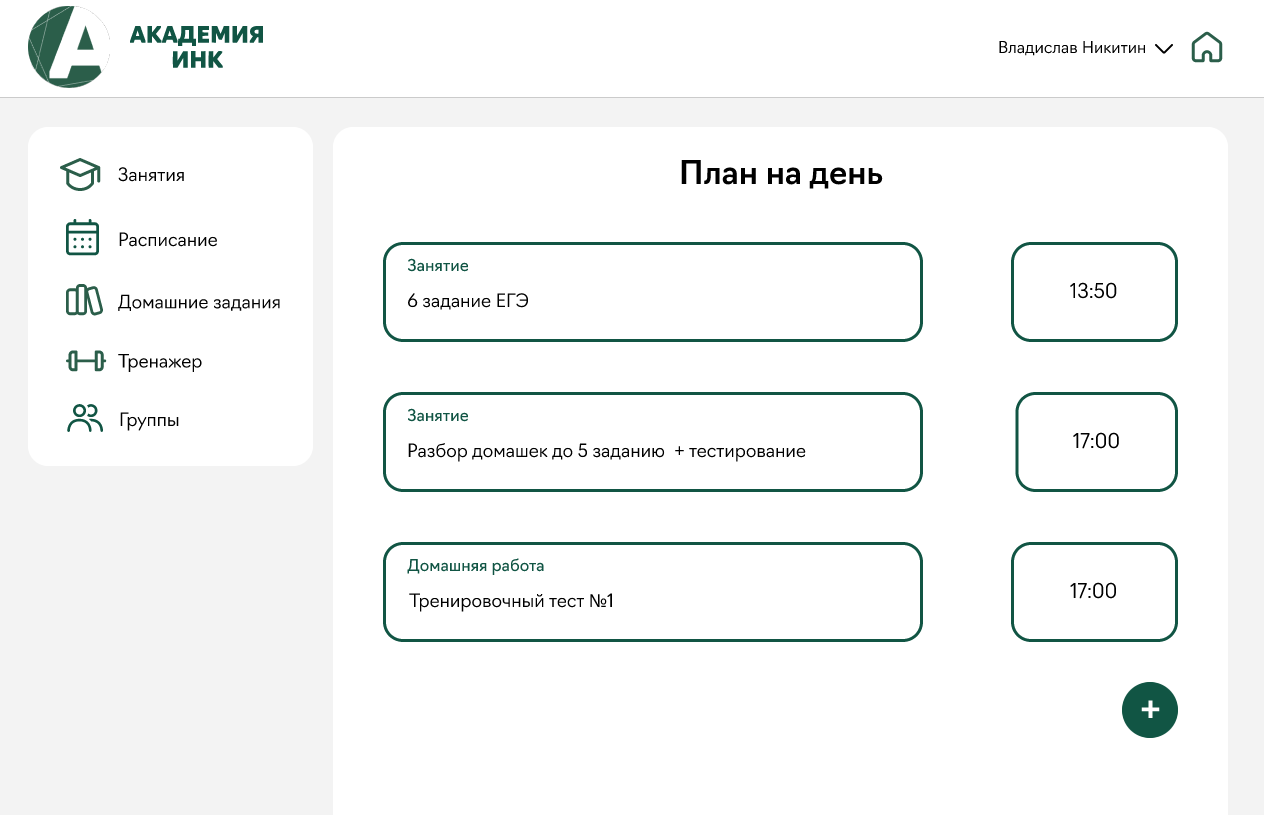


Рисунок 5 – Макет дизайна страницы «План на день»

Данная страница выполнена в лаконичном стиле, с акцентом на четкое представление информации: блоки с занятиями и домашними заданиями структурированы по временной шкале, что позволяет быстро соориентироваться с планами на сегодня.

Боковое меню, закрепленное в левой части сайта, является важной навигационной панелью, доступной на всех страницах платформы. Оно содержит пункты, такие как «Занятия», «Расписание», «Домашние задания», «Тренажер», а также раздел для преподавателя – «Группы». Такое меню обеспечивает быстрый доступ к основным функциям платформы и помогает пользователям легко перемещаться между различными разделами.

Страница с занятиями разработана так, что на ней отображаются все запланированные, прошедшие и просмотренные уроки. Каждое занятие сопровождается информацией о времени и дате, а при нажатии на конкретное занятие пользователь попадает на детальную страницу, где расположен блок с видео или вебинаром, описание урока и дата проведения (рис.6). На этой странице также предусмотрены кнопки для перехода к материалам занятия и выполнения домашнего задания.

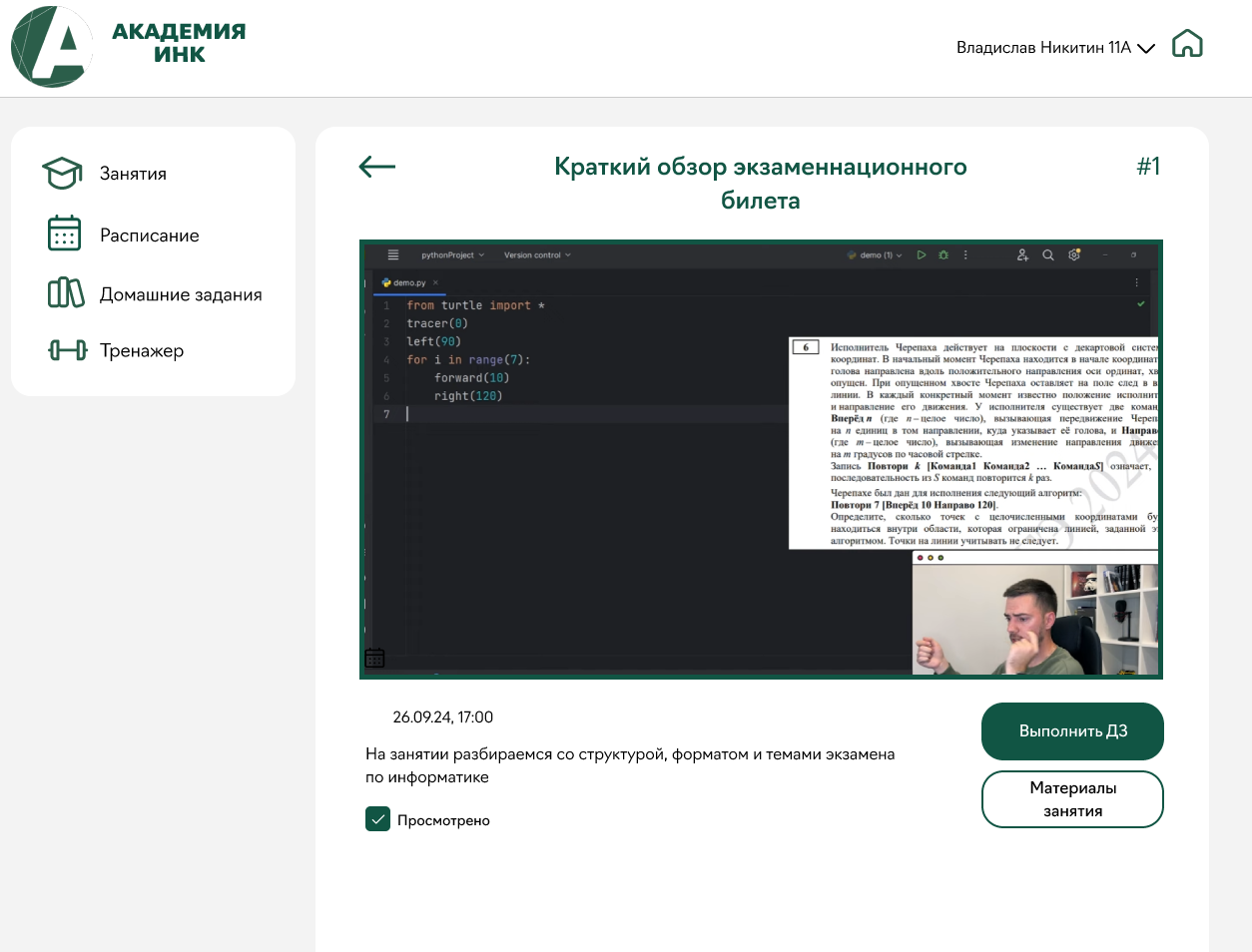


Рисунок 6 – Макет страницы занятия

При нажатии на кнопку «Материалы занятия» пользователь перенаправляется на страницу с прикрепленными учебными материалами, предоставленными преподавателем, а кнопка «Выполнить ДЗ» открывает отдельную страницу с подробным описанием домашнего задания, где школьник может прикрепить файлы и оставить текстовый комментарий к отклику (рис.7). Функционал позволяет как отправить ответ, так и отредактировать его впоследствии, а также отображать отклик преподавателя с оценкой.

