

МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
“ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ”
(ФГБОУ ВО «ВГУ»)

Факультет компьютерных наук

Кафедра программирования и информационных технологий

Платформа для дистанционного управления учебным процессом «Ratingus»

Курсовой проект

по дисциплине

Технологии программирования

09.03.04. Программная инженерия

Информационные системы и сетевые технологии

Преподаватель _____ В.С. Тарасов, ст. преподаватель _____.20__

Обучающийся _____ Д.Г. Шлыков, 3 курс, д/о

Обучающийся _____ П.А. Сапегин, 3 курс, д/о

Обучающийся _____ Е.А. Саков, 3 курс, д/о

Руководитель _____ Е.Д. Проскуряков, ассистент

Воронеж 2024

СОДЕРЖАНИЕ

СОДЕРЖАНИЕ	2
ВВЕДЕНИЕ	4
1 Обзор предметной области.....	6
1.1 Глоссарий	6
1.2 Актуальность работы	9
1.3 Обзор аналогов	12
1.3.1 Балльно-рейтинговая система для факультета компьютерных наук Воронежского государственного университета.....	12
1.3.2 Дневник.ру	13
1.3.3 Расписание занятий – Weeklie	15
1.3.4 Итоги анализа аналогов	17
1.4 Постановка задачи.....	18
1.4.1 Функциональные требования	19
2 Реализация.....	22
2.1 Архитектура платформы	22
2.2 Средства реализации.....	23
2.2.1 Серверная часть	24
2.2.2 Клиентская часть (веб-приложение)	24
2.2.3 Клиентская часть (мобильное приложение).....	25
2.3 Серверная часть.....	25
2.3.1 Механика работы сервера	25
2.3.2 Слой контроллеров	26
2.3.3 Слой бизнес-логики	26
2.3.4 Слой доступа к данным	27
2.3.5 Модели и DTO	27
2.3.6 База данных.....	30
2.4 Клиентская часть (мобильное приложение).....	32
2.5 Клиентская часть (веб-приложение)	39
2.5.1 Неавторизованный пользователь.....	39
2.5.2 Авторизованный пользователь	40
2.5.3 Ученик	41

2.5.4 Учитель.....	42
2.5.5 Локальный администратор.....	45
2.5.6 Менеджер платформы	46
2.6 Развертывание	47
3 Анализ проделанной работы.....	49
3.1 Тестирование	49
3.2 Сбор метрик	50
ЗАКЛЮЧЕНИЕ	55
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	56
ПРИЛОЖЕНИЕ А	57

ВВЕДЕНИЕ

Неотъемлемой частью процесса обучения каждого является контроль знаний и отслеживание выполняемой работы. Современный человек обучается постоянно: сначала это школа, школы искусств, затем высшие или средние учебные заведения, различного рода курсы и тренинги.

Никакое обучение невозможно проводить без получения обратной связи по уровню знаний: практически в любом образовательном учреждении присутствуют те или иные формы оценок, которые помогают понять, насколько усвоен пройденный материал. Также обычно в процессе обучения преподаватель постоянно выдаёт домашнее задание для закрепления пройденного материала. И, конечно, всегда есть расписание, согласно которому проводятся обучающие занятия.

К сожалению, на сегодняшний день нет единого стандарта, по которому учебные организации предоставляли бы доступ к оценкам, расписанию и домашнему заданию своим ученикам. Иногда это просто фотографии и сообщения в мессенджерах, с помощью которых неудобно решать подобные задачи. Кроме того, от школы к школе могут использоваться разные платформы для образования. А платформы школ и вузов различаются кардинально.

Выход из сложившейся ситуации – разработка единого сервиса, включающего в себя сайт и мобильное приложение для контроля успеваемости и отслеживания необходимой учебной информации. Это должно быть решение, которое в перспективе может стать единым решением для учебных организаций любого рода.

С каждым днём информационные технологии становятся всё более используемыми и близкими людям. С их помощью получается упрощать сложные процессы, легко налаживать коммуникацию и получать самую разнообразную информацию. Эти же технологии могут и должны упростить

контроль процесса обучения школьникам и студентам. Актуальность этого заявления подтверждается опросом, результаты которого представлены в разделе «Обзор предметной области».

1 Обзор предметной области

1.1 Глоссарий

В настоящей курсовой работе применяются следующие термины с соответствующими определениями:

- Авторизация – предоставление определённого лицу или группе лиц прав на выполнение определённых действий, а также процесс проверки данных прав при попытке выполнения этих действий.
- Аккаунт – хранимая в компьютерной системе совокупность данных о пользователе, необходимая для его опознавания и предоставления доступа к его личным данным и настройкам.
- Клиент – аппаратный или программный компонент вычислительной системы, посылающий запросы серверу (в контексте настоящего документа клиент – это браузер или мобильное приложение пользователя сервиса).
- Код приглашения – то же, что уникальный одноразовый код.
- Контроллер – это класс, предназначенный для непосредственной обработки запросов от клиента и возвращения результатов.
- Лендинг – страница сайта или экран приложения, которые призывают пользователя что-то сделать.
- Логин – имя пользователя, которое выступает в качестве идентификатора пользователя (или идентификатора учётной записи).
- Локальный администратор – одна из возможных ролей, которой может обладать пользователь отдельно в рамках каждой виртуальной организации. Пользователь с данной ролью занимается управлением и настройкой виртуальной учебной организации.

- Маппер – вспомогательный класс, предназначенный для преобразования объектов между классами.
- Менеджер платформы – одна из пользовательских ролей на разрабатываемой платформе. Пользователь с данной ролью занимается созданием учебных организаций.
- Мессенджер – это программа для мгновенного обмена текстовыми сообщениями, файлами и медиа между зарегистрированными пользователями через интернет.
- Паттерн проектирования – часто встречающееся решение определённой проблемы при проектировании архитектуры программ.
- Репозиторий – класс, который обеспечивает связь с хранящимися в базе данными.
- Роль (пользователя) – абстракция, принятая для деления пользователей на группы по уровням доступа и возможностей.
- Сервер – программный компонент вычислительной системы, выполняющий сервисные (обслуживающие) функции по запросу клиента, предоставляя ему доступ к определённым ресурсам или услугам.
- Сервис – разрабатываемый программный продукт, который рассматривается с точки зрения функций и возможностей, предоставляемых пользователю.
- Уникальный одноразовый код (код приглашения) – уникальный код, используемый внутри платформы. Представляет из себя последовательность символов и генерируется менеджером платформы или локальным администратором. С помощью ввода

кода пользователь может подключиться к учебной организации. После первого использования код становится недействительным.

- Ученик – одна из возможных ролей, которой может обладать пользователь отдельно в рамках каждой виртуальной организации. Пользователь с данной ролью может просматривать информацию об оценках, посещаемости, выданном домашнем задании, а также расписание в рамках виртуальной учебной организации, к которой имеет доступ.
- Учитель – одна из возможных ролей, которой может обладать пользователь отдельно в рамках каждой виртуальной организации. Пользователь с данной ролью может в рамках виртуальной учебной организации, к которой имеет доступ, заполнять информацию об оценках и посещаемости своих учеников, а также выдавать им домашнее задание.
- API (Application Programming Interface) – описание способов взаимодействия одной компьютерной программы с другими.
- Backend – это серверная часть веб-приложения, которая занимается обработкой данных, взаимодействием с базами данных и выполнением бизнес-логики.
- CI/CD – это комбинация непрерывной интеграции и непрерывного развертывания программного обеспечения в процессе разработки.
- DTO (Data Transfer Object) – один из шаблонов проектирования, используется для передачи данных между подсистемами приложения.

— Frontend – это публичная часть web-приложений (вебсайтов), с которой пользователь может взаимодействовать и контактировать напрямую.

— FSD (Feature-Sliced Design) – это архитектурная методология для проектирования клиентской части приложений.

1.2 Актуальность работы

Для оценки актуальности работы были проведены исследования, результаты которых представлены ниже.

На рисунках 1 и 2 содержатся диаграммы, из которых можно сделать вывод: и школьники, и студенты считают, что платформы для дистанционного управления учебным процессом упрощают учёбу.

Кто вы?

51 ответ

 Копировать

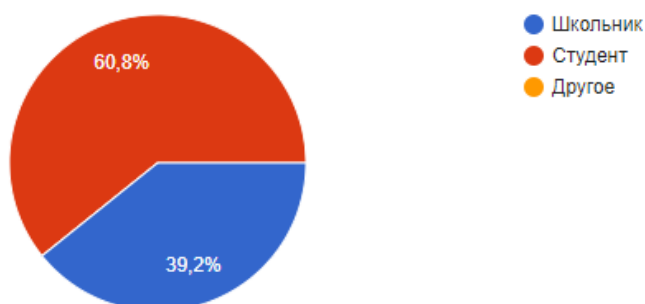


Рисунок 1 - Диаграмма, отражающая контингент участников опроса

Считаете ли Вы, что платформы для дистанционного управления учебным процессом делают обучение более удобным?

 Копировать

51 ответ

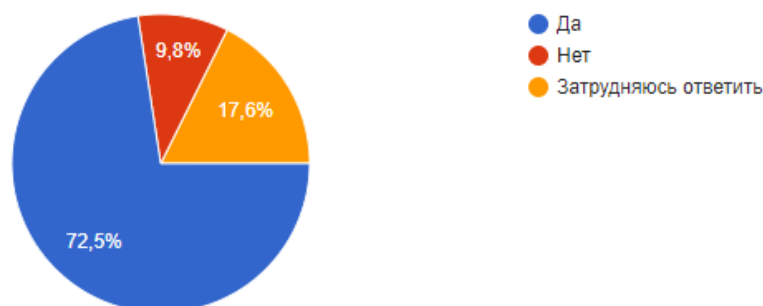


Рисунок 2 - Диаграмма, отражающая удобство использования информационных технологий для контроля процесса обучения

Также заинтересованность учащихся в создании платформы для дистанционного управления учебным процессом видно из последних двух вопросов формы. Например, на рисунке 3 отображены результаты ответов на вопрос о желаемых функциях приложения.

Представьте, что есть приложение, которое призвано помочь Вам в организации Вашего обучения. Какие функции Вы бы хотели в нём видеть?

 Копировать

51 ответ

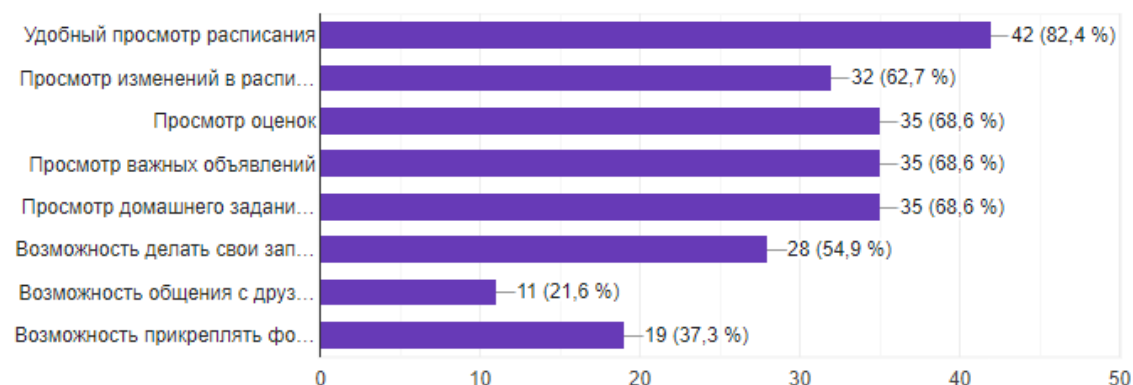



Рисунок 3 - Диаграмма ответов на вопрос о желаемых функциях в приложении для контроля учебного процесса

Варианты ответов сверху вниз:

- удобный просмотр расписания;
- просмотр изменений в расписании;
- просмотр оценок;
- просмотр важных объявлений;
- просмотр домашнего задания, записанного преподавателем;
- возможность делать свои записи в дневник;
- возможность общения с друзьями;
- возможность прикреплять фотографии выполненного домашнего задания.

Результаты ответов на вопрос «Стали ли бы Вы пользоваться приложением с такими функциями?» представлены на рисунке 4.

Стали ли бы Вы пользоваться приложением с такими функциями?

 Копировать

51 ответ

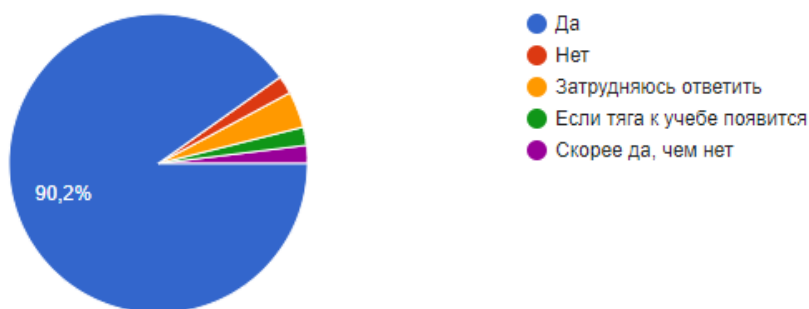


Рисунок 4 - Диаграмма, показывающая заинтересованность обучающихся в приложении для контроля процесса обучения

Итак, из результатов опроса видно, что учащиеся действительно заинтересованы в платформе для дистанционного управления учебным процессом. Согласно рисунку 3, наиболее важными функциями для учащихся являются:

- удобный просмотр расписания;
- просмотр оценок;
- просмотр важных объявлений;
- просмотр домашнего задания, записанного преподавателем.

В данной курсовой работе пойдёт речь о создании платформы для контроля учебных процессов. Результаты, полученные при проведении опроса, послужат отправной точкой для формулировки цели работы.

1.3 Обзор аналогов

1.3.1 Балльно-рейтинговая система для факультета компьютерных наук Воронежского государственного университета

Балльно-рейтинговая система (БРС) – это онлайн-платформа в виде сайта для факультета компьютерных наук Воронежского государственного университета. Она имеет очень простой, понятный интерфейс и предназначена для отслеживания успеваемости студентов [1]. Главная страница сайта представлена на рисунке 5.

