Содержание

[Введение 3](#_Toc41751244)

[1. Постановка задачи 5](#_Toc41751245)

[1.1 Описание предметной области 5](#_Toc41751246)

[1.2 Обзор существующих аналогов 6](#_Toc41751247)

[1.3 Функциональное назначение 9](#_Toc41751248)

[2. Проектирование задачи 11](#_Toc41751249)

[2.1 Логическое моделирование 11](#_Toc41751250)

[2.2 Проектирование интерфейса 12](#_Toc41751251)

[2.3 Выбор и обоснование инструментов разработки 16](#_Toc41751252)

[3. Программная реализация 19](#_Toc41751253)

[3.1 Физическая структура 19](#_Toc41751254)

[3.2 Описание разработанных модулей 21](#_Toc41751255)

[4. Тестирование 28](#_Toc41751256)

[5. Применение 38](#_Toc41751257)

[5.1 Описание процесса установки и запуска приложения 38](#_Toc41751258)

[5.2 Руководство пользователя 38](#_Toc41751259)

[6. Экономический раздел 62](#_Toc41751260)

[6.1 Технико-экономическое обоснование проекта разработки мобильного приложения на платформе Android «Расписание учебного заведения» 62](#_Toc41751261)

[7. Охрана труда 81](#_Toc41751262)

[7.1 Проектирование и расчет искусственного освещения светотехнической среды для работы с ПЭВМ при использовании мобильного приложения на платформе Android «Расписание учебного заведения» 81](#_Toc41751263)

[Заключение 91](#_Toc41751264)

[Список используемых источников 92](#_Toc41751265)

[Приложение А (Обязательное) Код программы 94](#_Toc41751266)

# Введение

Тема дипломного проекта: мобильное приложение на платформе Android «Расписание учебного заведения».

Роль учебных заведений невозможно переоценить. Они оказывают большое влияние на развитее всей страны. На учебные заведения государство возлагает социальную ответственность за развитие регионов, тем самым стимулируя активное взаимодействие с органами местной власти, общественными организациями и промышленными предприятиями. В масштабах государства, учебные заведения выполняют роль хранителей, создателей и распространителей знания, способствуют обогащению людей и общества. В масштабах региона, вузы выступают в роли организации, которая вносит значительный вклад в развитие самого региона, предоставляет услуги, которые обогащают социальную, культурную и экономическую жизнь. От того какие виды услуг может оказать учебное заведение, будет зависеть дальнейшее развитие региона.

Высшие учебные заведения оказывают следующий перечень услуг:

* услуги в сфере образования. К таким услугам можно отнести: создание новых специальностей, открытие курсов повышения квалификации, непрерывный процесс образования;
* услуги в научно производственной и инновационной сфере. К ним относятся: фундаментальные, прикладные, мониторинговые исследования, студенческая научная работа, патентование и охрана интеллектуальной собственности, инновационное проектирование, создание новых технологий;
* услуги в сфере информации. К ним относятся: тиражирование научных исследований, научных статей, образовательных программ, развитие локальных и корпоративных сетей Интернет;
* услуги в социально-культурной сфере. Социально – культурное развитее общества, приобщение к культуре и творчеству.

А первое дело при организации учебного процесса – создание расписания и его дистрибуция учащимся.

Таким образом объектом исследования данной работы является деятельность учебного заведения.

Предметом исследования – составление расписания.

Целью выполняемой работы является разработка автоматизированной информационной системы составления и дистрибуции расписания.

Задачами дипломного проекта являются:

* анализ существующих мобильных приложений для предоставления расписания учащимся;
* проектирование структуры программного средства, в том числе логическое моделирование и проектирование интерфейса;
* реализация программного средства;
* тестирование и отладка разработанного программного средства.

Автоматизация процессов создания и распространения расписания позволит учащимся более оперативно отслеживать изменения в учебном процессе, а учебному заведению сократит время на создание расписания.

В ходе разработки дипломного проекта был создана пояснительная записка. Пояснительная записка состоит из семи разделов.

В первом разделе «Постановка задачи» описывается предметная область, функциональное назначение разрабатываемого ПО, а также анализируются существующие аналоги.

Во втором разделе «Проектирование задачи» определяется структура проекта, проектируется графический пользовательский интерфейс и выбираются инструменты разработки.

В разделе «Программная реализация» описываются разработанные модули проекта и его структура на уровне файлов.

В четвертом разделе «Тестирование» производится тестирование разработанного программного средства, создается тест-кейс и подводятся итоги тестирования приложения.

В пятом разделе «Применение» описывается процесс инсталляции приложения и излагается исчерпывающее руководство пользователя.

В экономическом разделе производится технико-экономическое обоснование разработки программного средства для дипломного проекта.

В седьмом разделе рассматривается такой вопрос по охране труда, как проектирование и расчет искусственного освещения светотехнической среды для работы с ПЭВМ.