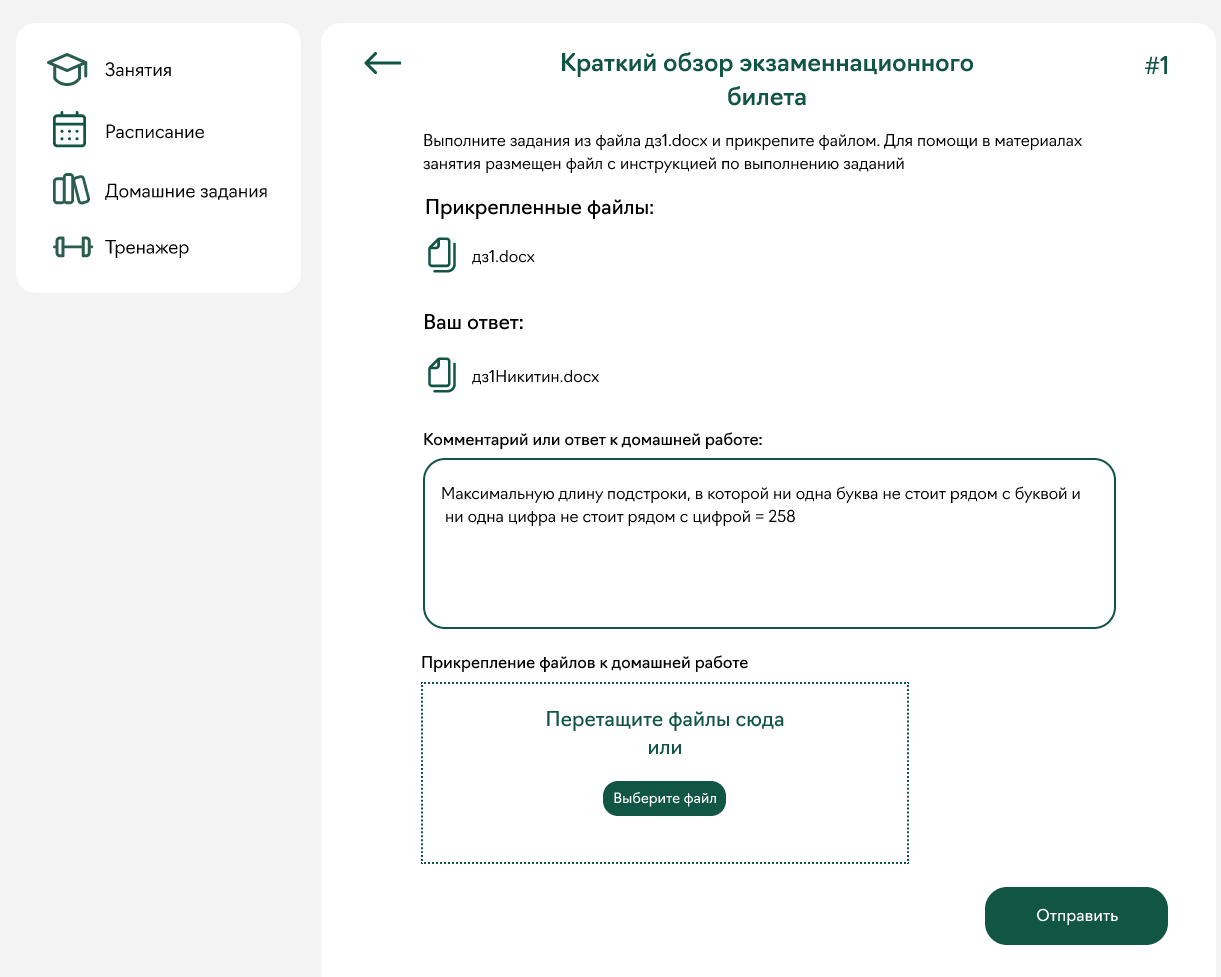


Рисунок 7 – Макеты страницы домашнего задания с отправкой отклика

Страница с домашними заданиями включает в себя список всех домашних заданий с указанием статуса (оценено, ожидает оценки, текущее, просрочено) и дат дедлайна (рис. 8). При выборе конкретного домашнего задания пользователь переходит на страницу с подробной информацией о задании, где отображаются текст задания, прикрепленные изображения и файлы, а также форма для отправки ответа.

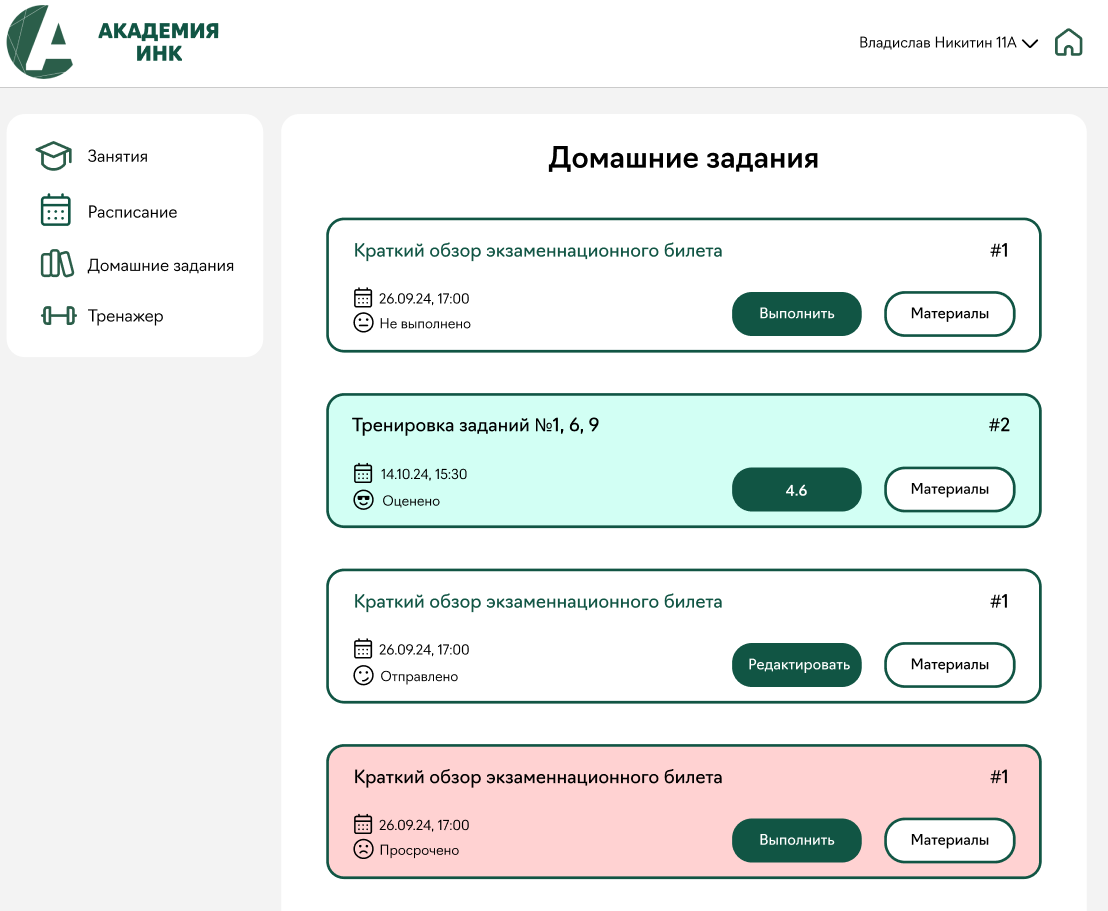


Рисунок 8 – Макет страницы списка домашних заданий

Отдельное внимание уделено странице с расписанием (рис. 9), где интегрирован удобный календарь. Пользователь может листать календарь и быстро находить нужную дату, отмеченную точкой, если на ней запланировано занятие или домашнее задание. При наведении курсора на конкретную дату отображается краткая информация о запланированном событии, а по клику осуществляется переход на страницу с детальной информацией о занятии или домашнем задании. Также есть возможность фильтровать, что пользователь хочет видеть в календаре – запланированные занятия или домашние задания. Календарь позволяет легко ориентироваться в расписании и планировать учебный процесс.

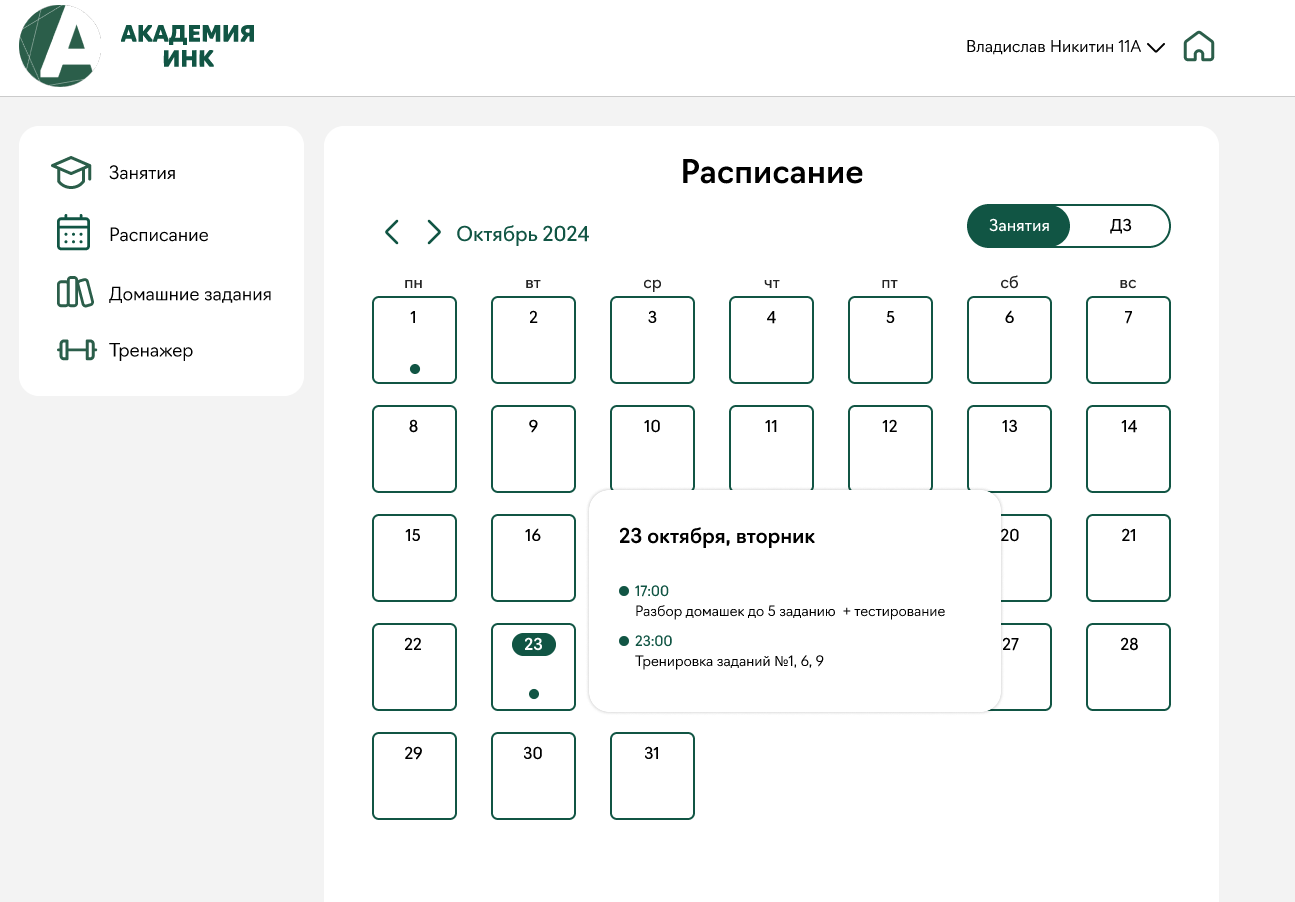


Рисунок 9 – Макет страницы расписания

Профиль пользователя разработан так, чтобы отображать все данные о студенте, включая общий балл и информацию о группе, к которой он принадлежит. В верхней части сайта, в шапке, размещены логотип ИНК Академии и имя пользователя с указанием группы, а также выпадающее меню с пунктами «Профиль» и «Контакты». Страница «Контакты» содержит информацию о преподавателях, технической поддержке и другие служебные данные, что позволяет пользователям быстро найти необходимую помощь.