Главная

Списки студентов

Староста / Куратор

Рейтинг студентов

Выход

Свод оценок студента

Ф.И.О.:

Курс: 3

Семестр: 6

Группа: 7 (1)

Направление / специальность:

09.03.04 Программная инженерия(Информационные системы и сетевые технологии) Бакалавр(ФГОС3++)

Учебный год	Семестр	Курс	Предмет	Отчётность	Преподаватель	1	2	3	% посещ.	взеш. балл	Экзамен	Доп. балл	Итог. балл	Итог
2023-2024	6	3	Физика (Б1.О.18)	п Экзамен	Крыловецкая Т. А.				—					
2023-2024	6	3	Информационные сети (Б1.О.28)	п Экзамен	Коваль А. С.				100.0					
2023-2024	6	3	Информационные технологии (Б1.О.38)	п Экзамен	Вахтин А. А.				100.0					
2023-2024	6	3	Технологии программирования (Б1.О.39)	п Зачёт с оценкой	Тарасов В. С.				—		—			
2023-2024	6	3	Теория компиляторов (Б1.В.04)	п Зачёт	Соломатин Д. И.				—	—	—	—		
2023-2024	6	3	Основы автоматизированного проектирования (Б1.В.05)	п Зачёт	Чижов М. И.				—	—	—	—		
2023-2024	6	3	Основы DevOps (Б1.В.13)	п Зачёт	Косых Е. В.				—	—	—	—		
2023-2024	6	3	Язык программирования C++ (Б1.В.ДВ.04.01)	п Зачёт	Лысачев П. С.				—	—	—	—		

Рисунок 5 - Главная страница БРС

БРС – это платформа для студентов ВГУ факультета компьютерных наук. С одной стороны, эта особенность позволяет оптимизировать функционал платформы под конкретные нужды, плотно интегрировать её в учебный процесс, с другой – ей не могут пользоваться другие учебные организации. Это является серьёзным минусом.

В платформе можно выделить следующие положительные моменты:

- она позволяет сохранить список студентов в виде файла .xlsx список студентов своей группы;
- посмотреть ФИО и контакты старосты, заместителя старосты и куратора группы;
- есть рейтинг студентов, где авторизованный пользователь может посмотреть рейтинг всех студентов факультета определённого курса за любой семестр.

Однако БРС не предоставляет таких функций, как:

- просмотр оценок и посещаемости за отдельные пары;
- просмотр расписания;
- просмотр и запись домашнего задания.

Здесь можно просматривать лишь общую статистику посещаемости и оценки за рубежные аттестации и экзамены или зачёты.

1.3.2 Дневник.ру

Дневник.ру — цифровая образовательная платформа, которая делает образование в России качественным и доступным [2].

Сайт Дневник.ру предоставляет все необходимые функции для контроля процесса обучения (оценки, расписание, домашнее задание). Отсутствует только возможность самостоятельной записи домашнего задания.

Также к плюсам можно отнести:

- наличие у родителей аккаунтов, что позволяет контролировать успеваемость ребёнка;
- можно менять статус домашнего задания («Выдано», «В работе» и др.);
- можно задавать вопросы учителям;
- есть функции аналитики оценок и рейтинга в классе (платная).

Однако у Дневника есть несколько серьёзных минусов:

- предназначен для школьников и их родителей. Вузы не могут использовать его в своих целях;
- Дневник представляет собой ещё и социальную сеть, что излишне перегружает его;
- мобильное приложение Дневника предоставляет большую часть необходимого ученикам функционала только за платную подписку

На рисунке 6 представлен скриншот главного экрана приложения Дневника.

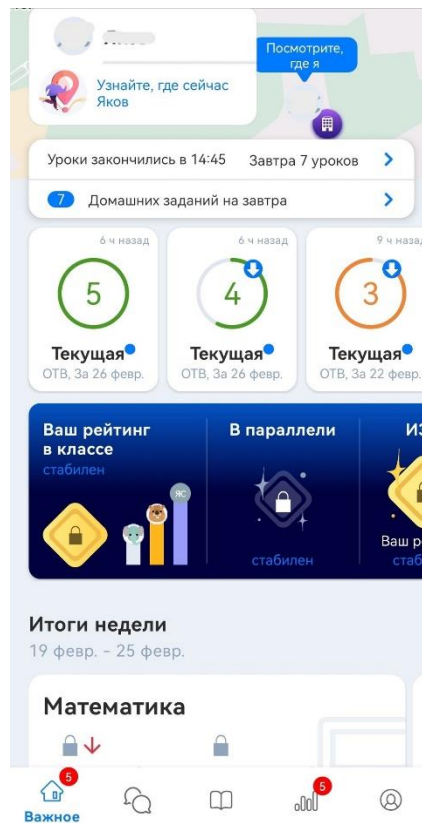


Рисунок 6 - Скриншот главного экрана Дневника

1.3.3 Расписание занятий – Weeklie

Сервис представляет из себя простое приложение для мобильных устройств, основной функционал которого заключается в заполнении расписания и затем его просмотра и редактирования [3]. На рисунках 7 и 8 представлены скриншоты интерфейса приложения.

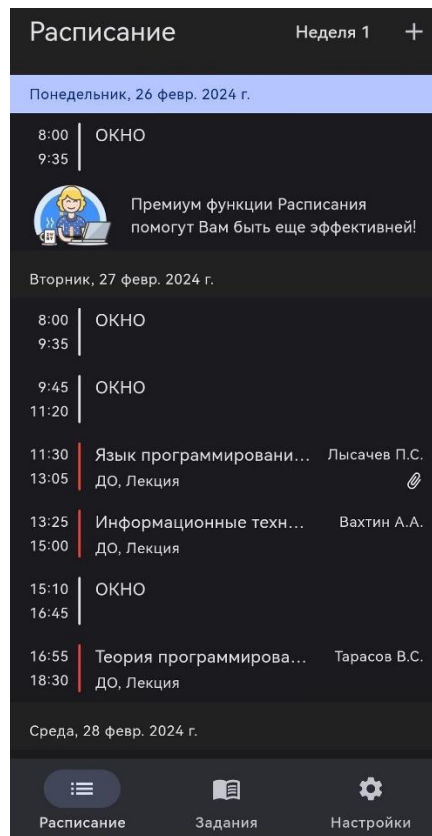


Рисунок 7 - Скриншот приложения Weeklie (экран расписания)

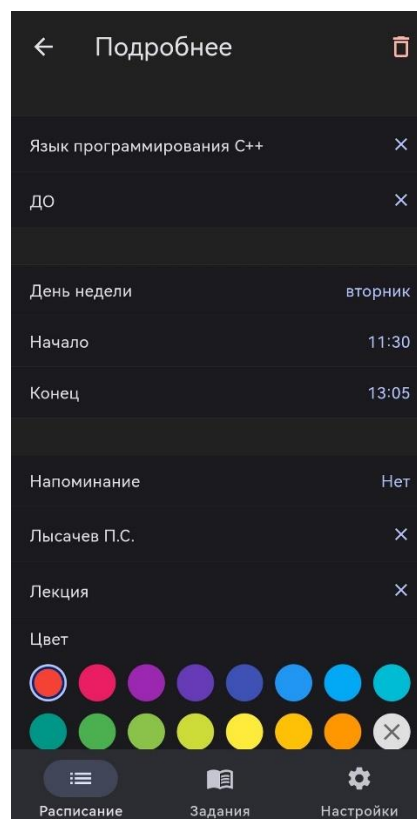


Рисунок 8 - Скриншот приложения Weeklie (экран предмета)

Weeklie – стороннее приложение, никак не связанное с учебными организациями.

К его плюсам можно отнести следующие аспекты:

- имеет простой и удобный интерфейс;
- позволяет создавать расписание, указывая ФИО преподавателя, номер аудитории и др.;
- позволяет делиться расписанием с другими людьми;
- пользователь может самостоятельно вносить домашнее задание;
- с помощью цвета можно дополнительно структурировать информацию в приложении;
- использовать приложение может каждый желающий;
- есть функция напоминаний (за плату).

Но у приложения есть и ряд существенных недостатков:

- оно никак не связано с учебными организациями;
- в нём нет никаких данных о посещаемости, оценках;
- расписание нужно периодически обновлять, чтобы поддерживать в актуальном состоянии;
- домашнее задание в приложении может вносить только сам учащийся.

1.3.4 Итоги анализа аналогов

Есть немало приложений и платформ, которые решают те или иные выделенные проблемы, но совокупно все они не покрываются ни одним сервисом.

Дневник.ру, который потенциально является прямым конкурентом, предназначен только для школьников, не имеет необходимого бесплатного функционала в мобильном приложении, а также имеет лишний функционал социальной сети.

Следовательно, учащиеся не могут с помощью какой-либо платформы иметь актуальные сведения об оценках, посещаемости, домашнем задании и расписании. Потребность в разрабатываемом сервисе является актуальной.

1.4 Постановка задачи

Цель данного курсового проекта – разработка сайта и мобильного приложения, предназначенных для контроля и структуризации учащимися учебных процессов. Цель создаваемой платформы – предоставить учащимся удобный доступ к учебной информации, которую смогут загружать работники учебных организаций.

Под учебной информацией подразумевается следующая информация:

- расписание занятий для учащихся;
- выданное домашнее задание;
- поставленные оценки и отмеченная посещаемость;
- важные объявления.

Задача курсовой работы – предоставить учащимся доступ к этой информации.

Важно отметить, что платформа должна также обладать функциями, которые будут позволять загружать на неё необходимую информацию. Эта функциональность будет реализована в веб-приложении.

Мобильное приложение, в свою очередь, будет предназначено только для учащихся.

На платформе будет реализована следующая система ролей с различными правами:

- «Менеджер платформы»;
- «Локальный администратор»;
- «Учитель»;
- «Ученик»;
- «Авторизованный пользователь»;
- «Неавторизованный пользователь».

1.4.1 Функциональные требования

Выделим функциональные требования к разрабатываемой платформе.

Для работников учебных организаций на сайте должны быть представлены следующие возможности:

- выставление оценок учащимся;
- отметка посещаемости учащихся;
- создание и изменение расписания уроков;
- выдача домашнего задания;
- написание и удаление объявлений;
- добавление и удаление учащихся в учебную организацию;
- изменение данных об учебной организации.

Для учащихся в мобильном приложении должны быть представлены следующие возможности:

- просмотр своих оценок и посещаемости;

- просмотр домашнего задания;
- запись домашнего задания;
- просмотр расписания;
- просмотр объявлений.

Рисунки А.1, А.2, А.3, А.4, А.5 в приложении А представляют из себя диаграммы вариантов использования для пользователей различных ролей.

Немаловажным аспектом работы платформы является создание учебной организации. За создание учебных организаций отвечает Менеджер платформы.

Этот процесс в виде диаграммы активности представлен на рисунке 9.

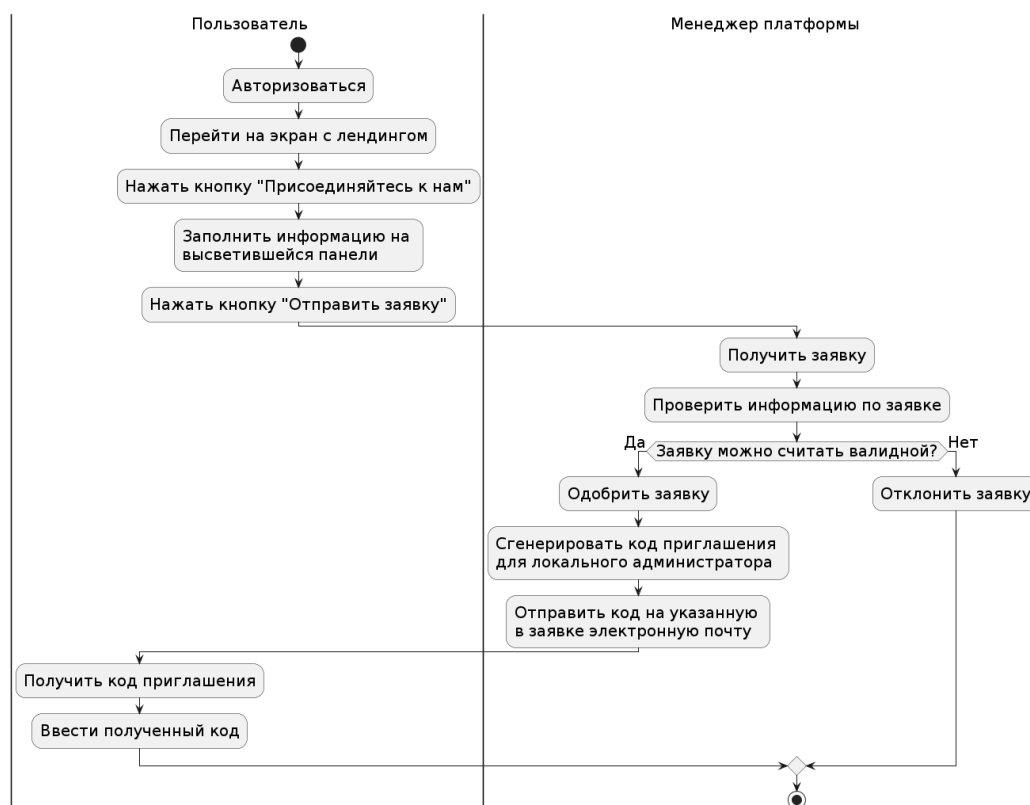


Рисунок 9 - Диаграмма активности, отображающая процесс создания учебной организации

На рисунке 10 представлена диаграмма сотрудничества, которая отображает взаимодействие пользователей различных ролей между собой.