# 1. Постановка задачи

## **1.1 Описание предметной области**

При описании предметной области расписания стоит начать с определения понятия учебного заведения.

Учебное заведение – общее название школ, гимназий, техникумов, колледжей, университетов и других мест получения знаний.

Расписание – вид календаря, для которого указана информация о предстоящих (планируемых или потом произошедших) событиях. Оформляется обычно в виде таблицы [8].

У всех учебных заведений есть какое-либо расписание, будь то расписание занятий или расписание работы столовой. Расписание занятий составляется из пар или уроков, в зависимости от учебного заведения.

Урок – временной промежуток длительностью 45 минут – 1 академический час (длительность может изменяться по различным причинам, например, сокращенные уроки в честь праздничных дней), во время которого учащиеся должны овладеть изучаемым материалом. На уроке, помимо учащихся так же присутствует учитель. Его основная роль – воспитание учащихся и помощь в освоении материала [9].

Учащийся – тот, кто учится в учебном заведении.

Пара – урок, длительность которого увеличение в 2 раза, то есть 1 час 35 минут (5 минут выделены на перерыв межу первыми и вторыми 45 минутами) – 2 академических часа. Пары ведут преподаватели [12].

Слова преподаватель и учитель – синонимы, но исторически сложилось, что для различных учебных заведений принято использовать различные термины. Так, например, в вузах университетах и колледжах пары ведут преподаватели, а в школах и гимназиях уроки ведут учителя.

Современные технологии проникли во все сферы человеческой жизни, образование не является исключением. Сейчас трудно представить жизнь современного человека без таких устройств как смартфон, планшет или таких технологий как интернет, мессенджеры, электронная почта и социальные сети.

На фоне всех этих технических инноваций, прочно закрепившихся в нашей жизни, традиционное расписание, представленное в форме доски, кажется недостаточным.

Каждая современная организация зачастую имеет своё информационное пространство, в которое входит сайт образовательного учреждения, электронно-образовательный портал, электронно-библиотечная система, внутренняя система взаимодействия сотрудников (корпоративный портал, мессенджер, электронная почта). На фоне этого невозможность узнать своё расписание с применением современных гаджетов вызывает достаточно сильный дискомфорт.

Создание мобильного приложения для просмотра расписания позволило бы любому студенту, находясь в любом месте, узнать какие занятия у него будут, в какое время и в какой именно аудитории, помогут найти преподавателя в случае необходимости. Так же появляется возможность мгновенно отслеживать изменения в графике учебного процесса.

## **1.2 Обзор существующих аналогов**

Приложений для расписания бывает два вида.

Первый - общего назначения, когда пользователь сам создает расписание и обновляет его. Оно может составляться для любой предметной области.

Второй – создается специально для конкретной цели. Само расписание скачивается из внешних источников, и пользователь не может его редактировать, только просматривать.

Поскольку мое программное средство относится ко 2 категории, рассматривать аналоги мы будем также 2 категории.

Первый аналог – приложение «Расписание БГУИР», рабочее окно которого изображено на рисунке 1.1 [5]. С его помощью можно просматривать расписание сразу для нескольких групп, читать информацию о преподавателях, менять шрифт и язык.

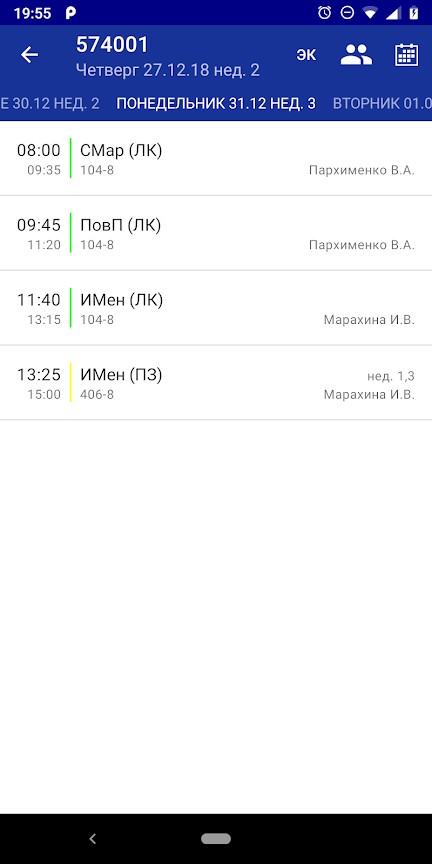


Рисунок 1.1 – Мобильное приложение «Расписание БГУИР»

Из недостатков этого приложения можно отметить неудобный интерфейс. Если вам необходимо посмотреть расписание на пятницу, придется делать четыре свайпа влево. Свайп влево – достаточно неудобный жест, особенно если пользоваться телефоном одной рукой. А тут их целых четыре. Намного удобнее было бы сделать навигацию по дням недели списком, или поместить все расписание в один столбец, так, чтобы пролистать вниз, к пятнице, достаточно было бы сделать всего один сильный свайп вверх.