Особый раздел платформы отведен под тренажер (рис. 10), предназначенный для подготовки школьников к ЕГЭ по информатике. На странице тренажера представлен список всех заданий ЕГЭ из КИМа, где пользователь может выбрать нужное задание и запустить тест. Каждое задание будет сопровождаться подробным описанием и указанием максимального количества баллов за его выполнение.

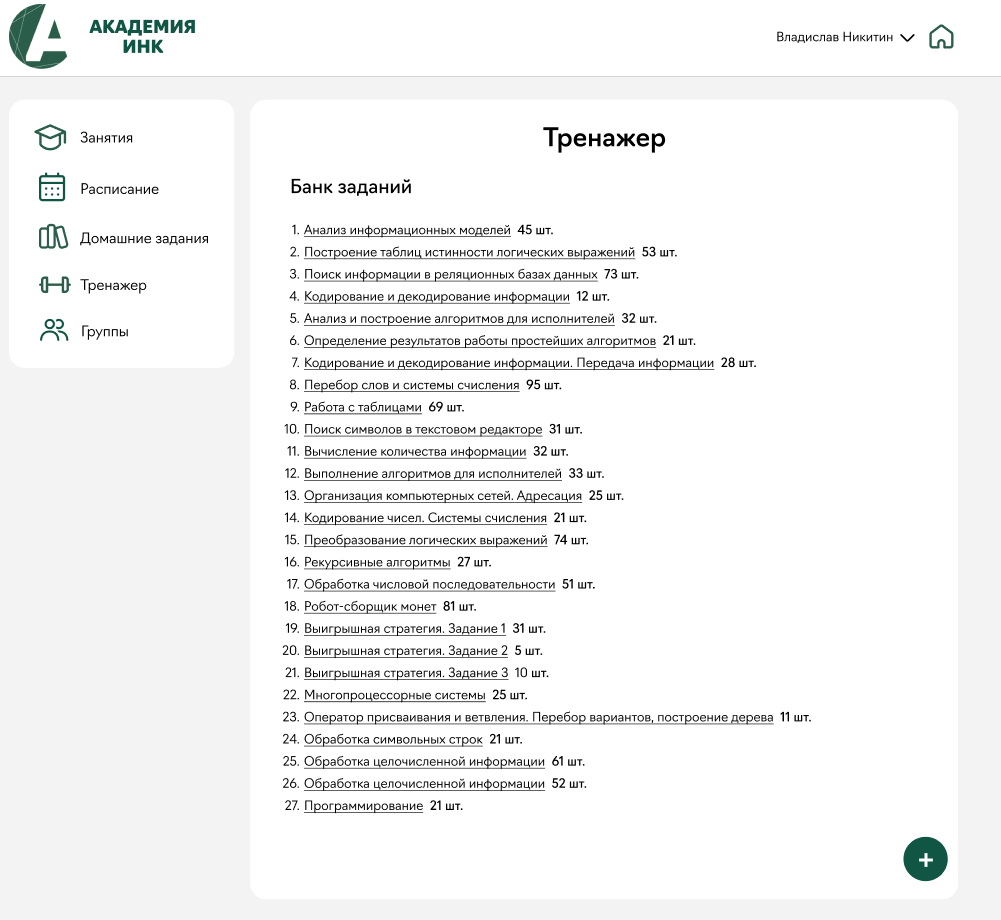


Рисунок 10 – Макет страницы тренажера

При запуске теста боковое меню трансформируется в специальное меню теста, которое отображает оставшееся время, количество заданий и содержит кнопки для перехода между вопросами (рис. 11). Структура тестовых заданий адаптируется под тип задания – от ввода одиночной цифры до заполнения таблицы. После прохождения теста пользователь видит свою оценку, количество правильно решенных заданий и имеет возможность выйти из режима тестирования. В будущем можно сохранять результаты тестов в профиле пользователя для дальнейшего их анализа.

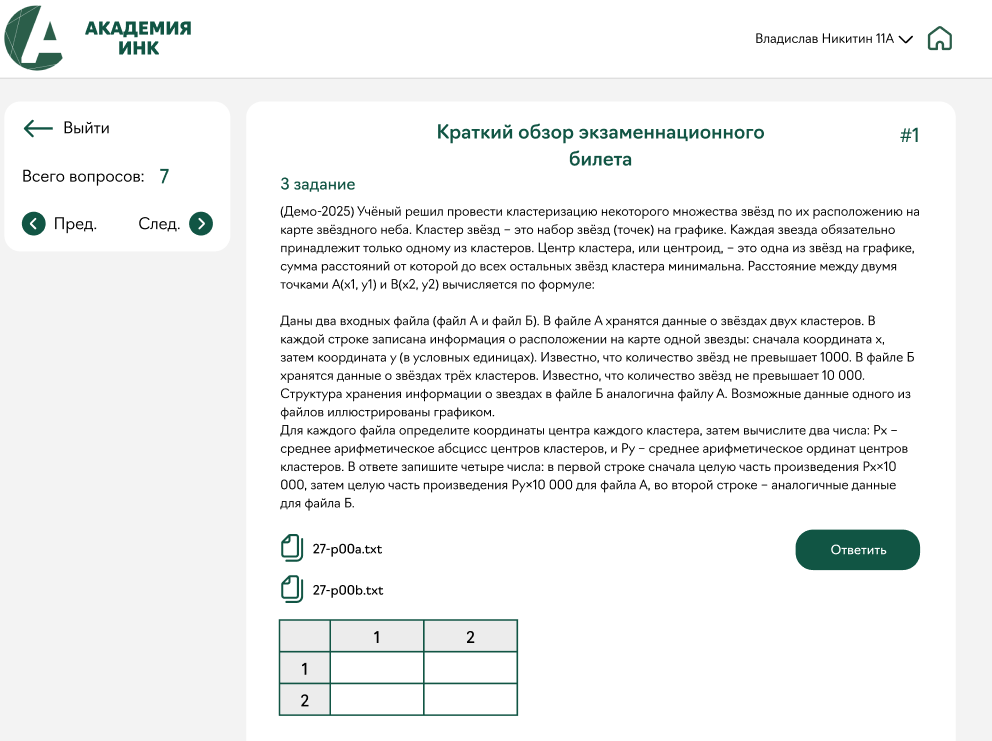


Рисунок 11 – Макет страницы прохождения тестов

Дизайн для преподавателя включает те же страницы, что и для учащихся, но с добавлением служебных функций для управления контентом. Преподавателю доступны страницы для добавления и редактирования домашних заданий, планирования уроков, а также для создания тестов к ДЗ. В боковом меню появляется новый пункт «Группы», позволяющий преподавателю создавать группы, генерировать коды для вступления и просматривать список участников с данными о школьниках и их успеваемости. Кроме того, в разделе тренажера преподаватель может вручную добавлять тестовые задания или собирать их из уже загруженных вариантов, что позволяет генерировать случайные варианты КИМа для тренировок.

Разработка макета дизайна проводилась с учетом анализа решений на рынке образовательных платформ. Все исходные файлы макетов доступны для ознакомления в GitHub репозитории, ссылка на который приведена в приложении Б.

* 1. Развертывание локального веб-сервера

В данном проекте структура репозитория организована таким образом, что в его корневой директории созданы два основных модуля, предназначенных для разработки отдельных компонентов платформы. Один модуль отведён для серверной части (backend), реализованной с использованием FastAPI, а другой – для клиентской части (frontend), разработанной с помощью Vue,js. В рамках разработки платформы был создан репозиторий на GitHub, ссылка на который приведена в приложении Б. Это позволило обеспечить централизованное управление исходным кодом и отслеживание внесенных изменений посредством интеграции с Visual Studio Code. Такой подход позволяет вести эффективный контроль версий, каждая новая реализация и изменение в коде фиксируются в репозитории.

В серверной части проекта реализована API-логика с использованием FastAPI [8], а сервер запускается посредством Uvicorn. Для этого предварительно было создано виртуальное окружение (venv), в котором установлены все необходимые библиотеки, что позволяет изолировать зависимости проекта от глобальной конфигурации Python. После настройки виртуального окружения и установки необходимых пакетов серверная часть запускается с помощью команды «uvicorn main:app –reload», обеспечивающей перезагрузку кода при изменениях, что значительно ускоряет процесс отладки и тестирования API. В папке backend (рис.12) находится файл main.py, в котором реализована основная логика эндпоинтов с использованием FastAPI. Эндпоинты отвечают за обработку входящих запросов по протоколу HTTP или HTTPS и возврат соответствующих ответов, каждый соответствует определённому URL-адресу и методу HTTP (GET, POST, PUT, DELETE и т. д.) Это позволяет централизованно управлять маршрутизацией и обработкой запросов. Кроме того, структура папки организована таким образом, что в каталоге app размещены файлы database.py и schemas.py, отвечающие за настройку подключения к базе данных и определение структур данных соответственно. Папка helpers содержит вспомогательные модули, предназначенные для реализации утилитарных функций и поддержки основной логики приложения, а каталог uploads предназначен для хранения файлов, загружаемых на сервер. В директории backend также находятся виртуальное окружение и файл requirements.txt, что позволяет поддерживать консистентность зависимостей и обеспечивает воспроизводимость сборки проекта.