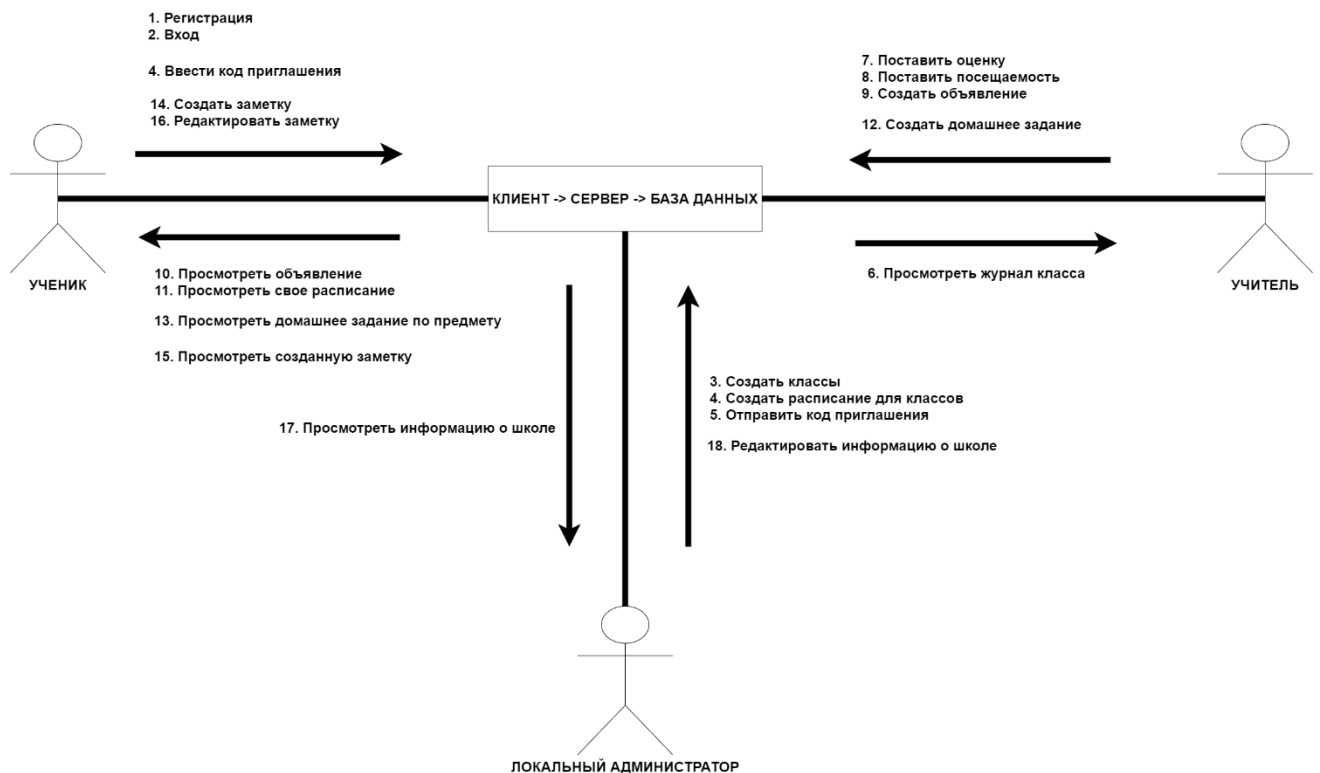


Рисунок 10 - Диаграмма сотрудничества пользователей с различными ролями

2 Реализация

2.1 Архитектура платформы

Платформа реализована в соответствии с клиент-серверной архитектурой на основе REST API. Мобильное приложение и сайт (в качестве клиентов) могут обмениваться данными с сервером. Сервер обрабатывает запросы, при необходимости делая запросы в базу данных, и возвращает ответ. Схематичное изображение структуры архитектуры платформы представлено на рисунке 11.

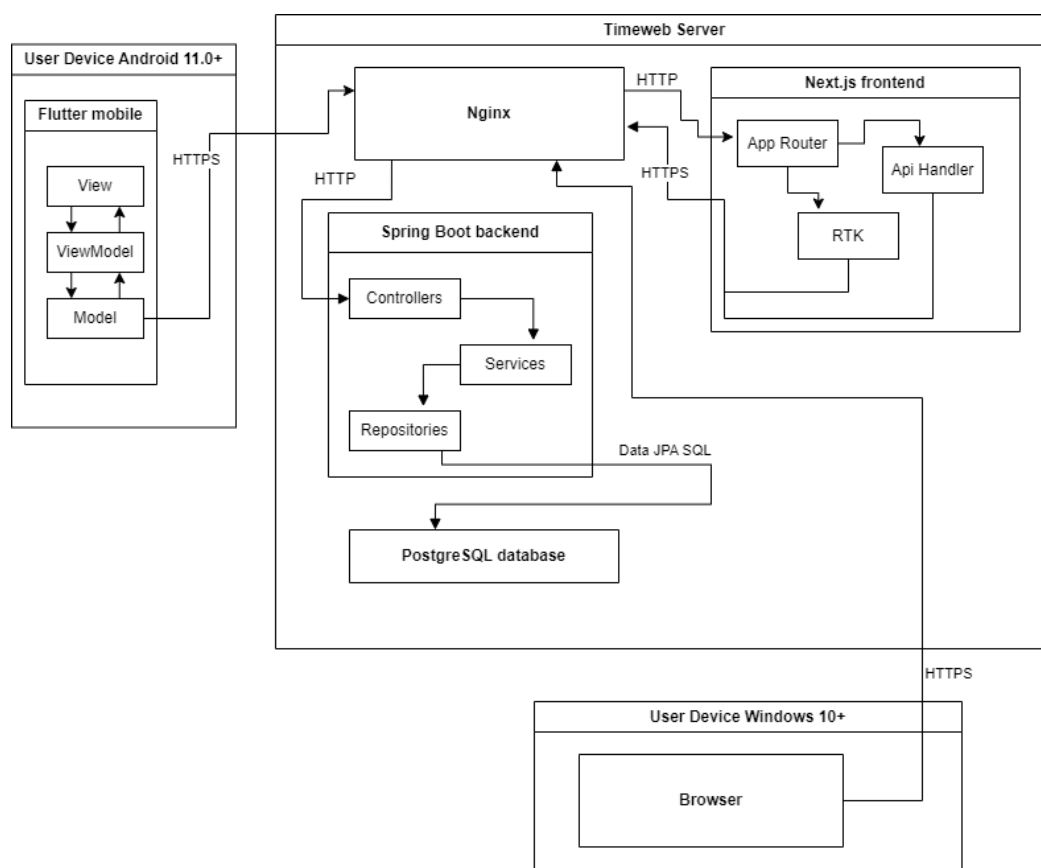


Рисунок 11 - Схематичное изображение структуры проекта

На рисунке 12 представлена диаграмма последовательности, которая является примером общения клиента с сервером при создании заметки учеником.

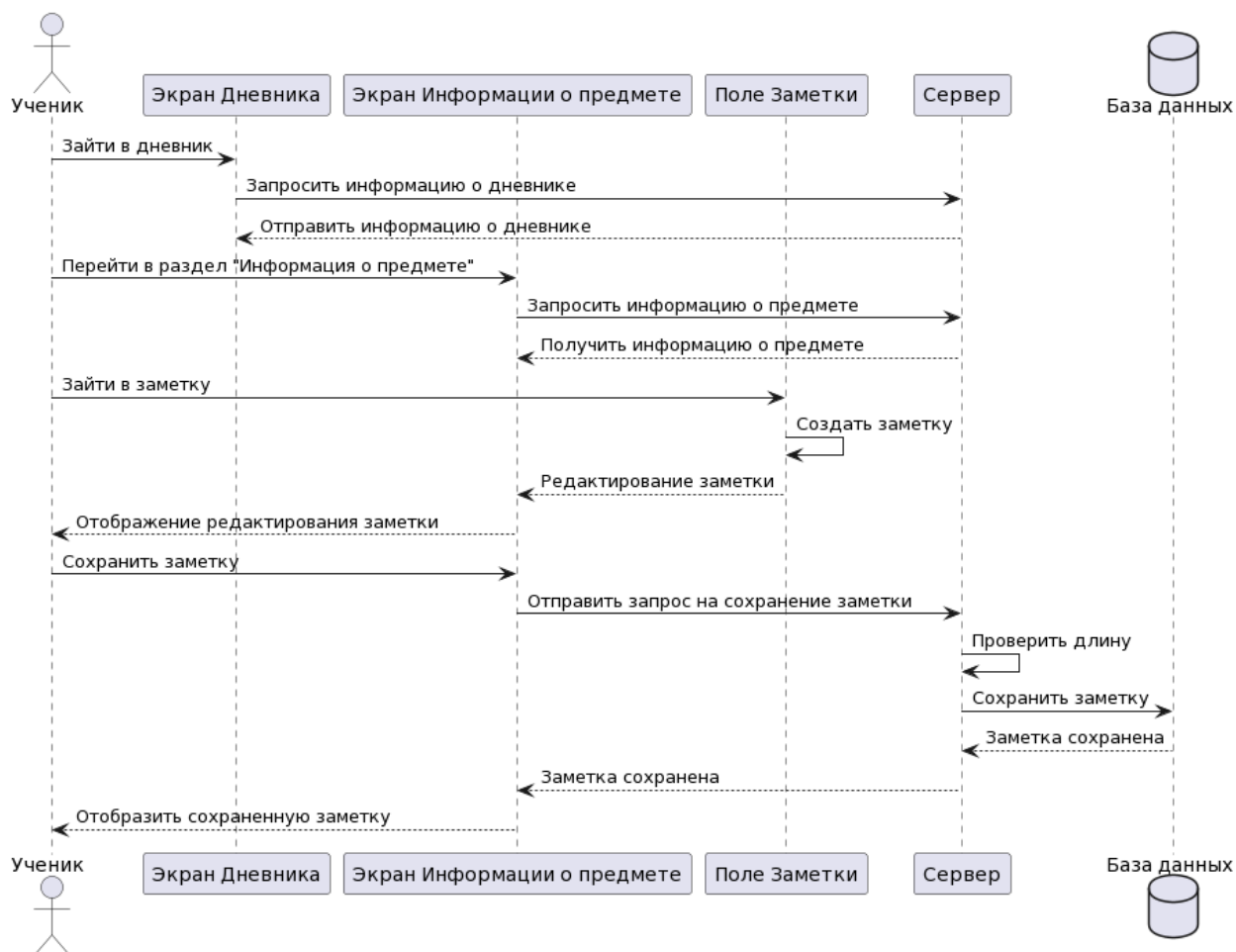


Рисунок 12 - Диаграмма последовательности создания заметки учеником

2.2 Средства реализации

При написании программного кода использовался паттерн MVC (от английского Model-View-Controller), то есть разделение логики работы представления (отображения), бизнес-логики и связующего звена. Данный паттерн позволяет делать реализацию каждого слоя независимо от других, предоставляя только определённый API [4]. В качестве view – представления данных – используются мобильное и веб-приложения, а model и controller реализованы на серверной стороне. Controller обрабатывает запросы от view (является посредником), делая запросы к model для работы с данными.

2.2.1 Серверная часть

В качестве языка программирования для серверной части была взята Java 21. Java – язык для объектно-ориентированного программирования со статической типизацией. С помощью объектно-ориентированного подхода легко в виде кода представить сущности платформы и связи между ними. Статическая типизация предупреждает от ошибок несовместимости типов на этапе работы приложения. Кроме того, код, написанный на Java, быстро работает и требует меньше оперативной памяти по сравнению с другими языками [5]. Также для Java есть актуальная документация и множество библиотек, облегчающих разработку.

Для автоматической генерации Swagger при написании кода использовалась библиотека springdoc-openapi. В коде классов над контроллерами, методами и необходимыми DTO указывались специальные аннотации с комментариями.

В качестве фреймворка был взят Spring Boot 3.2.3, который предоставляет множество необходимых для веб-разработки инструментов, а также сокращает количество кода за счёт внедрения зависимостей [6].

В качестве хранилища всей необходимой информации приложения использовалась база данных PostgreSQL.

2.2.2 Клиентская часть (веб-приложение)

Для написания веб-приложения за основу была взята архитектурная методология FSD, так как она помогает писать структурный и поддерживаемый код.

Сайт был написан на языках JavaScript ES6 и Typescript 5.0.4.

Язык JavaScript широко распространён и является фактически основным языком для разработки веб-приложений. Он поддерживается всеми

современными браузерами, что делает его идеальным выбором для клиентской части.

TypeScript добавляет строгую типизацию к JavaScript, что помогает обнаруживать ошибки на этапе компиляции, а не во время выполнения. Это значительно упрощает отладку и повышает надежность кода.

Node.js 18.18.2 была взята как среда выполнения JavaScript, позволяющая запускать JavaScript на сервере. Она обеспечивает высокую производительность и масштабируемость для серверной части frontend приложения.

В качестве фреймворка был взят Next.js 14.1.1. Он ускоряет загрузку страниц с контентом, а также имеет встроенную систему маршрутизации.

2.2.3 Клиентская часть (мобильное приложение)

Мобильное приложение было написано на языке Dart 3.3.0 с использованием фреймворка Flutter 3.19.2.

Dart также является строго типизированным языком, что, несомненно, упрощает разработку. Кроме того, Dart не сложен в освоении, а приложения на нём имеют высокую производительность [7].

Flutter, в свою очередь, тоже имеет высокую производительность. Он активно развивается и поэтому содержит множество готовых решений и библиотек.

2.3 Серверная часть

2.3.1 Механика работы сервера

Серверная часть приложения работает следующим образом.

Когда на контроллер приходит запрос, тот перенаправляет его в слой бизнес-логики – какой-то из сервисов. В сервисе происходит обработка данных. При необходимости сервис обращается в слой доступа к данным для

работы с базой данных – сохранения, обновления и удаления записей в ней. После завершения работы сервиса контроллер, который к нему обратился, отправляет ответ [6].

2.3.2 Слой контроллеров

Слой контроллеров реализован на основе архитектурного стиля REST. Каждый контроллер помечен специальной Spring Boot аннотацией «@RestController».

Каждый метод контроллера настроен на принятие запросов по определённому пути и поддерживает один из типов запросов: GET, POST, PUT или DELETE. Эти функции реализованы с помощью аннотаций, соответствующих типам запросов («@GetMapping» и другие).

В целях упрощения поддержки кода слой контроллеров выполнен в виде интерфейсов. Реализация их хранится в отдельном пакете.

Основные контроллеры – AdminPanelController и DiaryController. AdminPanelController отвечает за настройку школы: через него происходит работа с пользователями учебной организации, школьными данными. К этому контроллеру имеет доступ только пользователь с ролью «Локальный администратор» в данной учебной организации.

Через DiaryController происходит получение учащимися сведений об уроках (вместе с оценками и прочей важной информацией) и создание заметок.

2.3.3 Слой бизнес-логики

Слой бизнес-логики, или слой сервисов, содержит код, который решает поставленные бизнес-задачи. В сервисах при необходимости происходит конвертация объектов типа dto в объекты-сущности базы данных и наоборот. В методах сервисов происходит создание и обновление объектов, а также этот слой выполняет запросы в слой доступа к данным. Это могут быть запросы на получение, сохранение, обновление или удаление данных.

В приложении каждый сервис помечен специальной Spring Boot аннотацией «@Service» [6].

2.3.4 Слой доступа к данным

Слой доступа к данным является промежуточным звеном между бизнес-логикой и базой данных. Он представляет из себя удобную абстракцию для упрощения доступа к данным, а также отделения этих обращений от бизнес-логики.

В приложении доступ к данным реализован с помощью создания интерфейсов репозитория, которые наследуются от интерфейса «JpaRepository». Каждый репозиторий помечен специальной Spring Boot аннотацией «@Repository» [6].

Такой подход позволяет не писать отдельный код для обращения к базе данных (за исключением сложных запросов). Процесс генерации необходимого кода оптимизирует SpringBoot.