В дополнение к указанным каталогам, в структуре проекта присутствует папка alembic, предназначенная для управления миграциями базы данных. Библиотека Alembic обеспечивает версионирование схемы базы данных и автоматизацию процессов обновления её структуры при изменениях в модели данных. Здесь хранятся файлы конфигурации и скрипты миграций, позволяющие применять последовательные изменения, гарантируя целостность и согласованность базы данных с текущей логикой приложения.

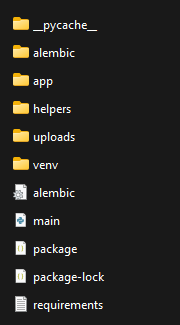


Рисунок 12 – Структура папки backend

В рамках организации клиентской части проекта была создана отдельная директория frontend, содержащая проект, инициированный с использованием Vue CLI [9]. Для установки Vue CLI на рабочей станции под управлением Windows была выполнена глобальная установка посредством команды «npm install -g @vue/cli», что обеспечило наличие инструментов для инициализации и последующей настройки проекта. Создание проекта происходило с применением команды «vue create my-project», после чего был предложен выбор предустановленных конфигураций, включающих поддержку маршрутизации через Vue Router, что позволило автоматически создать соответствующий каталог и файл конфигурации.

Структура проекта в директории my-project (рис.13) организована следующим образом. Каталог node\_modules содержит все установленные зависимости, необходимые для функционирования приложения, и формируется на основании файла package.json, в котором перечислены используемые библиотеки и версии пакетов. Файл package.json представляет собой основное описание проекта, содержащее метаданные, скрипты для сборки и тестирования, а также перечень зависимостей, что позволяет централизованно управлять конфигурацией и версиями используемых пакетов. Файл vue.config.js служит для настройки параметров сборки и поведения Vue CLI, позволяя модифицировать стандартные настройки проекта под специфические требования разработки, такие как проксирование запросов к серверной части. Файл babel.config.js обеспечивает трансформацию современного синтаксиса JavaScript в более совместимый вариант для исполнения в различных браузерах, а файл jsconfig.json упрощает работу с модульной структурой кода, предоставляя поддержку автодополнения и разрешения путей в редакторе кода.

Директория public содержит файл index.html, который является шаблоном для генерации итогового HTML-кода приложения. В каталоге src располагаются все исходные файлы, непосредственно определяющие логику и визуальное оформление приложения. В каталоге assets хранятся медиафайлы, необходимые для корректного отображения графических элементов, таких как изображения, шрифты и иконки. В свою очередь, каталог components структурирован по функциональному признаку: здесь находятся Vue-компоненты, разделённые на подкаталоги, предназначенные для различных ролей в приложении, таких как компоненты для работы со студентами, преподавателями, а также отдельные ui-компоненты, например, кнопки, и общие компоненты, используемые в различных частях проекта. Файл конфигурации маршрутов (обычно router.js) определяет маршруты для перехода между различными представлениями.

Связь между FastAPI и Vue CLI была реализована посредством настройки прокси-сервера в конфигурационном файле Vue (vue.config.js), где указывался адрес и порт, на котором запущен сервер Uvicorn. Это позволяет клиентской части направлять запросы к API без возникновения проблем, связанных с политикой единого источника (CORS). Таким образом, запросы, поступающие с фронтенда, автоматически перенаправляются на соответствующие конечные точки серверной части. Такая интеграция позволяет создать согласованную среду разработки, где каждый компонент функционирует независимо, но при этом обеспечивает целостность системы в рамках единого проекта.

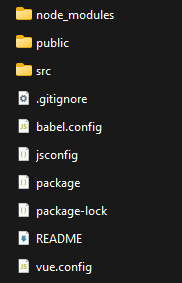


Рисунок 13 – Структура папки frontend/my-project

* 1. Разработка прототипа приложения

На текущем этапе проекта основное внимание уделялось разработке функционального прототипа платформы. Целью прототипирования являлось создание минимально жизнеспособной версии системы, которая демонстрировала бы базовые возможности приложения и служила фундаментом для дальнейшего развития функционала. Прототип позволяет пользователям регистрироваться и получать доступ к основным функциям, таким как планирование занятий, просмотр расписания и работа с домашними заданиями.

На данном этапе реализована следующая структура базы данных (рис.14): таблица пользователей (users) хранит данные о пользователях, включая имя, адрес электронной почты, пароль, роль (студент или преподаватель) и информацию о привязке к учебной группе. Таблица учебных групп (study\_groups) содержит сведения о группах, такие как уникальные коды, названия и дату создания. Таблица уроков (lessons) предусматривает хранение информации о проведённых занятиях, включая название, описание, ссылки на видео и дополнительные материалы, а также дату проведения, при этом уроки связаны с группами посредством промежуточной таблицы. Таблица домашних заданий (homeworks) фиксирует описание заданий, прикреплённые файлы, изображения, дату создания, текст задания и его статус, при этом задания также связываются с группами через промежуточную таблицу. Таблица отправленных домашних заданий (homework\_submissions) содержит данные о выполненных работах, включая время отправки, оценки, комментарии и статус проверки. Дополнительно предусмотрены таблицы для хранения файлов, прикреплённых к домашним заданиям и ответам преподавателя, таблица истории оценок, а также система управления токенами (refresh\_tokens), которая обеспечивает безопасное управление сессиями пользователей.

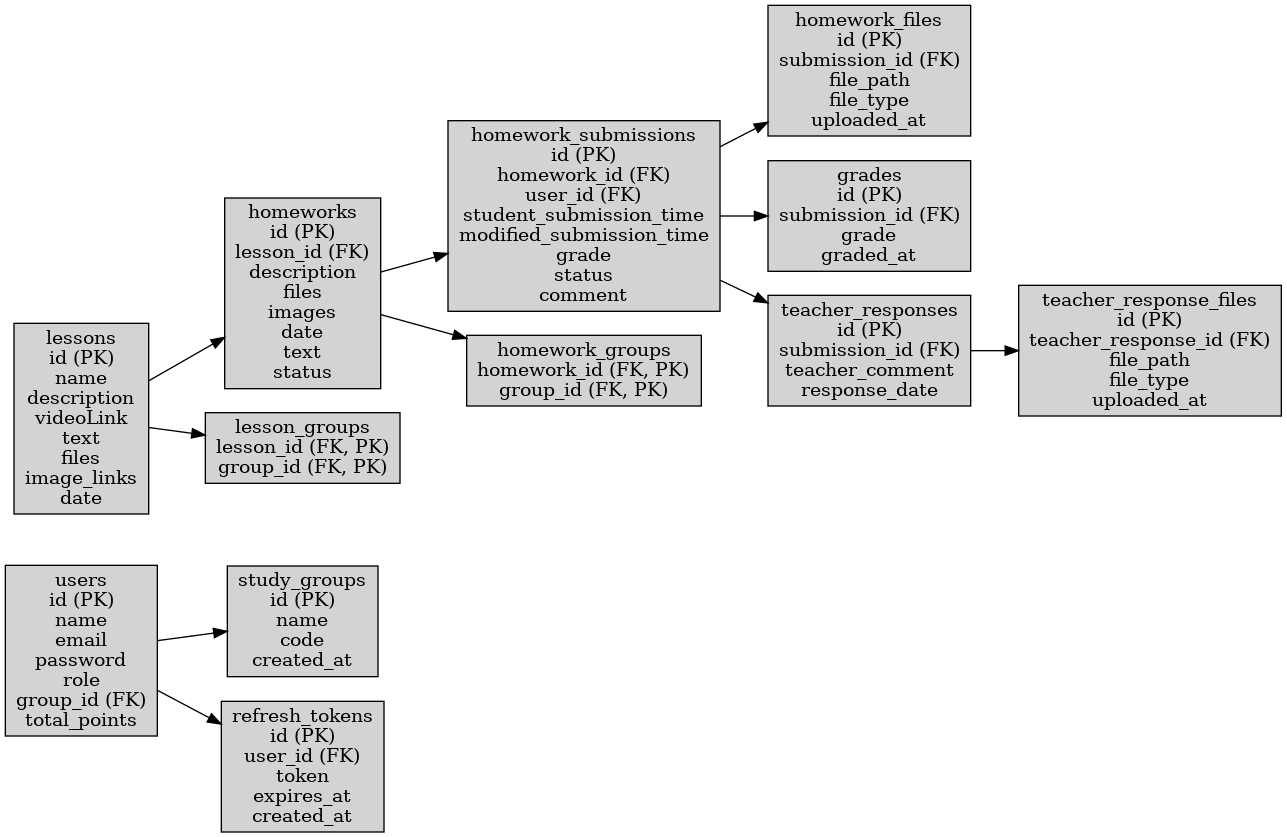


Рисунок 14 – Схема базы данных

MySQL скачана и настроена по инструкциям с официального сайта [10].

Одностраничное приложение (SPA) на Vue.js начинается с базового HTML-файла, который служит отправной точкой для всего интерфейса. В файле index.html определяется стандартная структура HTML-документа с обязательными метатегами, которые обеспечивают корректное отображение на различных устройствах и браузерах. Внутри тега <body> находится единственный контейнер с идентификатором app. Именно этот элемент предназначен для монтирования корневого Vue-компонента, что позволяет динамически подгружать и обновлять содержимое страницы без необходимости полной перезагрузки браузера.

После загрузки index.html происходит инициализация Vue-приложения, где корневым компонентом является файл App.vue. Этот компонент формирует базовый макет всей страницы, объединяя в себе такие элементы, как шапка (header) с логотипом и блоком информации о пользователе, а также основной контейнер для динамически загружаемых компонентов через тег <router-view>. В App.vue реализована базовая логика работы с авторизацией: при монтировании компонента происходит чтение данных пользователя из localStorage, установка заголовка авторизации для HTTP-запросов и регистрация обработчиков событий, таких как закрытие выпадающего меню при клике вне его области. Таким образом, App.vue задает основу одностраничного приложения, обеспечивая единое хранилище данных и навигацию между различными частями интерфейса.