2.3.5 Модели и DTO

Записи, которые хранятся в таблицах баз данных, SpringBoot позволяет удобно формализовать: таблиц можно представить в виде сущностей. Для этого в отдельном пакете реализованы классы, помеченные аннотацией «@Entity».

Поля в этих классах представляют собой столбцы таблиц базы. С помощью аннотаций можно указать идентификатор сущности, названия столбцов и таблиц, а также связи между сущностями.

Таким образом, получение, сохранение и использование данных из базы автоматизировано.

На рисунке 13 представлена диаграмма классов, которая отображает основные сущности платформы.

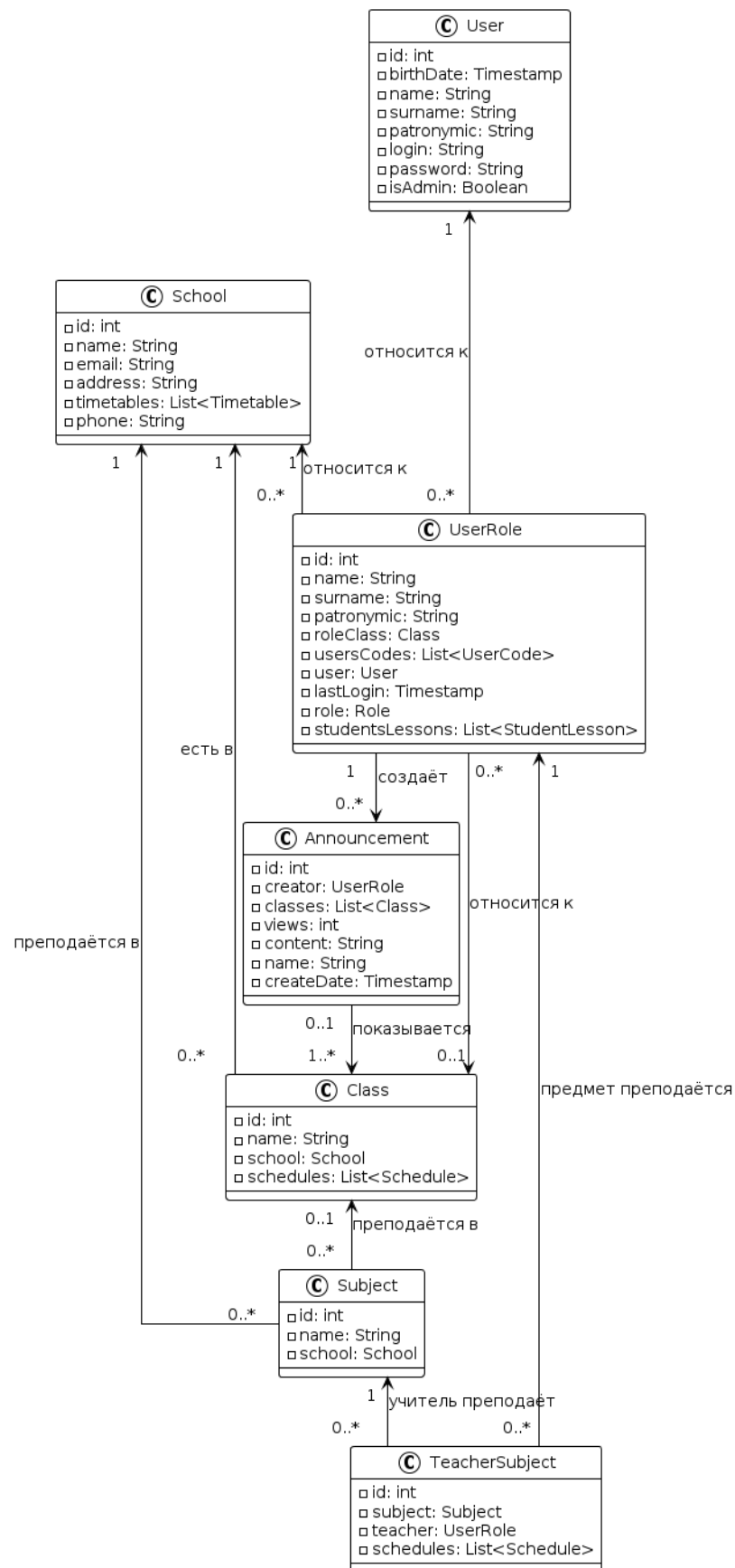


Рисунок 13 - Диаграмма основных классов

На рисунке 14 представлена диаграмма объектов, которая служит конкретным примером состояний объектов.

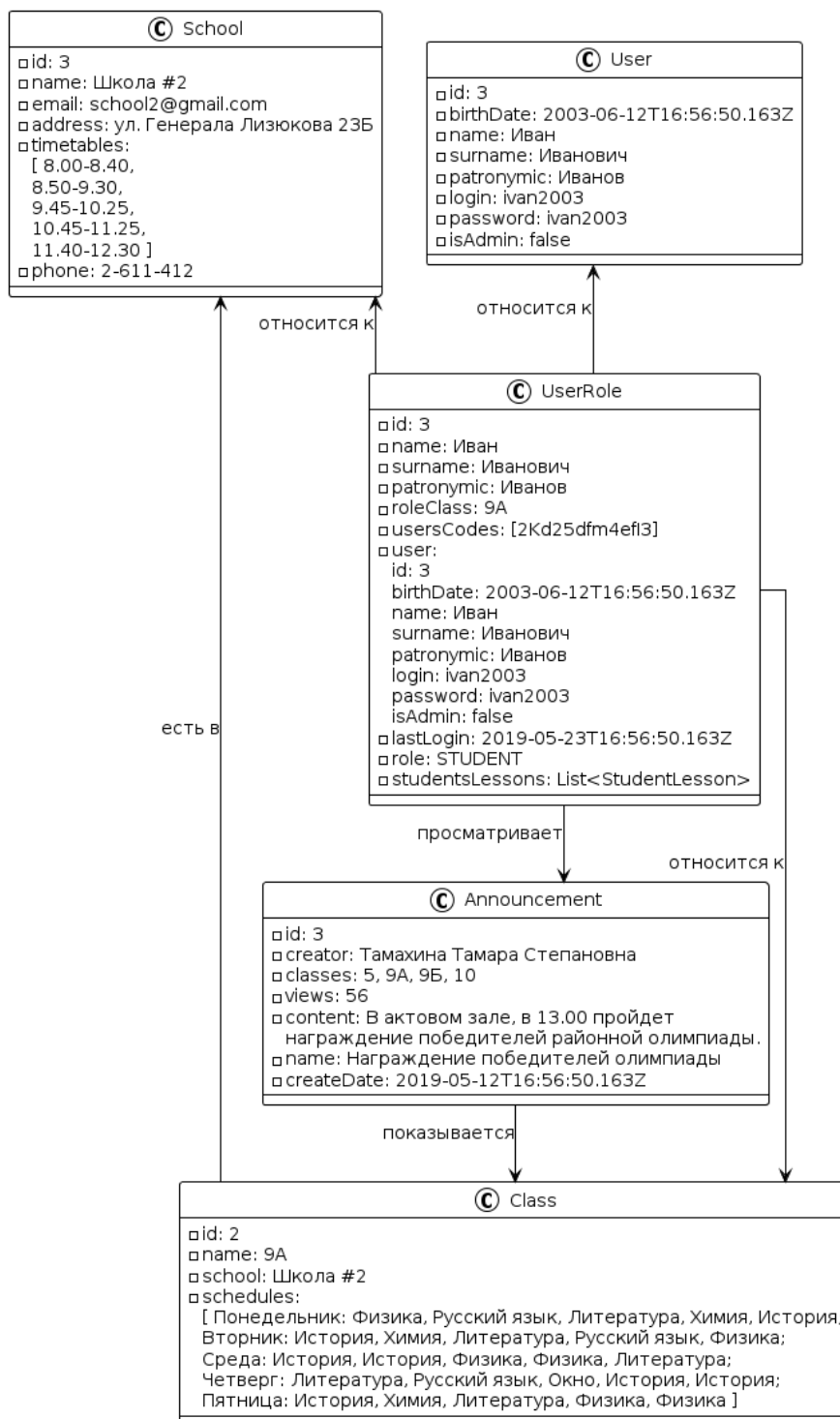


Рисунок 14 - Диаграмма объектов основных классов платформы

При общении сервера с клиентом происходит обмен DTO, настроенных под конкретные запросы клиента. Для перевода сущностей в DTO и наоборот в

приложении используются мапперы. Они реализованы в виде интерфейсов в отдельном пакете. Для их создания используется библиотека Mapstruct.

2.3.6 База данных

В базе данных хранится вся информация, с которой работают пользователи платформы.

Чтобы обеспечить полноценную работу платформы, необходима большая база данных с множеством таблиц. Таблицы и связи между ними можно подробно рассмотреть на ER-диаграмме, представленной на рисунке 15.

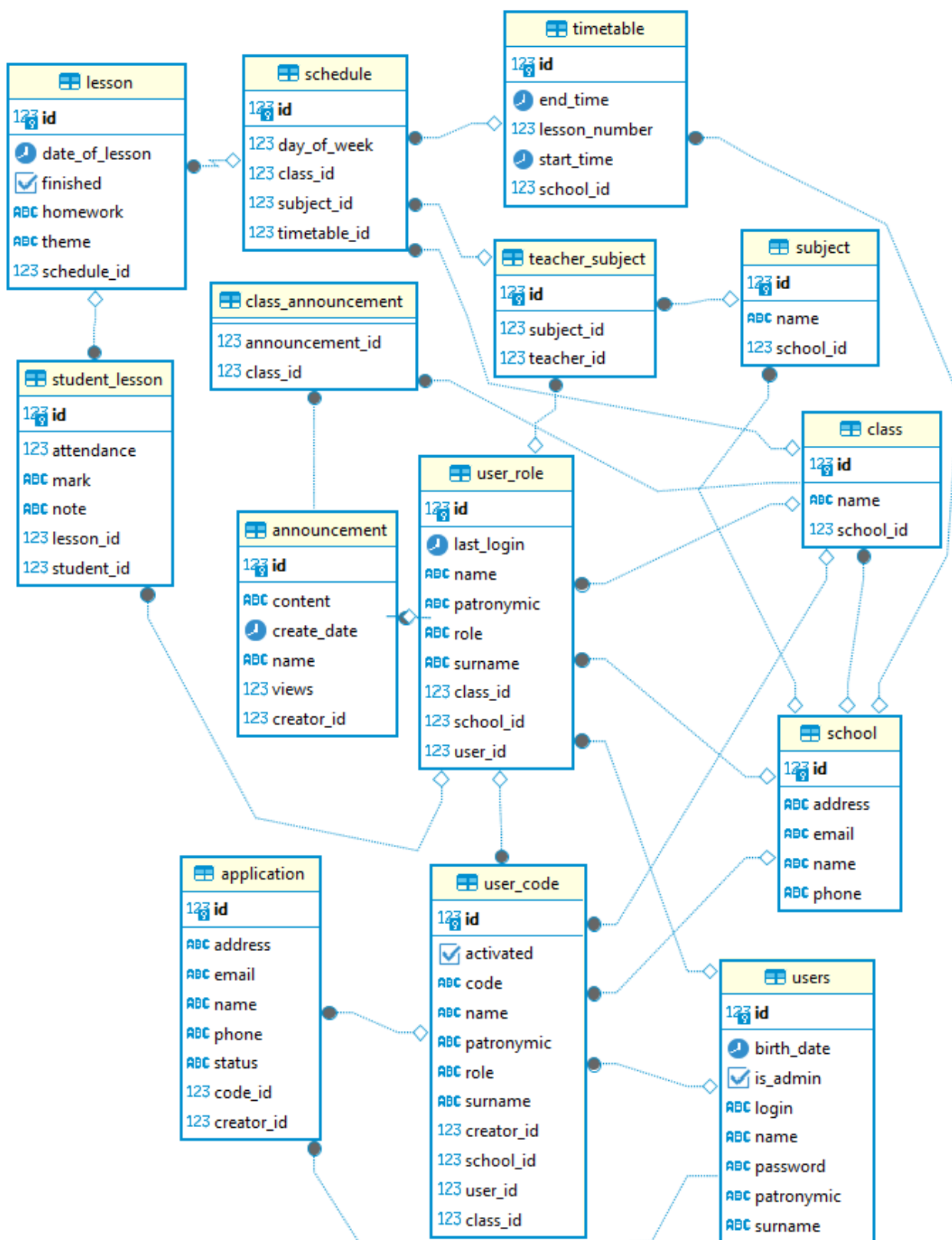


Рисунок 15 - ER-диаграмма базы данных платформы

Главные таблицы – user, user_role, school. В них соответственно хранятся данные о пользователях, пользовательских ролях в конкретных школах и самих школах.

2.4 Клиентская часть (мобильное приложение)

Для обзора мобильного приложения рассмотрим основной пользовательский сценарий в нём. В качестве пользователя будет выступать ученик «9А» класса «Иванов Иван Иванович» 10.10.2003 года рождения с логином ivan2003.

При первом входе в приложение пользователю открывается экран входа и регистрации (см. Рисунок 16). Зарегистрироваться в приложении может каждый желающий.

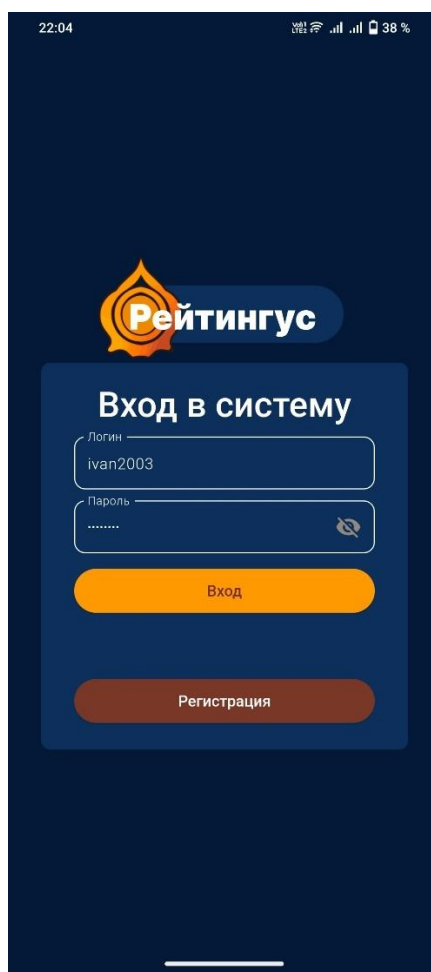


Рисунок 16 - Снимок экрана входа в приложение

После входа каждый пользователь попадает в разделе «Профиль». Там содержится информация о нём, а также кнопки для ввода кода приглашения и настроек и список учебных организаций, в которых состоит ученик (см. Рисунок 17).



Рисунок 17 - Снимок экрана профиля пользователя с ролью «Ученик»

Ученик состоит в двух учебных заведениях: «Школа #2» и «Музыкальная школа имени Бетховена» (пользователям, у которых не подключена ни одна учебная организация, доступен только раздел «Профиль»). Чтобы подключить ещё одно учебное заведение, ему нужно нажать на кнопку в левой верхней части экрана. Откроется экран для ввода кода приглашения (см. Рисунок 18).



Рисунок 18 - Снимок экрана для ввода кода приглашения

После ввода кода, который может сгенерировать администратор организации, в списке учебных организаций у ученика появится ещё одна.

После выбора одной из школ пользователь может перейти в раздел с объявлениями, чтобы просмотреть объявления школы или класса. Ученик может только смотреть их.

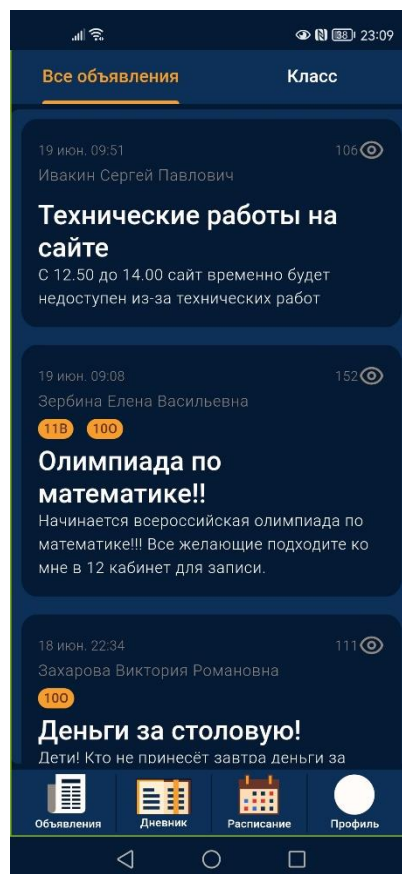


Рисунок 19 - Снимок экрана просмотра объявлений

Следующий раздел, который может просмотреть пользователь – «Дневник» (см. Рисунок 20). Тут по дням расписаны предметы, ФИО преподавателей, время и номер урока, а также, что самое важное – домашнее задание, оценки и посещаемость.

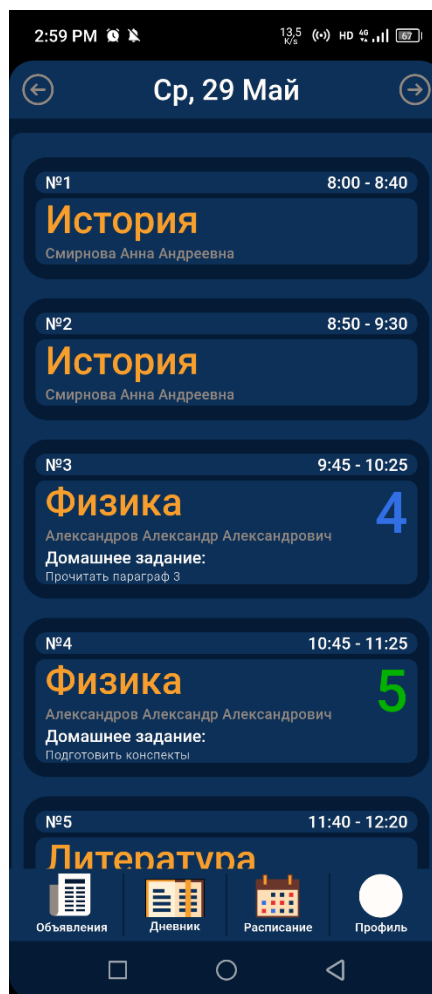


Рисунок 20 - Снимок экрана раздела «Дневник»

Можно переключаться между неделями с помощью кнопок. При нажатии на название дня недели можно просмотреть отдельно информацию по дню, а при нажатии на предмет можно просмотреть подробную информацию о предмете (см. Рисунок 21).

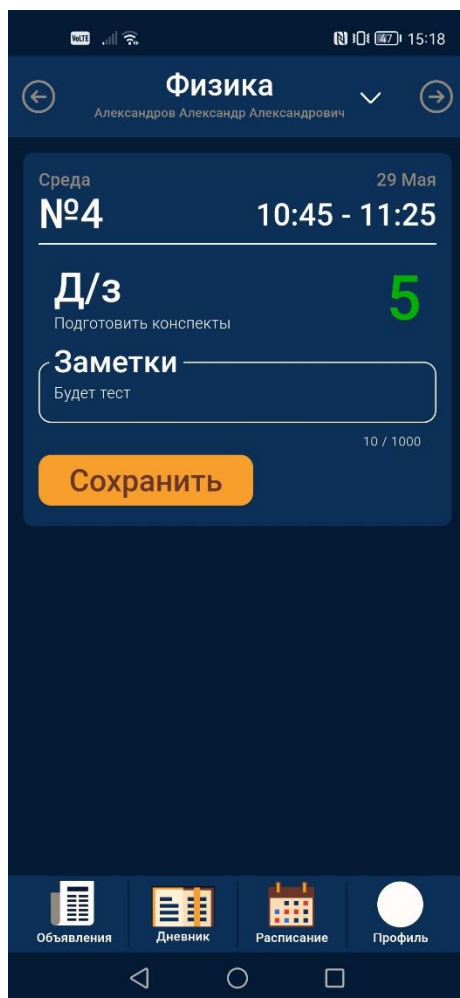


Рисунок 21 - Снимок экрана с информацией о предмете

Также на этом экране можно записать заметку к предмету. Это может быть какое-то поручение учителя, дополнительное домашнее задание или другая важная информация.

Процесс создания заметки учеником в виде диаграммы состояний представлен на рисунке 22.

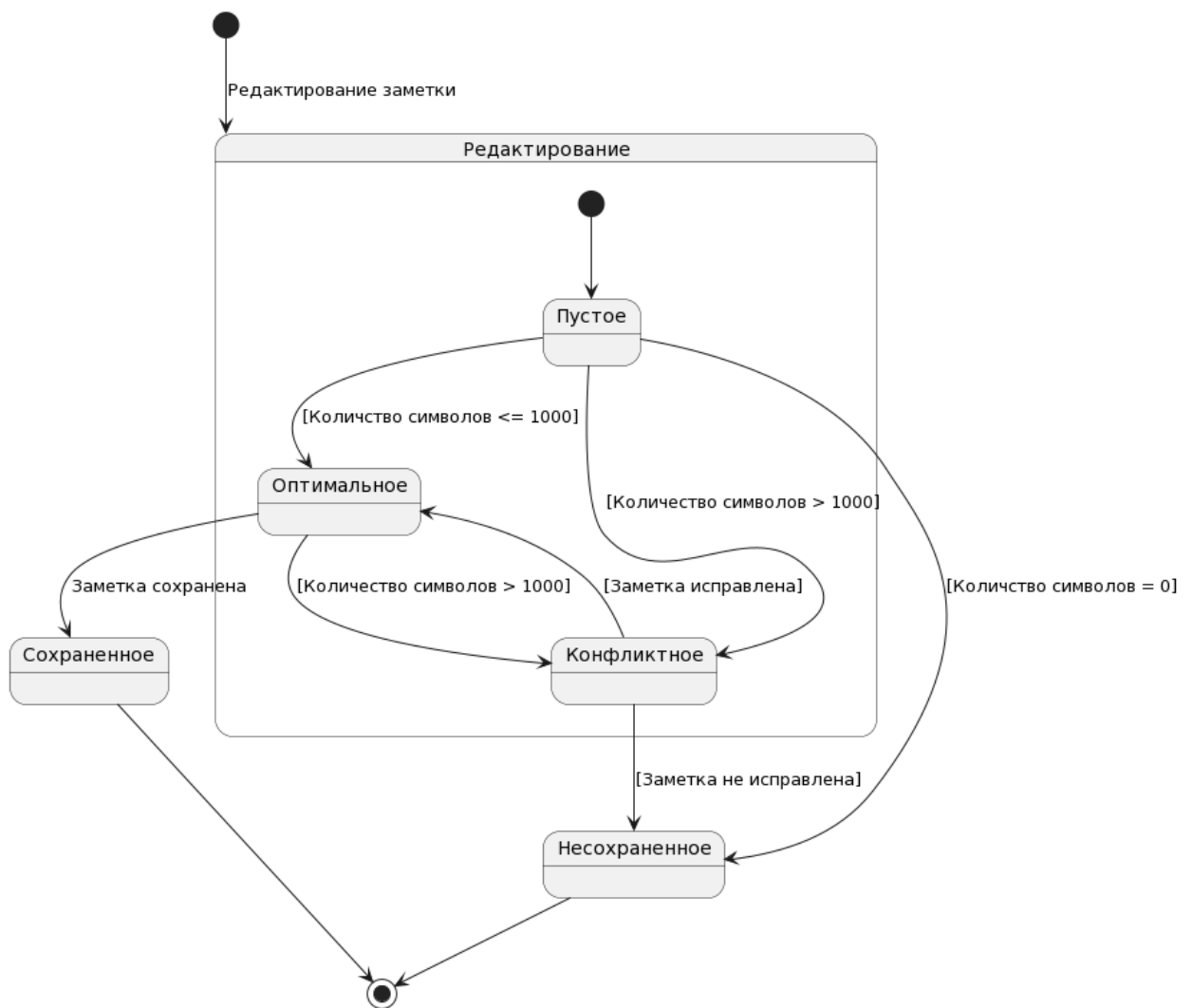


Рисунок 22 - Диаграмма состояний заметки при её создании

В разделе «Расписание» по умолчанию ученик может просмотреть расписание для своего класса. Также он может выбрать любой другой класс, который есть в школе, чтобы просмотреть и их расписания (см. Рисунок 23).

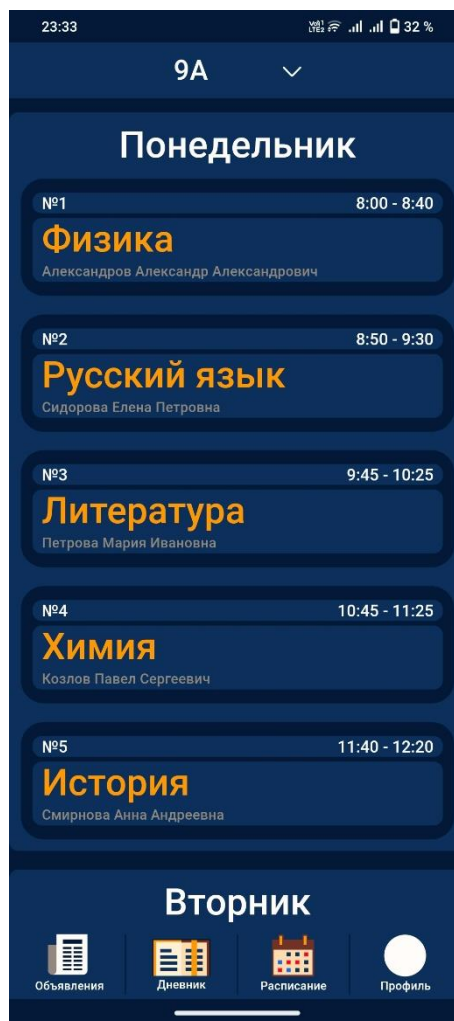


Рисунок 23 - Снимок экрана раздела «Расписание»

2.5 Клиентская часть (веб-приложение)

2.5.1 Неавторизованный пользователь

Каждый пользователь веб-приложения может просмотреть приветственный лендинг на главной странице сайта, призывающий отказаться от бумажных дневников и журналов. Справа сверху есть кнопка «Войти», после нажатия на которую можно войти или зарегистрироваться (см. Рисунок 24).

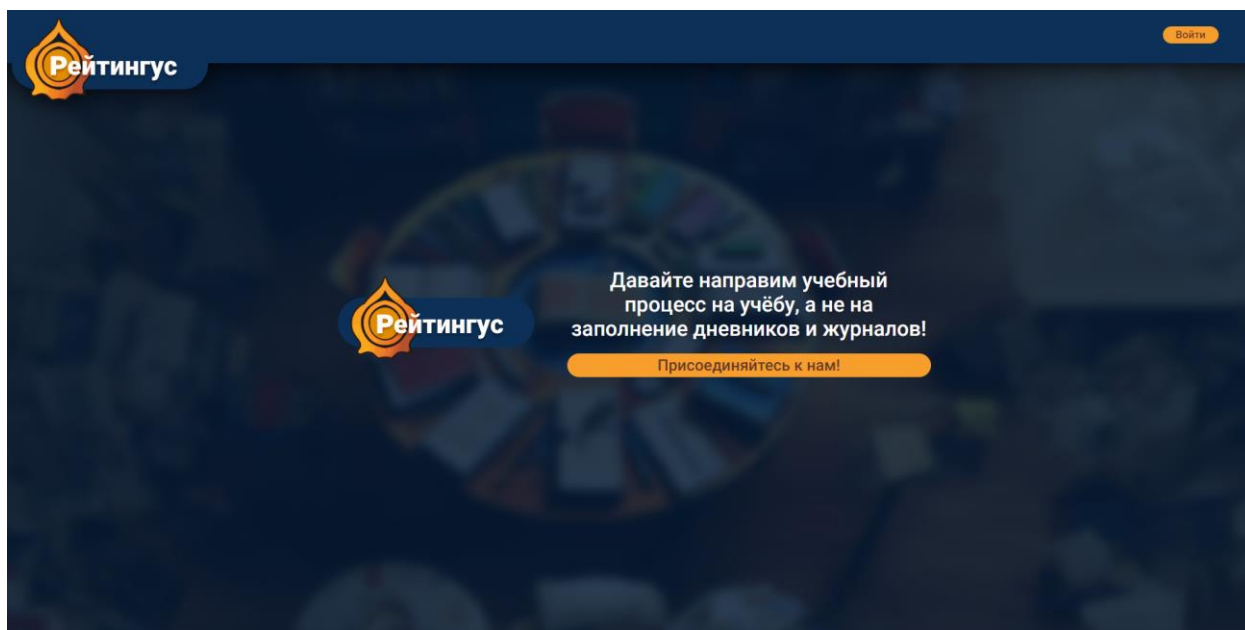


Рисунок 24 - Снимок страницы с лендингом

2.5.2 Авторизованный пользователь

Пользователь, который авторизовался на сайте, попадает на страницу профиля, где содержатся кнопки и информация, аналогичные мобильному представлению (см. Рисунок 25).