Далее приложение строится из отдельных компонентов, каждый из которых отвечает за определённый функционал. Например, компонент, отображающий план на день, отвечает за получение и вывод списка запланированных уроков. При этом, благодаря реактивности Vue, любые изменения в данных (получение нового списка уроков с сервера) автоматически отражаются на интерфейсе, не требуя перезагрузки страницы. Компоненты бокового меню SideBar и шапка платформы интегрируются в основное представление через App.vue. Они являются статическими и не обновляются.

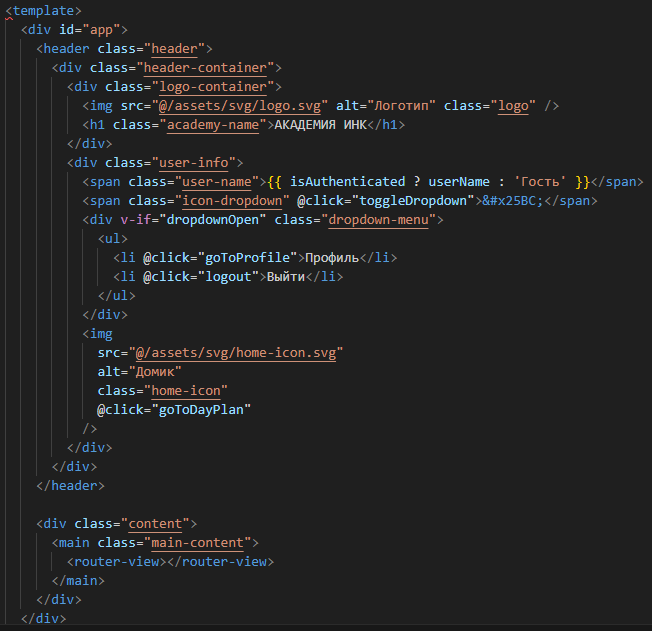


Рисунок 14 – Компонент App.vue

Сначала была разработана страница с формой регистрации и авторизации пользователей. Пользователь вводит адрес электронной почты, пароль и, в случае регистрации, своё имя. Все данные формы привязаны к состоянию компонента посредством директивы v-model, что обеспечивает мгновенное обновление данных и реактивность интерфейса. При отправке формы срабатывает метод, который формирует объект с данными и в зависимости от режима отправляет запрос на эндпоинты FastAPI – /login для входа и /register для регистрации. В случае успешного ответа от сервера, access token вместе с данными пользователя (такими как имя, роль, идентификатор и, при наличии, информация о группе) сохраняется в localStorage, а refresh token устанавливается в куки с повышенными настройками безопасности.



Рисунок 15 – Отправка запроса со страницы авторизации и получение данных от сервера

На серверной стороне логика регистрации и входа реализована через эндпоинты FastAPI. При входе система проверяет наличие пользователя в базе данных и корректность введённого пароля. Если данные подтверждены, сервер генерирует JWT access token, который живет 200 минут и refresh token, который истекает спустя день неактивности пользователя. Особенностью данного решения является использование refresh token, который хранится в защищённых HTTP-куки с параметрами httponly и samesite="Strict", что препятствует доступу к нему через JavaScript и снижает риск атак типа XSS. Аналогичная логика применяется при регистрации, где дополнительно происходит проверка на уникальность email, а пароль пользователя хэшируется перед сохранением. В обоих случаях сервер возвращает JSON-ответ с необходимыми данными для аутентификации и управления сессией.

Ключевым элементом защиты приложения является проверка авторизации каждого запроса. На сервере реализован middleware, который отслеживает входящие запросы и, если маршрут не входит в список исключений (например, /register, /login, /refresh-token, /docs), проверяет наличие действительного JWT токена. Если токен отсутствует или является недействительным, запрос не обрабатывается, что эффективно предотвращает несанкционированный доступ к защищённым страницам. Это означает, что пользователь, не прошедший аутентификацию, не сможет попасть на страницы приложения.

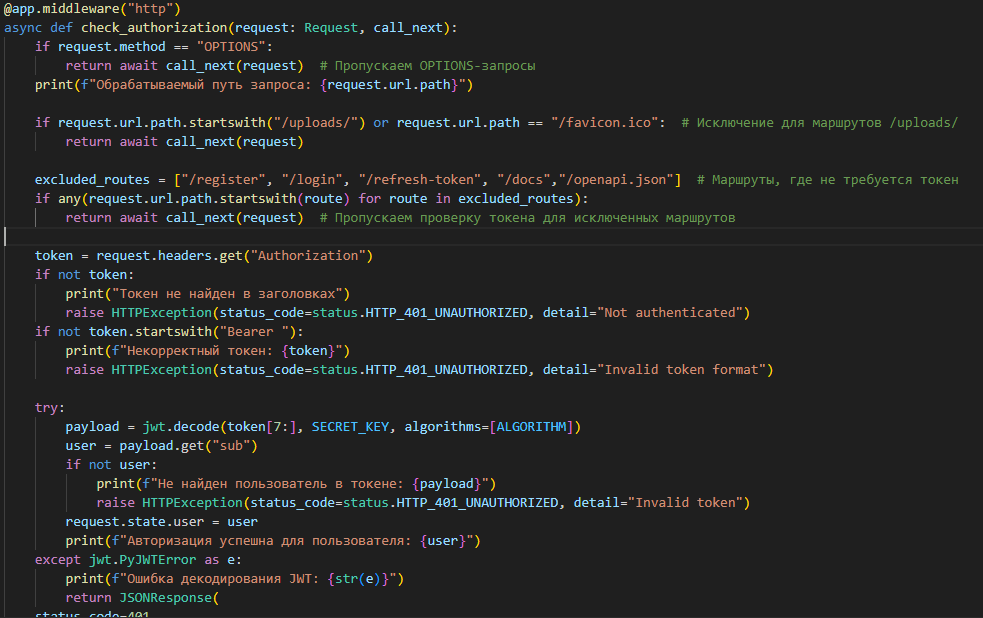


Рисунок 16 – Эндпоинт middleware

На клиентской стороне в файле main.js реализована глобальная настройка библиотеки axios для отправки запросов к серверу. В перехватчике запросов автоматически добавляется заголовок Authorization с access token, извлекаемым из localStorage, за исключением запросов на эндпоинт /refresh-token. Если сервер возвращает ошибку 401, что свидетельствует о том, что access token истёк или недействителен, срабатывает второй перехватчик, который инициирует процесс обновления токена. В этом случае происходит запрос к эндпоинту /refresh-token, и, при успешном получении нового access token, он сохраняется в localStorage и повторно выполняется исходный запрос. Если обновление токена не удаётся, система очищает данные пользователя из localStorage и перенаправляет его на страницу регистрации, что гарантирует, что неавторизованный пользователь не сможет продолжить работу в приложении.

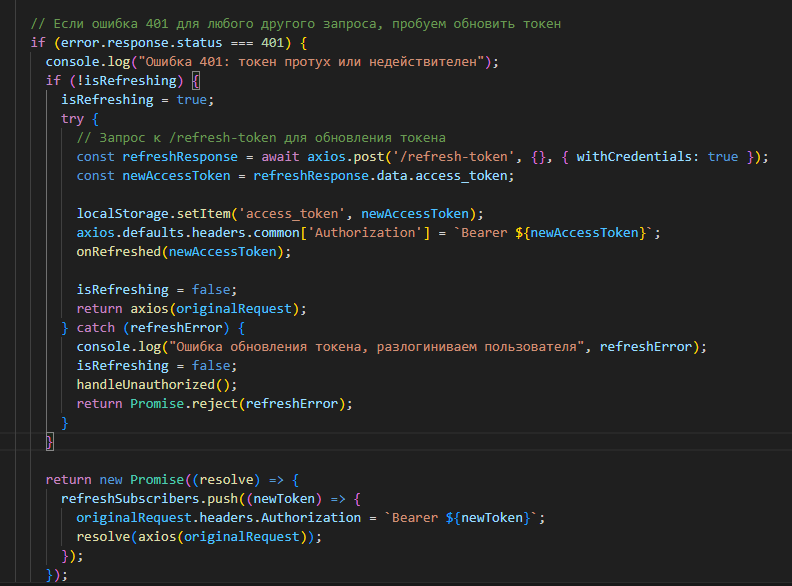


Рисунок 17 – Обновление токена авторизации в main.js

После авторизации пользователь будет перенаправлен на страницу «План на день», которая является основным домашней страницей. На данный момент на ней реализовано получение списка всех уроков с сервера, запланированных на «сегодня». В дальнейшем планируется доработать страницу таким образом, чтобы на ней отображались не только уроки, но и домашние задания, как на «сегодня», так и на ближайшие дни. Также на странице предусмотрен функционал, который позволяет преподавателю нажимать на кнопку с изображением плюса и запланировать новый урок.

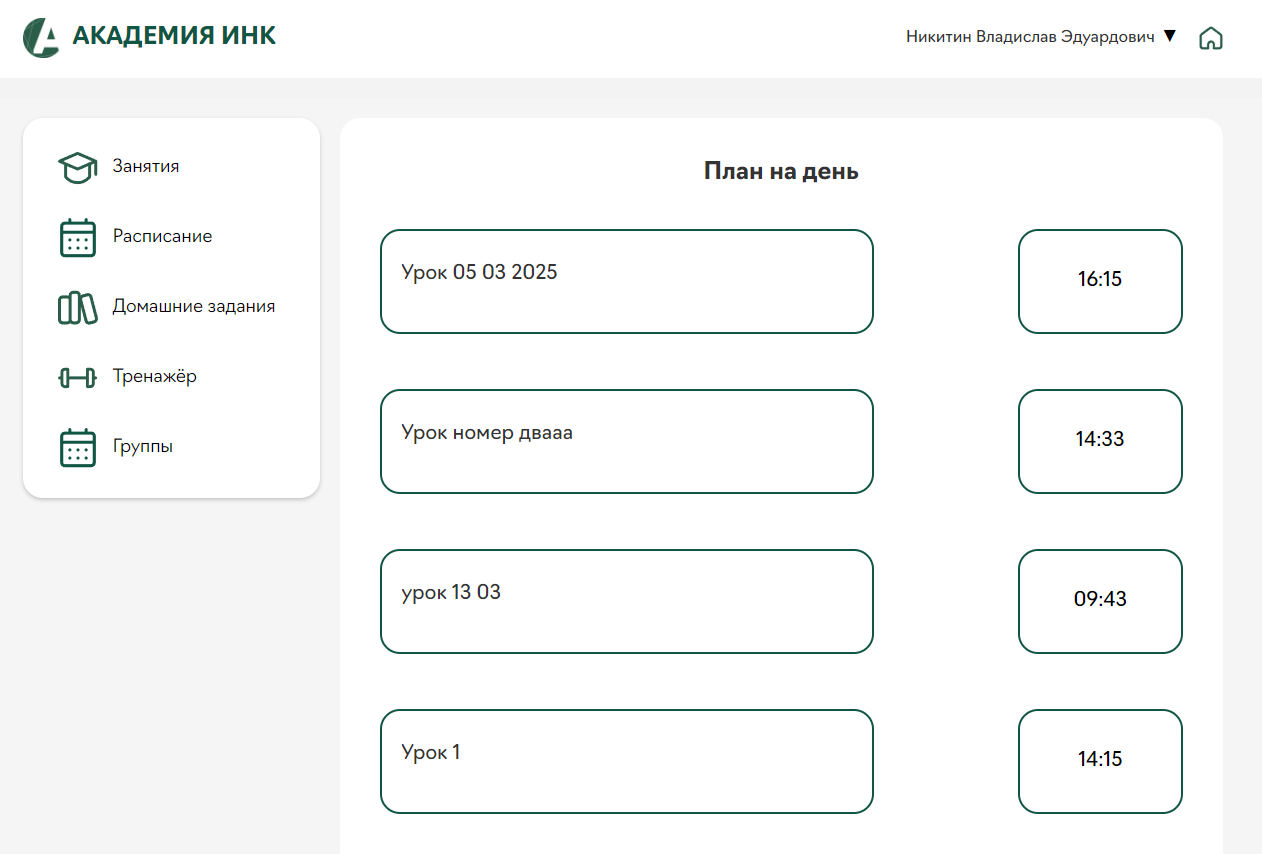


Рисунок 18 – Страница «План на день»

Разработка системы планирования занятий реализована с целью предоставить преподавателям удобный инструмент для создания и управления уроками, а студентам – оперативный доступ к расписанию и материалам занятий. В системе предусмотрена возможность добавления урока посредством специального интерфейса (рис. 19), который доступен преподавателю. Пользователь, являющийся учителем, заходит на страницу «Добавление урока», где через форму он вводит название и описание занятия, выбирает тип видео (обычное видео или стрим) и при необходимости указывает ссылку на видео (YouTube, VK) или вставляет HTML-код с iframe для стрима через ВК Видео. Форма дополнительно предусматривает возможность загрузки текстовых материалов, изображений и файлов, необходимых для проведения урока. В случае загрузки изображений или файлов предусмотрена поддержка drag-and-drop.

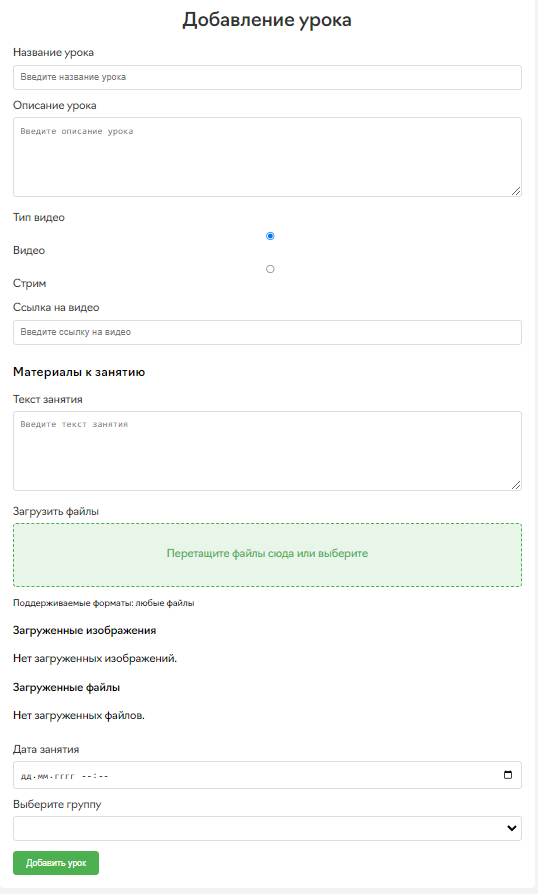


Рисунок 19 – Интерфейс добавления урока

При заполнении формы данные, введённые преподавателем, сохраняются в объекте состояния компонента, после чего происходит отправка данных на сервер с помощью метода handleSubmit. В этот момент происходит упаковка данных в формат FormData, что позволяет корректно передавать как текстовые данные, так и мультимедийные файлы. При отправке запроса используется эндпоинт FastAPI, предназначенный для создания нового урока. Сервер получает данные, сохраняет информацию о занятии в базе данных, а также создает отдельную папку для каждого урока, где организован процесс сохранения загруженных изображений и файлов. Для хранения файлов используется файловая система, а ссылки на файлы и изображения сохраняются в базе данных в виде строк, разделенных запятыми, которые затем преобразуются обратно в список для удобства дальнейшей работы.

На стороне сервера эндпоинт (рис. 20) обработки запроса реализован так, что помимо основных полей урока (название, описание, видео-ссылка, текст занятия, дата проведения) обрабатывается и выбор учебной группы, к которой относится урок. После успешного создания записи о занятии сервер дополнительно осуществляет поиск указанной группы и добавляет её в список групп урока. Пока занятия можно создавать только для одной группы. Далее создаются директории для хранения материалов урока, загруженные изображения сохраняются в отдельную подпапку, а файлы – в корневую папку урока. Полученные пути к изображениям и файлам нормализуются, объединяются в строку с разделителем и сохраняются в соответствующих полях базы данных.

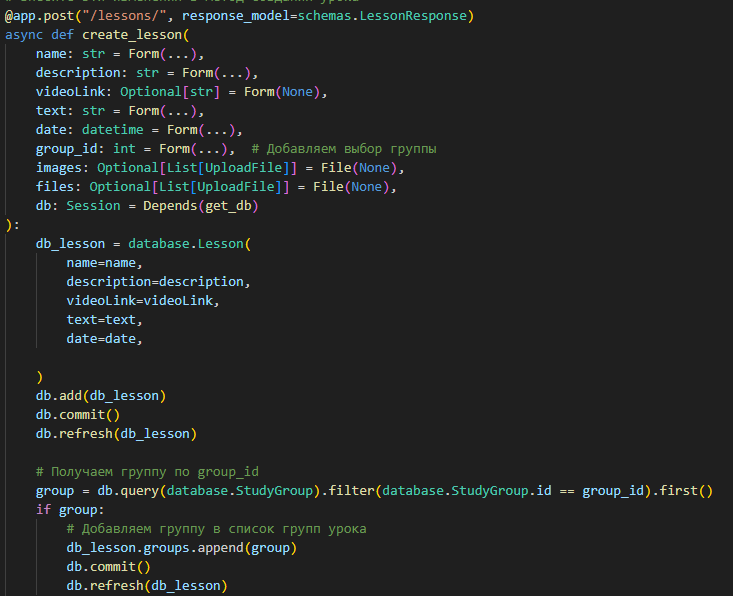


Рисунок 20 – Эндпоинт для добавления урока

Для удобства преподавателя и школьников была создана страница расписания в виде календаря, в котором отображаются занятия и домашние задания по датам. Пользователь может перемещаться между месяцами, а также переключаться между режимами просмотра занятий и домашних заданий. Интерактивные элементы, такие как индикаторы наличия событий и выпадающие списки, делают процесс взаимодействия с расписанием простым и наглядным.

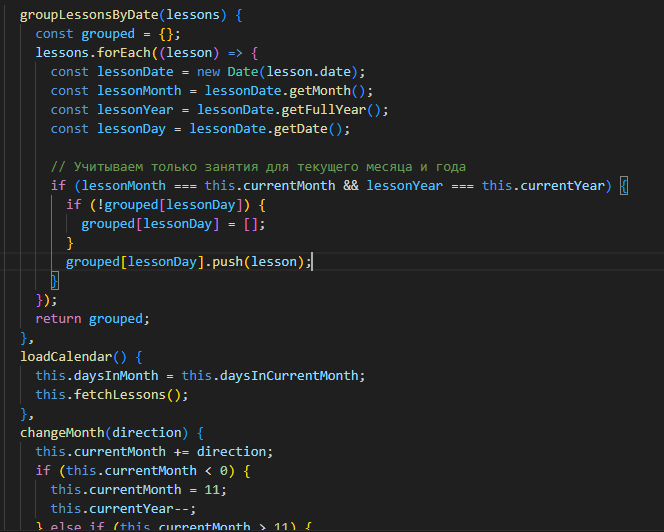


Рисунок 21 – Фрагмент кода, отвечающий за группировку занятий по датам, загрузку расписания и навигацию между месяцами в календаре

На серверной стороне данные о занятиях загружаются через эндпоинт /lessons, принимающий параметры месяца и года. В ответе сервер возвращает список занятий, которые затем группируются по датам и сохраняются в объекте lessonsByDate для удобного отображения. Этот механизм позволяет динамически загружать и обновлять данные при смене месяца. Аналогичным образом будет реализована загрузка домашних заданий в будущем.

Интерфейс расписания включает в себя стрелочки для переключения месяцев, блок дней недели и сетку календаря с ячейками для каждого дня. Если на определенную дату запланированы занятия, внутри ячейки появляется индикатор, а при наведении на день раскрывается список занятий. Пользователь может кликнуть на конкретное занятие, чтобы перейти к его деталям.