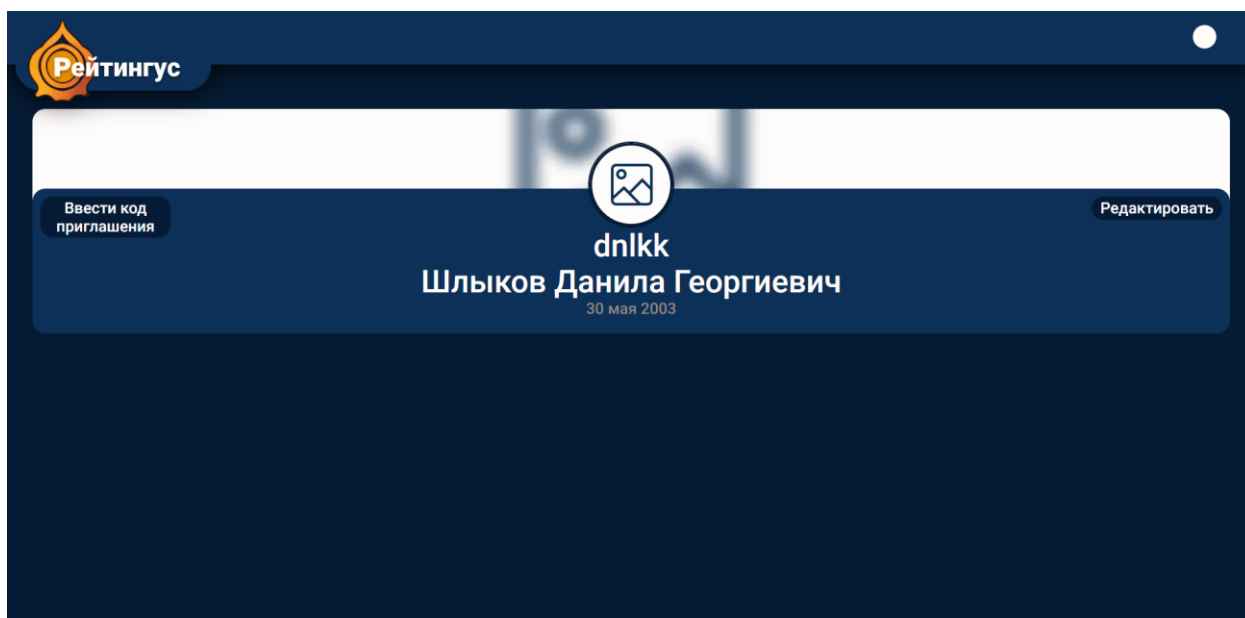


Рисунок 25 - Снимок страницы авторизованного пользователя

После ввода кода приглашения в профиле также высвечиваются доступные учебные организации.

На странице лендинга, пролистав вниз, авторизованный пользователь может отправить заявку на создание учебной организации.

2.5.3 Ученик

Ученик в веб-приложении обладает теми же инструментами, что и в мобильной версии. Примеры веб-интерфейса представлены на рисунках 26 и 27.



Рисунок 26 - Снимок экрана Дневника

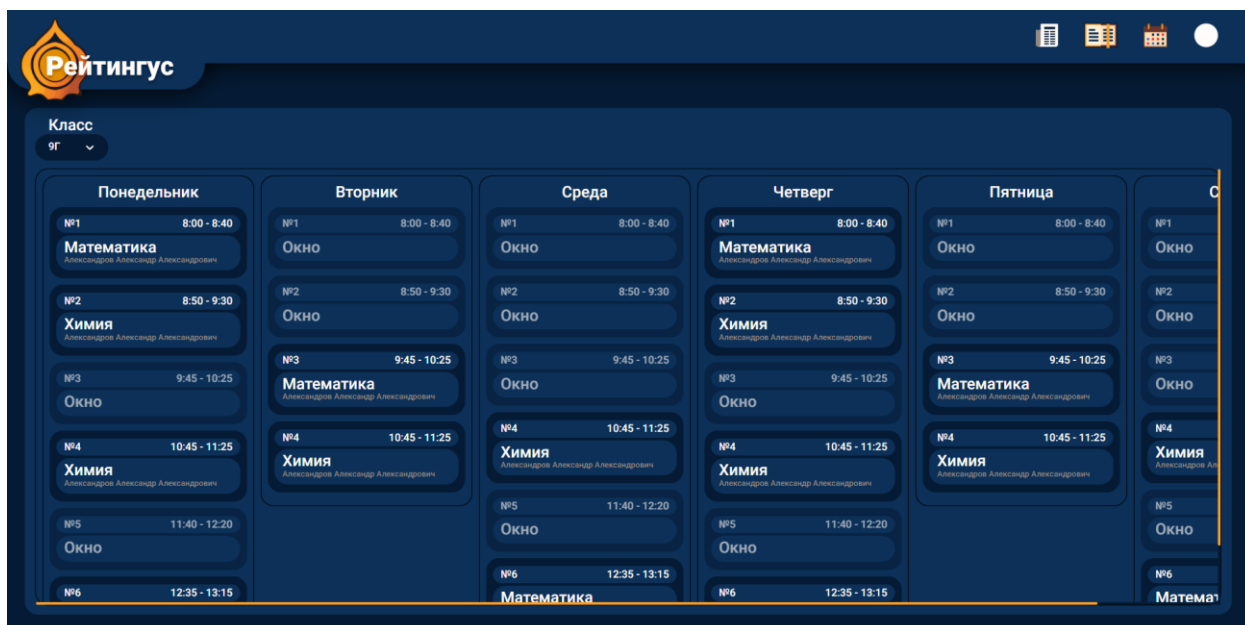


Рисунок 27 - Снимок экрана Расписания

2.5.4 Учитель

У пользователя с ролью «Учитель» есть несколько доступных разделов.

Во-первых, это раздел «Объявления». В отличие от учеников учитель может создавать объявления, выбрав для каких классов оно будет доступно и введя название и текст объявления (см. Рисунок 28). Кроме того, учитель может удалить объявления, созданные им.

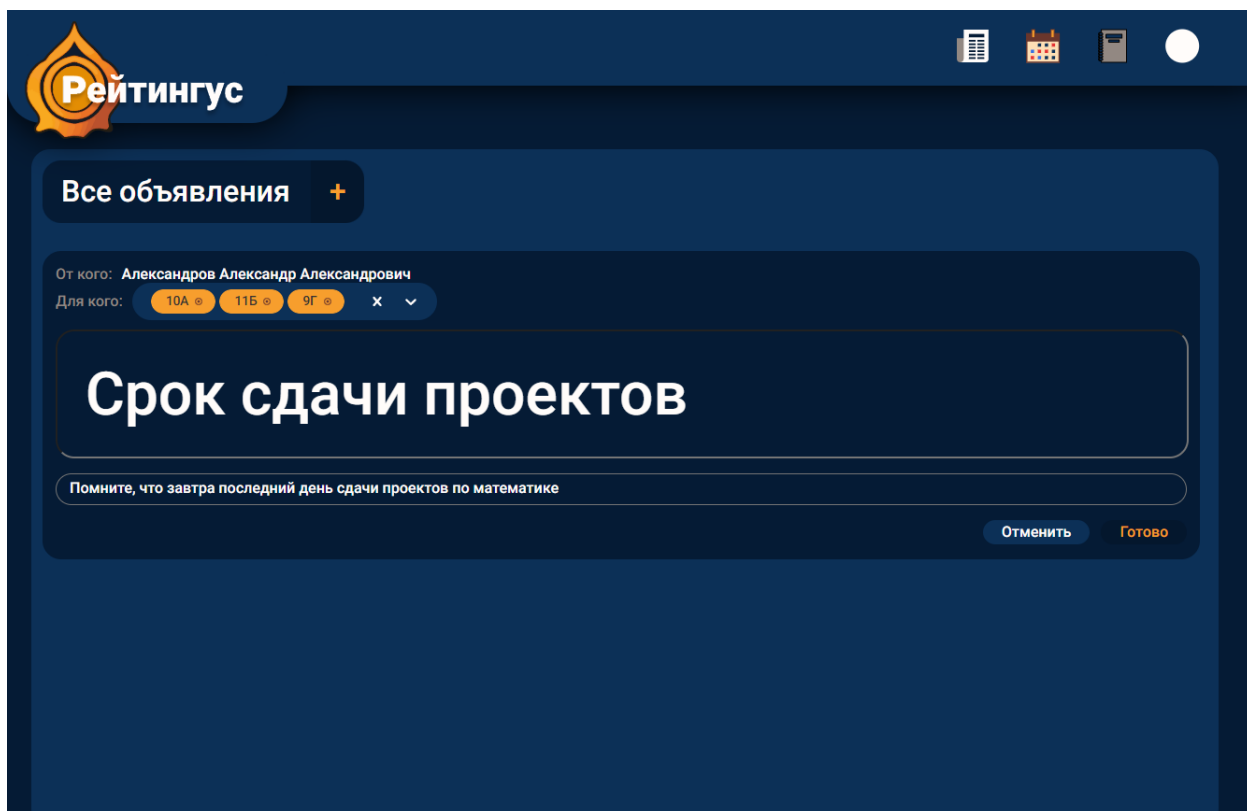


Рисунок 28 - Снимок страницы создания объявления

Также учитель может просмотреть раздел «Расписание», по функционалу схожий с мобильной версией.

Самый важный для учителя раздел – «Журнал». В подразделе «Ученики» учитель может просмотреть все оценки и посещаемость учеников по каждому из классов, выбрав класс и предмет в выпадающих списках (см. Рисунок 29).

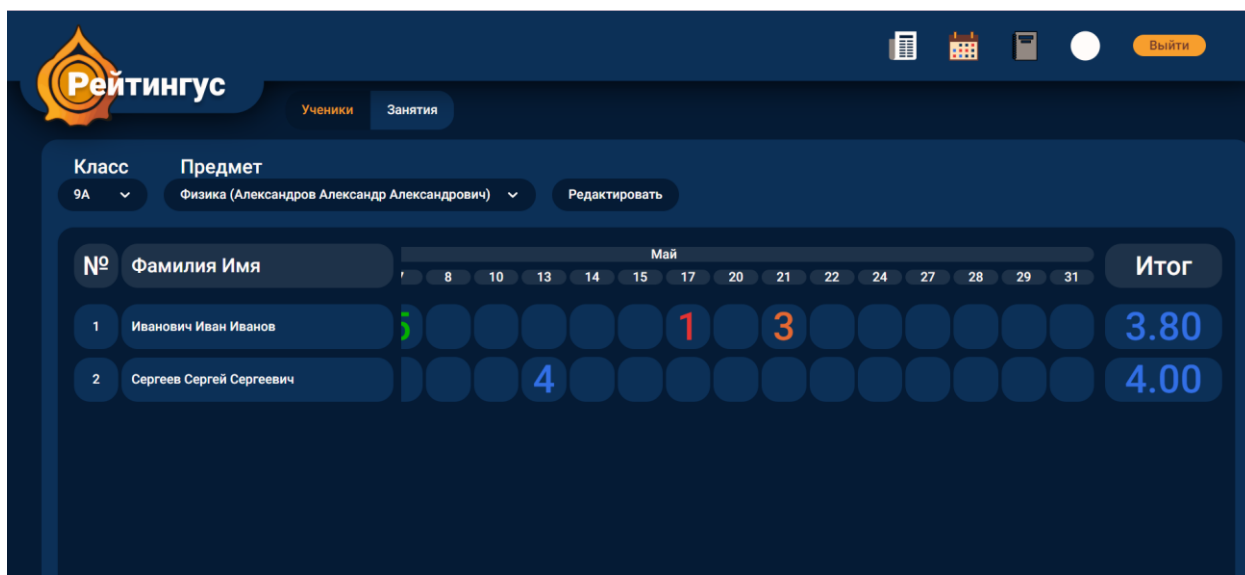


Рисунок 29 - Снимок страницы Журнала со списком учеников

В подразделе «Занятия», аналогичным образом выбрав класс и предмет, учитель может просмотреть темы занятий и выданные домашние задания.

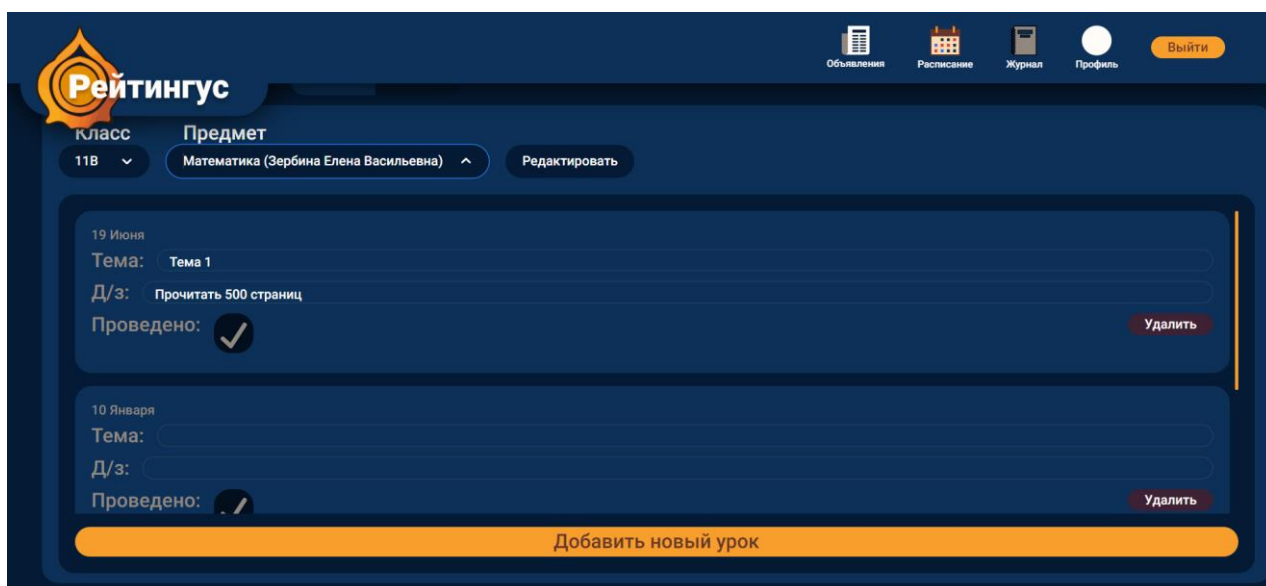


Рисунок 30 - Снимок страницы Журнала со списком занятий

Если учитель является учителем выбранного класса по выбранному предмету, то ему будет доступна кнопка «Редактировать». После нажатия на неё учитель может ставить оценки, отмечать посещаемость, записывать тему и домашнее задание для класса.