Переключение между режимами отображения занятий и домашних заданий реализовано с помощью кнопок "Занятия" и "ДЗ". Этот механизм можно расширить для удобства преподавателя, добавив дополнительные фильтры, например, отображение занятий или домашних заданий для определнных групп.

На клиентской стороне при изменении месяца автоматически выполняется новый запрос к API, чтобы загрузить соответствующую информацию. Такой подход минимизирует нагрузку на сервер и гарантирует, что пользователь всегда видит актуальные данные.

Система создания групп разработана для того, чтобы преподаватель мог организовать учебные группы, добавлять в них студентов и эффективно управлять распределением учащихся. Основной функционал реализован в компоненте StudentGroups, который отображает список уже созданных групп, привязанных к текущему преподавателю, и позволяет создавать новые группы.

На странице StudentGroups преподавателю представляется список групп в виде карточек, каждая из которых содержит название группы и уникальный код. При клике на карточку происходит выбор конкретной группы, после чего загружается список студентов, входящих в эту группу, с информацией о их имени, электронной почте и набранных баллах. Кроме того, предусмотрена возможность копирования кода группы: при нажатии на код он копируется в буфер обмена.

На странице присутствует форма создания новой группы. Преподаватель вводит название новой группы и нажимает кнопку «Создать группу». В этот момент вызывается метод, который проверяет корректность введённых данных, а затем с помощью axios отправляет POST-запрос на сервер, передавая название группы и идентификатор преподавателя. Сервер, используя FastAPI, генерирует уникальный код для новой группы, сохраняет её в базе данных и возвращает данные о созданной группе. После получения ответа интерфейс обновляет список групп, и преподавателю выводится сообщение об успешном создании группы вместе с её уникальным кодом.

На серверной стороне реализована логика управления группами. Эндпоинт POST /groups/ принимает название группы, генерирует уникальный код и сохраняет новую группу в базе данных. Дополнительно реализованы эндпоинты для получения списка всех групп, а также для получения списка студентов, входящих в конкретную группу.

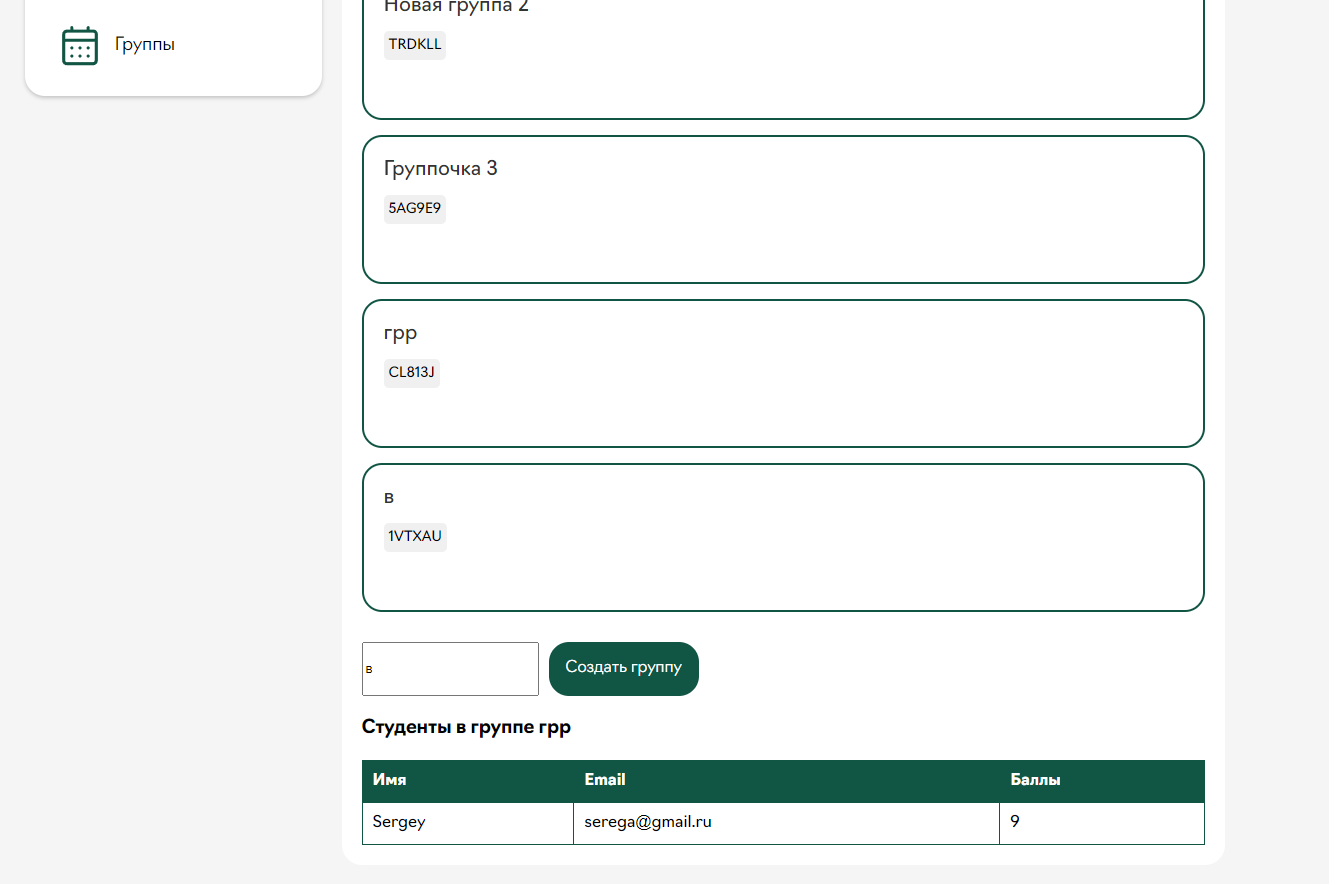


Рисунок 22 – Интерфейс компонента с группами

В реализованной странице профиля пользователя отображаются личные данные пользователя, такие как имя, электронная почта и информация о группе. Если студент еще не состоит в группе, на этой странице ему предоставляется возможность вступить в группу. Для этого в нижней части страницы отображается форма, где студент может ввести уникальный код группы. При нажатии на кнопку «Вступить» вызывается метод joinGroup, который проверяет корректность введенного кода и отправляет POST-запрос на сервер по соответствующему эндпоинту. Если сервер подтверждает успешное добавление студента в группу, обновляется информация о группе как в локальном состоянии компонента, так и в localStorage, что гарантирует синхронизацию данных. Кроме того, пользователю выводится сообщение об успешном вступлении, информирующее его о том, что он теперь является участником выбранной группы.

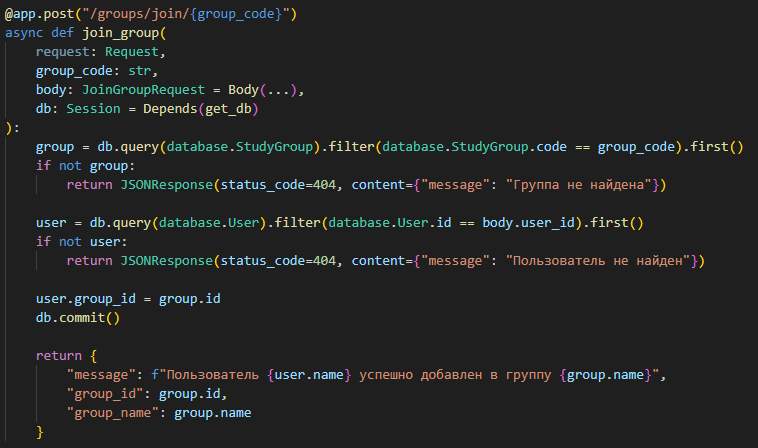


Рисунок 23 – Эндпоинт присоединения к группе

Была реализована система домашних заданий. Преподаватель может создавать задания для студентов, прикреплять к ним различные материалы, такие как текст, изображения и файлы, а затем редактировать или обновлять их при необходимости.

На серверной стороне создание домашнего задания происходит через эндпоинт POST /homeworks/. При отправке запроса сервер принимает данные урока (идентификатор урока, описание, текст задания, дату проведения), а также загруженные файлы и изображения, передаваемые через форму. Сначала проверяется наличие урока, к которому прикрепляется задание, и создается новая запись в базе данных. После этого домашнее задание автоматически связывается с группами, которым принадлежит урок. Для хранения загруженных материалов создается отдельная папка, в которой файлы и изображения сохраняются на сервере, а пути к ним преобразуются в строку в формате JSON для последующего удобного извлечения.

Для получения домашнего задания предусмотрены два эндпоинта. Один возвращает список всех заданий, а второй – конкретные задания, связанные с заданным уроком, что позволяет клиентской части динамически загружать и отображать актуальные данные. При этом файлы и изображения, сохраненные в базе данных в виде JSON-строк, преобразуются обратно в списки, чтобы их можно было корректно использовать в интерфейсе.

Редактирование домашнего задания реализовано через эндпоинт PUT /homeworks/{homework\_id} (рис. 24).

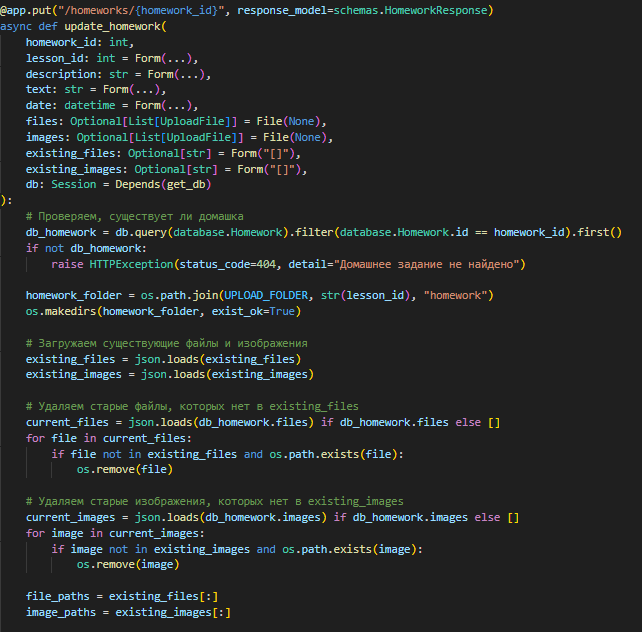


Рисунок 24 – Фрагмент эндпоинта изменения домашней работы преподавателем

При обновлении проверяется существование задания, затем создается папка для хранения материалов, если она еще не существует, и происходит обработка уже существующих файлов и изображений.

Если в обновленном запросе (рис. 25) отсутствуют некоторые ранее загруженные файлы, они удаляются с сервера, а новые файлы и изображения добавляются к уже имеющимся. Такой подход позволяет поддерживать актуальность и чистоту данных, а также обеспечивает возможность модификации домашнего задания без необходимости создания новой записи.



Рисунок 25 – Фрагмент компонента с отправкой обновленных данных на сервер

На клиентской стороне разработан отдельный интерфейс для создания домашнего задания. Страница добавления домашнего задания предоставляет преподавателю форму, где он может ввести название и текст задания, загрузить изображения и файлы (например, PDF-документы), а также установить дедлайн для выполнения задания. Форма поддерживает предварительный просмотр загруженных изображений и отображение списка прикрепленных файлов, что позволяет убедиться в корректности выбранных материалов до отправки данных на сервер. После успешной отправки формы данные передаются на сервер с использованием FormData, а при положительном ответе преподавателя перенаправляют на страницу урока.

Также реализована страница редактирования домашнего задания, которая позволяет преподавателю изменять существующие данные. Интерфейс редактирования напоминает форму добавления, но при этом загруженные материалы уже подгружаются с сервера с возможностью удаления. Преподаватель может удалить изображения или файлы, а затем загрузить новые, после чего обновленные данные отправляются на сервер через соответствующий эндпоинт PUT. В процессе редактирования система сравнивает старые и новые файлы, удаляя те, которые были исключены, и добавляя новые, что обеспечивает целостность данных и удобство управления домашними заданиями.

На странице компонента домашней работы (HomeworkDetails) отображается информация о задании, включая его название, описание и прикрепленные файлы. Студенты могут отправлять свои отклики на домашнее задание через модальное окно. В этом окне они могут загрузить файлы и оставить комментарий. Отклик можно редактировать до тех пор, пока преподаватель не поставит оценку. Когда оценка выставляется, кнопка редактирования становится неактивной и серой. Связанные с этим эндпоинты на сервере — это POST /submit\_homework для отправки отклика и PUT /update\_submission/{submission\_id} для редактирования отклика.

Преподаватель может просматривать отклики студентов на специальной странице, которая содержит список студентов, отправивших или не отправивших отклик. На этой странице преподаватель может открыть детали отклика студента, оставить комментарий, загрузить файлы с ответом и выставить оценку. Для этого используется компонент ShowSubmissions.vue, который запрашивает список откликов по конкретному домашнему заданию. Связанные с этим действием эндпоинты — это GET /homework\_responses/{homework\_id} для получения списка студентов и их статусов по домашнему заданию, GET /teacher\_response/{submission\_id} (рис. 24) для получения ответа преподавателя по отклику, и PUT /update\_teacher\_response/{submission\_id} для обновления комментария, файлов и оценки преподавателя.



Рисунок 24 – Эндпоинт для получения ответа преподавателя на отклик

Преподаватель также может прикреплять файлы в своем ответе на отклик студента. При обновлении ответа преподаватель может загрузить новые файлы или удалить ранее прикрепленные. Когда преподаватель удаляет файлы, сервер получает список файлов для удаления, проверяет их наличие на диске и удаляет их. Все эти действия обрабатываются через эндпоинт PUT /update\_teacher\_response/{submission\_id}, который управляет как комментариями, так и файлами преподавателя.

Преподаватель может выставить или изменить оценку студента. Когда оценка установлена, студент больше не может редактировать свой отклик. Если преподаватель обновляет оценку, система вычисляет разницу между новой и старой оценкой и корректирует total\_points студента. Если оценка удаляется, total\_points уменьшается. Эти действия реализованы в модальном окне проверки откликов преподавателя и обрабатываются через эндпоинт PUT /update\_teacher\_response/{submission\_id}.

Студент может увидеть комментарий преподавателя и выставленную оценку на странице деталей домашнего задания. Для этого используется эндпоинт GET /homeworks/{homework\_id}/feedback который возвращает комментарий и оценку для конкретного студента. Этот запрос выполняется при загрузке страницы HomeworkDetails.vue. Если у студента есть комментарий и оценка, они отображаются в интерфейсе.

## Выводы по главе

В данной главе выполнены задачи №4, №5 и №6.

Был разработан макет дизайна платформы, позволяющий визуализировать основную структуру и функциональные возможности приложения.

Был развернут локальный веб-сервер. Для тестирования прототипа была выбрана и настроена необходимая серверная инфраструктура с использованием FastAPI и Vue.js. Настроен механизм авторизации с использованием JWT, реализована безопасная передача данных и хранение refresh token в защищённых HTTP-куки.

Также создан прототип платформы, который закладывает прочный фундамент для дальнейшего расширения функционала. Прототип включает основные возможности, такие как регистрация и авторизация пользователей, планирование уроков, управление расписанием и работа с домашними заданиями. Реализованы функциональные модули для загрузки и хранения учебных материалов, что обеспечивает удобство работы как для преподавателей, так и для студентов. Этот начальный этап разработки заложил фундамент для дальнейшего расширения функционала платформы и оптимизации работы как клиентской, так и серверной частей системы.

# ЗАКЛЮЧЕНИЕ

В ходе работы была изучена проблематика и современные тенденции в сфере онлайн-образования, с особым акцентом на подготовку школьников к ЕГЭ. Проведен анализ существующих образовательных платформ, что позволило выделить ключевые функциональные требования и определить минимальный набор возможностей, необходимых для эффективного обучения. Это дало возможность сформировать обоснованное техническое задание для разработки специализированного веб-приложения.

Была освоена современная технологическая база, включающая Python FastAPI для реализации серверной логики, Vue.js для создания динамичного клиентского интерфейса и MySQL для организации хранения данных. В результате проведенной работы был разработан макет дизайна платформы, отражающий продуманную структуру и удобный пользовательский интерфейс, что является важным шагом для обеспечения интуитивного взаимодействия пользователей с системой.

Развертывание локального веб-сервера позволило не только протестировать работу отдельных модулей, но и создать полноценную тестовую среду для интеграции клиентской и серверной частей приложения. Реализованный прототип включает ключевые функциональные блоки, такие как регистрация и авторизация пользователей, планирование занятий, работа с домашними заданиями и управление учебными группами.

Таким образом, общий результат практики демонстрирует успешное освоение и применение современных технологий веб-разработки, что позволило не только выполнить поставленные задачи, но и создать прототип, способный эффективно поддерживать образовательный процесс. Разработанные решения будут использованы для дальнейшего развития и масштабирования образовательного проекта.

Ежедневные записи можно увидеть в приложении А. Ссылки на разработанные материалы в приложении Б.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. REST API Tutorial [Электронный ресурс] URL: https://restfulapi.net/ (дата обращения 18.02.2025).
2. Официальная документация Vue.js [Электронный ресурс] URL: https://vuejs.org/ (дата обращения 19.02.2025).
3. React vs Vue vs Angular [Электронный ресурс] URL: https://habr.com/ru/companies/auriga/articles/703836/ (дата обращения 19.02.2025).
4. Django, Flask или FastAPI: что выбрать для изучения? [Электронный ресурс] URL: https://codelab.pro/django-flask-ili-fastapi-chto-vybrat-dlya-izucheniya/ (дата обращения 20.02.2025).
5. PostgreSQL или MySQL: какая из этих реляционных СУБД лучше впишется в ваш проект [Электронный ресурс] URL: https://cloud.vk.com/blog/postgresql-ili-mysql-kakaya-iz-etih-relyacionnyh-subd/ (дата обращения 20.02.2025).
6. The best ORMs for database-powered Python apps [Электронный ресурс] URL: https://www.infoworld.com/article/2335270/6-orms-for-every-database-powered-python-app.html (дата обращения 21.02.2025).
7. Сайт Академии ИНК [Электронный ресурс] URL: https://academy-ink.isu.ru/ (дата обращения 21.02.2025).
8. Официальная документация Fast API: https://fastapi.tiangolo.com/ (дата обращения 28.02.2025).
9. Официальная документация Vue CLI [Электронный ресурс] URL: https://cli.vuejs.org/ru/ (дата обращения 01.03.2025).
10. Официальная документация MySQL [Электронный ресурс] URL: https://dev.mysql.com/doc/ (дата обращения 02.03.2025).

# ПРИЛОЖЕНИЯ

## Приложение А

## Ежедневные записи студента по практике

|  |  |  |
| --- | --- | --- |
| Дата | Описание работы, выполненной студентом | Отметка руководителя |
| 18.02 | Анализ существующих решений по подготовке ЕГЭ, выявление требований к проектируемой платформе |  |
| 19.02 – 20.02 | Ознакомление с инструментами разработки. |  |
| 21.02 – 27.02 | Разработка макета дизайна платформы |  |
| 28.02 – 01.03 | Развертывание локального веб-сервера |  |
| 03.03 – 05.03 | Разработка домашней страницы и регистрации |  |
| 06.03 – 12.03 | Разработка системы планирования занятий |  |
| 12.03 – 15.03 | Разработка расписания |  |
| 17.03 – 22.03 | Разработка системы домашний заданий и системы групп школьников |  |
| 24.03 – 26.03 | Разработка системы авторизации пользователей с JWT токенами |  |
| 27.03 | Написание отчёта и создание презентации |  |
| 28.03 | Прохождение нормоконтроля |  |
| 29.03 | Защита отчета по практике |  |

Отзыв руководителя:

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Приложение Б

Ссылки на разработанные материалы

|  |  |
| --- | --- |
| Описание | Ссылка |
| GitHub репозиторий с макетом дизайна | https://github.com/Vloodek/ege\_platform/tree/main/База/Макет%20дизайна |
| GitHub репозиторий платформы | https://github.com/Vloodek/ege\_platform/tree/main |