2.5.5 Локальный администратор

Пользователь с ролью «Локальный администратор» имеет те же функции, что и учитель. Он может управлять журналом любого класса по любому предмету, создавать и удалять любые объявления.

Важнейшим разделом для администратора является «Админ-панель». Здесь он может:

- управлять пользователями учебной организации;
- создавать коды приглашения;
- редактировать информацию о школе;
- создавать расписание;
- связывать между собой предметы, учителей и классы.

На рисунках 31 и 32 представлены примеры страниц для локального администратора.

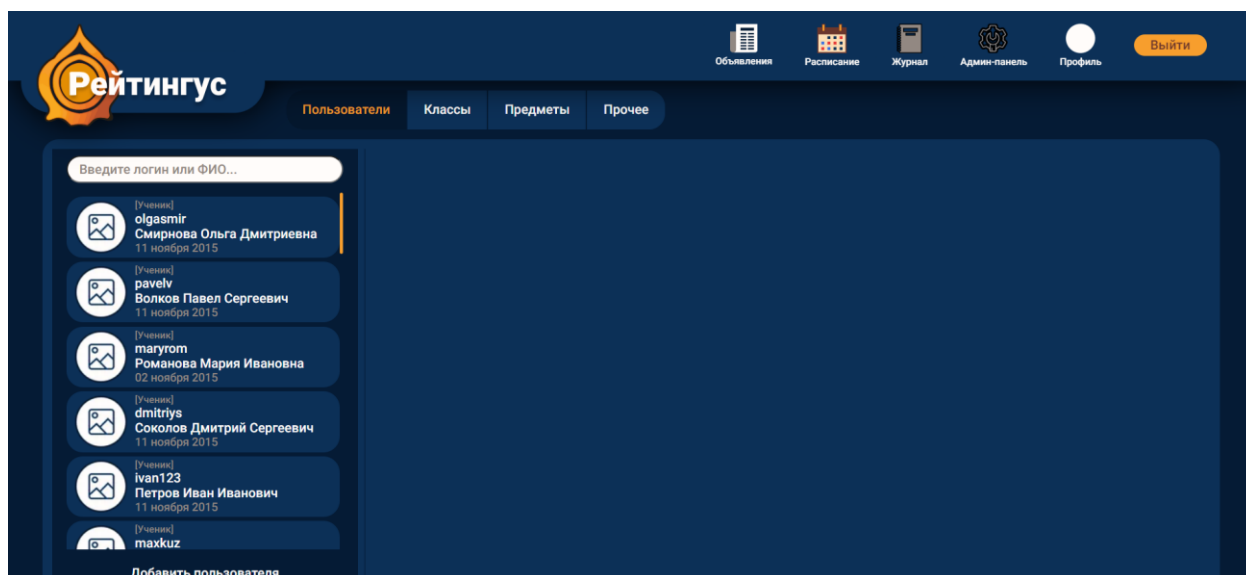


Рисунок 31 - Снимок страницы просмотра информации о пользователях внутри учебной организации

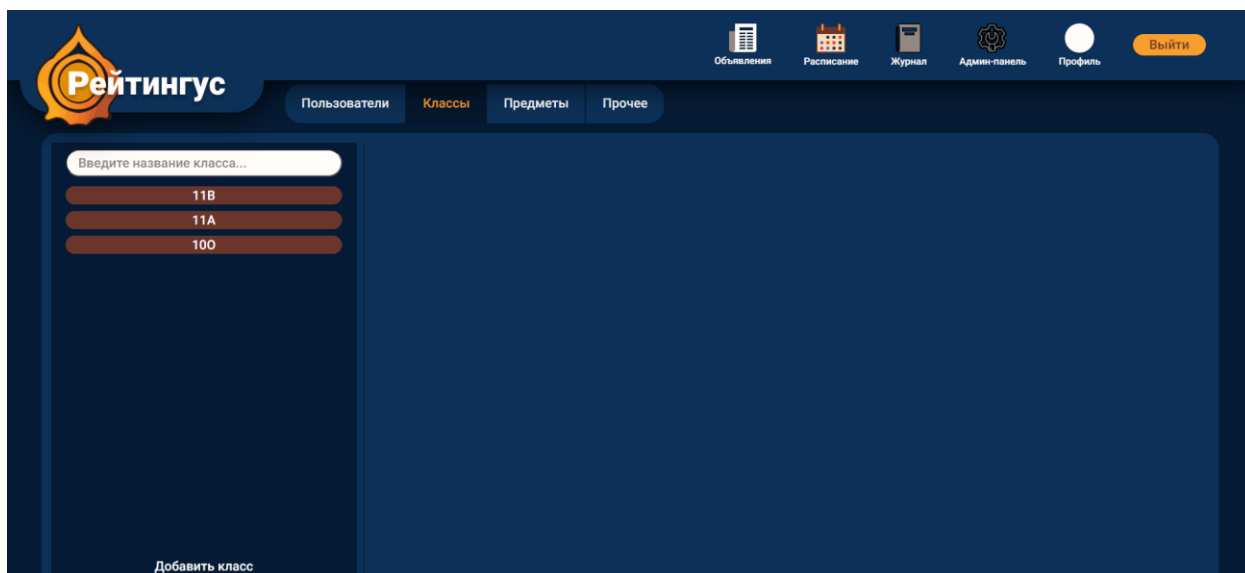


Рисунок 32 - Снимок страницы для работы с классами

2.5.6 Менеджер платформы

Менеджер платформы необходим для глобального управления платформой. Основная их задача – рассматривать заявки на создание учебных организаций и одобрять или отклонять их, а также создавать (и обновлять) код приглашения для локального администратора организации, заявка на создание которого была одобрена (см. Рисунок 33).

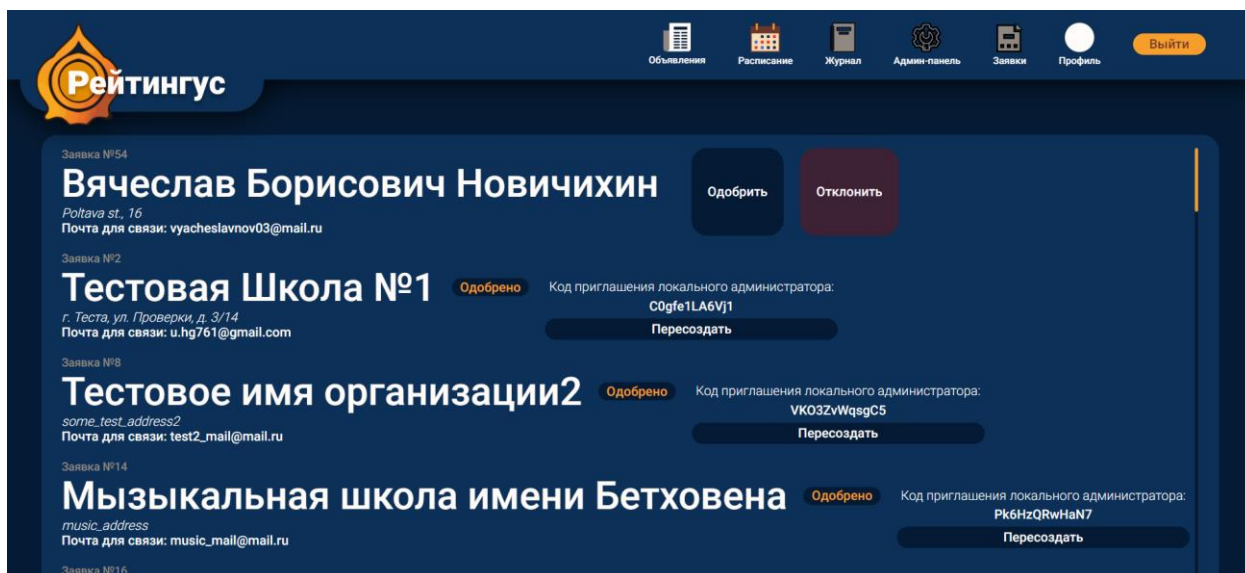


Рисунок 33 - Снимок страницы с заявками на создание учебной организации

2.6 Развертывание

Приложение развернуто на сервере Timeweb. Он был выбран по следующим причинам:

- является отечественной платформой;
- оптимален в отношении цены к качеству;
- лёгок в использовании;
- имеет удобный интерфейс.

Был реализован CI/CD средствами Github Actions и Ansible для программного кода вэб- и серверного приложений. GitHub при отправке кода в главную ветку собирает контейнер, помещает его в GitHub Container registry и вызывает скрипт Ansible, который выполняет перезагрузку контейнеров на сервере.

Backend-часть приложения также общается с базой данных, которая поднята в отдельном контейнере. Они связаны через JDBC.

Веб-сервер на Node.js для веб Frontend части и Backend между собой взаимодействуют по http-протоколу.

Все запросы на сервер проходят через nginx, который выступает как балансировщик нагрузки и маршрутизатор запросов. Nginx также работает в отдельном контейнере.

Схема развёртывания приложения изображена на диаграмме (см. Рисунок 34).

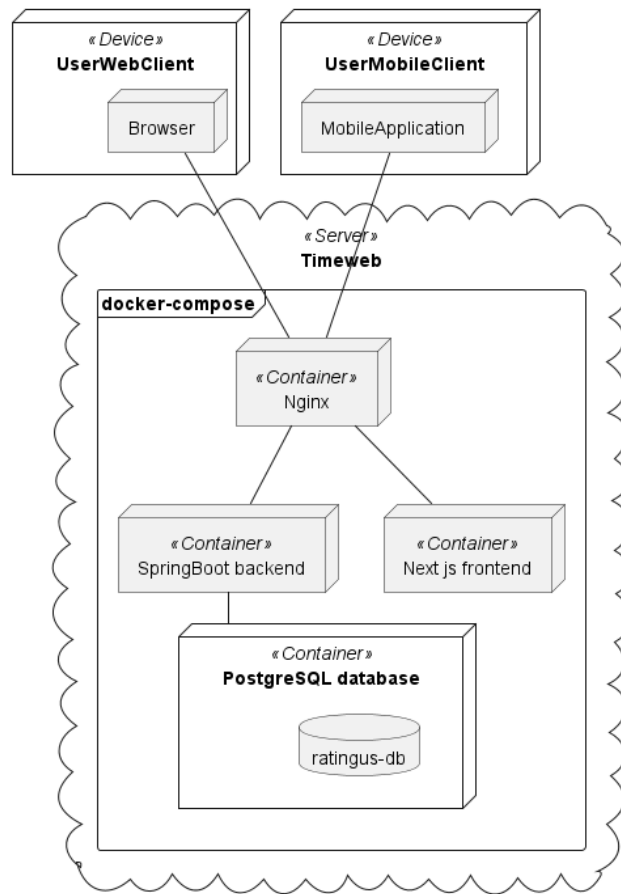


Рисунок 34 - Диаграмма развёртывания приложения

3 Анализ проделанной работы

3.1 Тестирование

По завершении главного этапа разработки было произведено нагрузочное тестирование. Для этого были привлечены третьи лица, которым была поставлена задача:

- установить приложение;
- зарегистрироваться;
- получить код приглашения (вне платформы) и ввести его;
- используя все доступные кнопки, изучить работу приложения;
- отмечать найденные ошибки в работе приложения.

Предварительно были внесены тестовые данные – объявления, расписание и дневник.

Всего было зарегистрировано 13 уникальных посетителей, часть из которых дала обратную связь, прикрепив необходимые снимки экрана.

Кроме того, тестирование готовой платформы проводилось членами команды согласно составленным на основе технического задания чек-листам (см. Рисунок 35).

СЦЕНАРИЙ	УДОБСТВО (0-5)	НАЙДЕННЫЕ БАГИ
Регистрация	5	
Вход	5	-
Ввести код для подключения к школе	4	нет какой либо надписи подтверждения о том что код введен
Посмотреть свое расписание в дневнике	5	
Посмотреть оценки в дневнике	5	-
Посмотреть посещаемость	5	-
Посмотреть домашнее задание	4	Шрифт побольше, очень мелко
Создать заметку в дневнике	5	Заметка не сохраняется
Редактировать заметку в дневнике	5	Заметка не сохраняется
Посмотреть информацию о предмете	5	(какая именно информация о предмете должна быть?)
Посмотреть опубликованное объявление	5	-
Посмотреть расписание в разделе "Расписание"	5	-

Рисунок 35 - Один из созданных на основе ТЗ чек-листов для тестирования

Также для более объективной оценки качества работы платформы был произведён обмен проектами с однокурсниками, которые проверяли сервис на наличие ошибок и проблем. Всего было произведено 5 таких обменов. Каждый обмен сопровождался обратной связью, на основе которой производилось исправление найденных ошибок.

3.2 Сбор метрик

Для анализа действий пользователей и сбора статистики к платформе были подключены AppMetrica и Яндекс Метрика. Сбор статистики ведётся по количеству посещений различных разделов приложения, а также по работе с кодом приглашения.

В таблице 1 представлен набор метрик и действий, по которым собирается статистика (сокращения: «М» – мобильное приложение, «В» – веб-приложение).

Таблица 1 - Таблица отслеживаемых метрик

Метрика/действие	Цель метрики	М	В
Количество созданных заметок	Оценить полезность и удобность функции	Да	Да
Время, проведённое на сайте	Понять, быстро ли решаются необходимые задачи	Да	Да
Посещаемость расписания	Оценить полезность и удобность функции	Да	Да
Посещаемость объявлений	Оценить полезность и удобность функции	Да	Да
Посещаемость объявлений для класса	Оценить полезность и удобность функции	Да	Да

Посещаемость дневника	Оценить полезность и удобность функции	Да	Да
Создание объявления	Оценить полезность и удобность функции	Нет	Да
Поставить оценку в журнале	Понять, как часто ставят оценки; Оценить полезность и удобность функции	Нет	Да
Выдать домашнее задание в журнале	Понять, как часто выдают домашнее задание; Оценить полезность и удобность функции	Нет	Да
Создать код приглашения	Понять, как часто создают коды приглашения	Нет	Да
Отправить заявку на создание учебной организации	Понять, сколько всего создаётся заявок	Нет	Да
Одобрить заявку на создание учебной организации	Понять, сколько заявок одобряется	Нет	Да

В общем случае сбор статистики настроен таким образом, чтобы:

- понять важность реализованных разделов и оценить частоту их использования;
- оценить удобство использования платформы;
- оценить конверсию платформы (а именно – создание и одобрение заявок на создание учебных организаций и работу с кодом приглашения).

Последний из перечисленных пунктов имеет наибольшую важность, поэтому на страницах просмотра отчётов были созданы диаграммы и таблицы, которые отображают данные в удобном виде (см. рисунки 36, 37, 38, 39).

Заявки на создание школы		⋮
Метрики		Итого
Визиты		121
Достижения цели: «Отправить заявку на создание учебной организации»		18
Достижения цели: «Одобрить заявку на создание учебной организации»		8

Рисунок 36 - Данные о работе с заявками на создание школы

Подключение учебных организаций		⋮
Метрики		Итого
Визиты		122
Достижения цели: «Зайти в профиль»		8
Достижения цели: «Ввести код организации»		2
Достижения цели: «Отправлен запрос на подключение пользователя к организации»		2
Достижения цели: «Пользователь добавлен в организацию»		2

Рисунок 37 - Данные о подключении учебных организаций

Использование кодов приглашения		⋮
Метрики		Итого
Достижения цели: «Сгенерировать код приглашения»		93
Достижения цели: «Пользователь добавлен в организацию»		86

Рисунок 38 - Данные об использовании кодов приглашения

Конверсия шагов

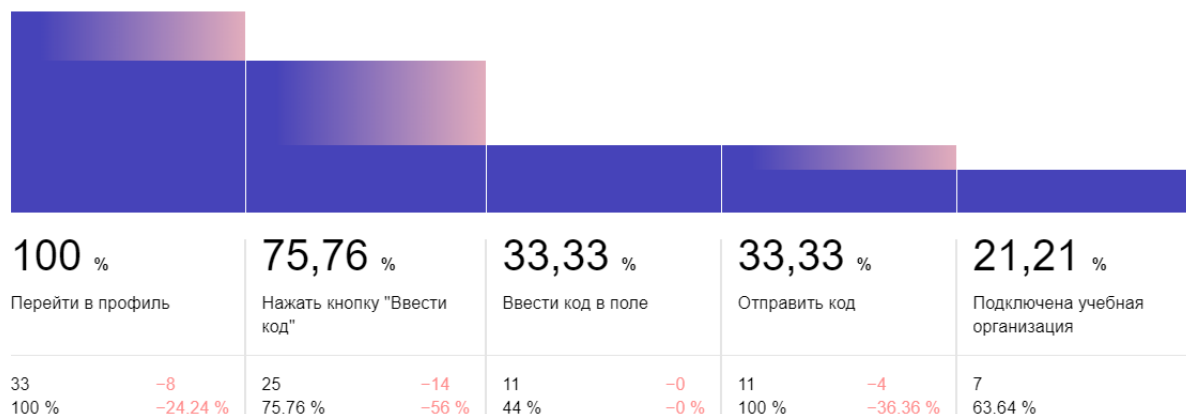


Рисунок 39 - Конверсия использования мобильного приложения

На рисунке 40 представлен снимок воронки, конечная цель которой – подключение пользователем учебной организации.



Рисунок 40 - Снимок воронки подключения учебных организаций пользователями

Настроенные метрики помогут выявить проблемные сценарии, которые будут устранены в последующих релизах. Например, по рисунку 40 видно, что треть пользователей не отправила введенный код приглашения. Если на большей выборке ситуация не изменится или ухудшится, то для упрощения

пути пользователя можно будет реализовать подключение учебных организаций через считывание QR-кода.

На рисунке 41 представлен снимок показателей App метрики, демонстрирующих процент пользователей, посетивших различные разделы приложения (рассматриваются только авторизованные пользователи).



Рисунок 41 - Посещаемость различных разделов мобильного приложения

ЗАКЛЮЧЕНИЕ

Таким образом, платформа для дистанционного управления учебным процессом была создана, а, значит, поставленная цель достигнута.

Для реализации проекта был проведён опрос учащихся, затем выполнен тщательный анализ предстоящих работ и продумана архитектура платформы. Затем был написан программный код для сервера, сайта и мобильного приложений, а также выполнено развёртывание платформы. Далее было проведено тестирование завершённой платформы и анализ выполненной работы.

Задачи, выделенные для достижения цели, были реализованы, и учащиеся с помощью платформы действительно могут получить сведения о расписании, оценках, объявлениях, а также полученных оценках и отмеченной посещаемости.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Балльно-рейтинговая система // Воронежский государственный университет Факультет компьютерных наук URL: <https://www.cs.vsu.ru/brs/login?next=%2Fbrs%2F> (дата обращения: 26.04.2024).
2. <https://ru.wikipedia.org/wiki/%D0%94%D0%BD%D0%B5%D0%B2%D0%BD%D0%B8%D0%BA.%D1%80%D1%83> Дневник.ру // Дневник URL: <https://dnevnik.ru/> (дата обращения: 26.04.2024).
3. Расписание занятий - Weeklie // Google Play URL: <https://play.google.com/store/apps/details?id=com.numen.timetable> (дата обращения: 26.04.2024).
4. Model-View-Controller // Википедия URL: <https://ru.wikipedia.org/wiki/Model-View-Controller> (дата обращения: 29.05.2024).
5. Rui Pereira., Marco Couto, Francisco Ribeiro, Rui Rua, Jácome Cunha, João Paulo Fernandes, João Saraiva Energy «How Does Energy, Time, and Memory Relate?» // «Efficiency across Programming Languages», Vancouver, Canada. 2017. С. 8. URL: <https://greenlab.di.uminho.pt/wp-content/uploads/2017/09/paperSLE.pdf> (дата обращения: 26.05.2024).
6. Documentation Overview // Spring URL: <https://docs.spring.io/spring-boot/documentation.html> (дата обращения: 29.05.2024).
7. Flutter – единственная правильная кроссплатформа для приложений // Surf URL: <https://surf.ru/flutter-iedinstviennaia-pravilnaia-krossplatforma/> (дата обращения: 26.05.2024).

ПРИЛОЖЕНИЕ А

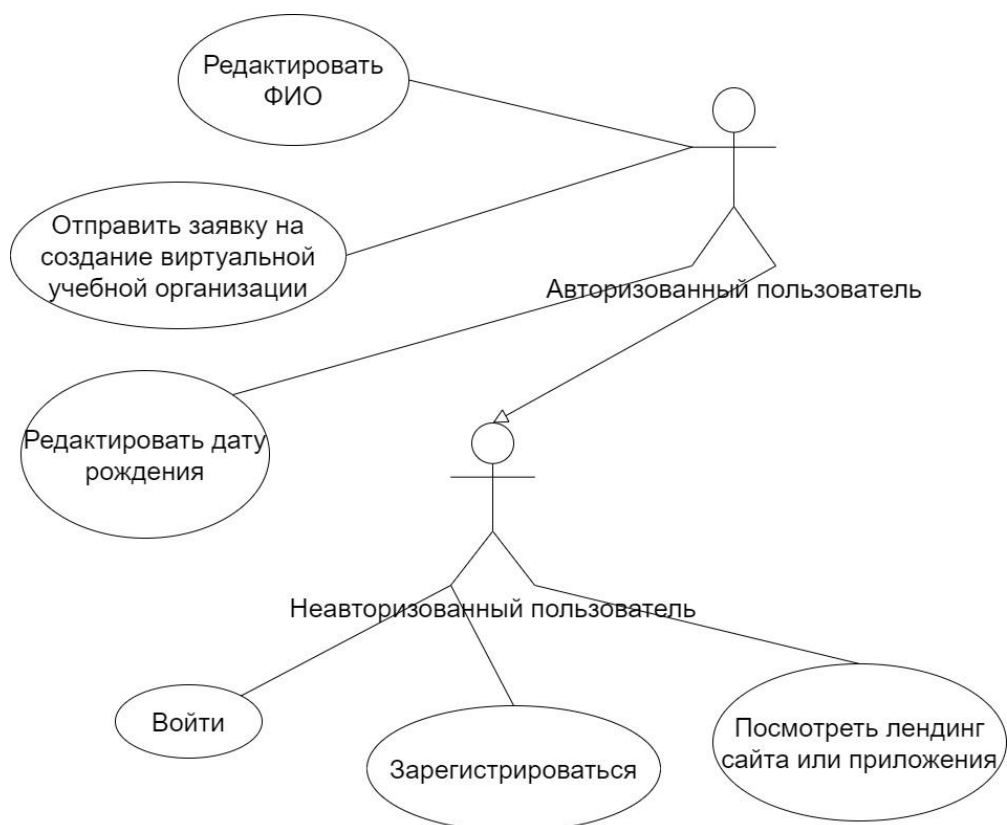


Рисунок А.1 - Диаграмма вариантов использования для пользователей с ролями «Неавторизованный пользователь» и «Авторизованный пользователь»



Рисунок А.2 - Диаграмма вариантов использования для пользователя с ролью «Ученик»

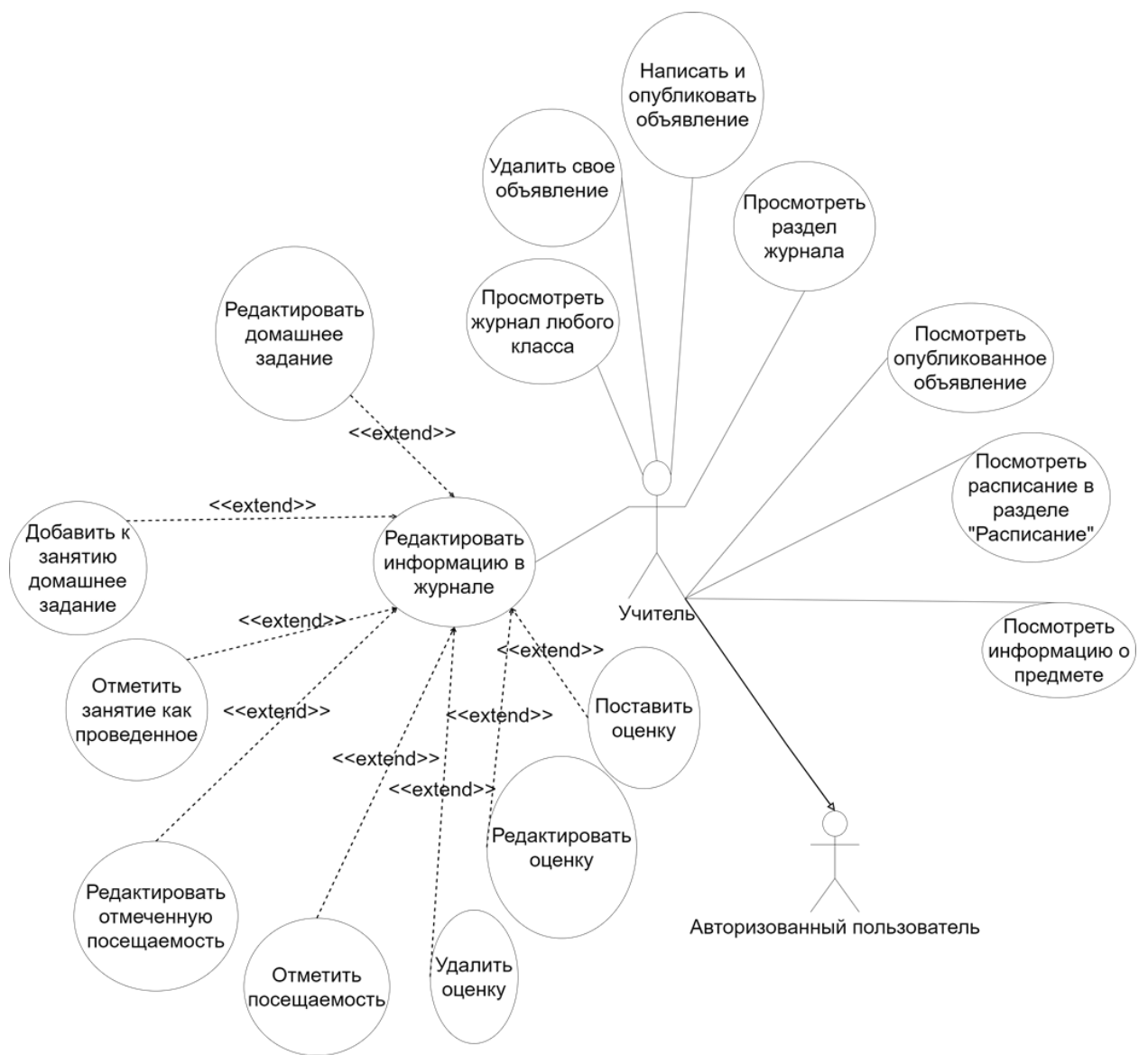


Рисунок А.3 - Диаграмма вариантов использования для пользователя с ролью «Учитель»



Рисунок А.4 - Диаграмма вариантов использования для пользователя с ролью «Локальный администратор»

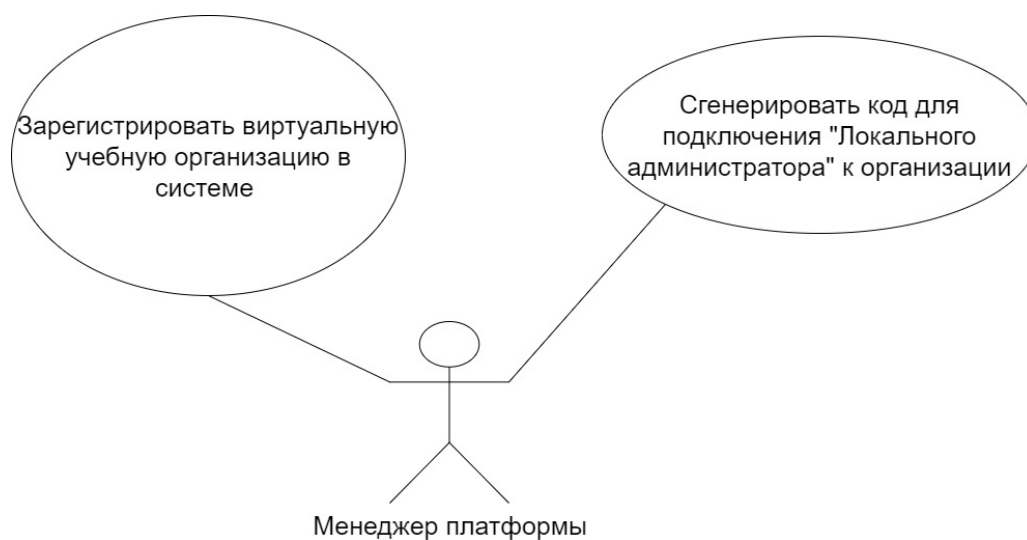


Рисунок А.5 - Диаграмма вариантов использования для пользователя с ролью «Менеджер платформы»