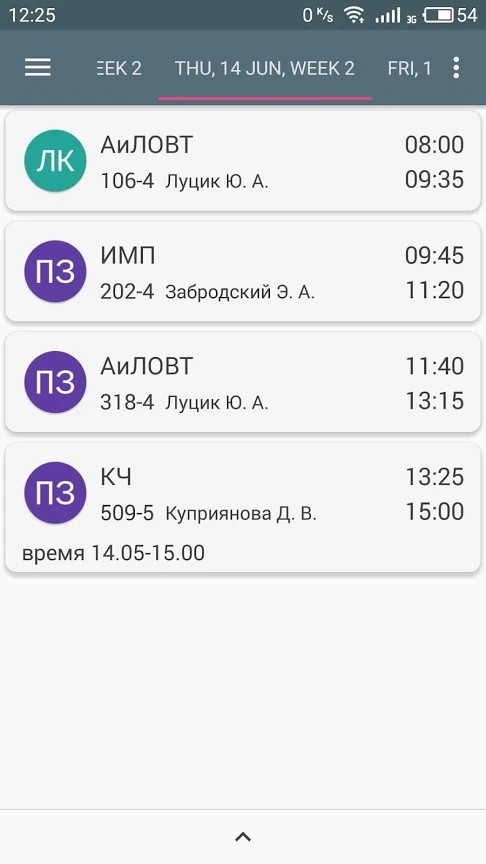


Рисунок 1.2 – Мобильное приложение «BSUIR Schedule»

Следующий аналог – мобильное приложение «BSUIR Schedule», рабочее окно которого изображено на рисунке 1.2 [6]. Оно обладает намного более широким функционалом. Помимо чтения расписания для нескольких групп, мобильное приложение «BSUIR Schedule» позволяет изменять локальную версию расписания, просматривать место нахождения преподавателей, имеет виджет на рабочий стол, и самое главное, темную тему.

Сразу же можно отметить такой же недостаток, как и у приложения «Расписание БГУИР»: неудобный интерфейс. Затем если присмотреться, то большинство функциональных кнопок спрятались за кнопку меню, что увеличивает время доступа к функциям этих кнопок. Это еще один недостаток интерфейса.

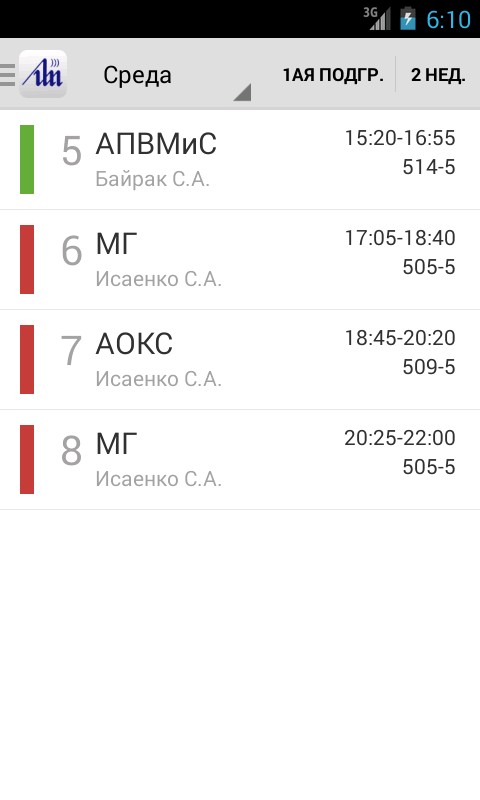


Рисунок 1.3 – Мобильное приложение «БГУИР Расписание»

И последний аналог – приложение «БГУИР Расписание», рабочее окно которого изображено на рисунке 1.3 [7]. Функционал этого приложения похож на функционал «BSUIR Schedule»: он позволяет изменять локальную версию расписания, имеет виджет на рабочий стол, темную тему, но отсутствует возможность посмотреть, где находится преподаватель.

В отличие от своих предшественников, это приложение добавило выпадающее меню для выбора для недели, что является достаточно неплохим решением по сравнению с решениями предыдущих приложений. У приложения минималистичный дизайн, все нужные кнопки находятся под рукой, нет ничего лишнего, или чего-нибудь, что мешало бы просмотру расписания.

На основе преимуществ и недостатков уже существующих решений можно понять, как сделать свое приложение лучше, на что следует обратить внимание, а от чего следует избавиться, чтобы получить качественный продукт.

## **1.3 Функциональное назначение**

У каждого учебного заведения есть расписание. Однако не каждое учебное заведение может его автоматизировать. Автоматизация процесса создания и распространения расписания позволит существенно облегчить и ускорить работу учебных заведений. Облегчение внедрения процессов автоматизации процессов создания и распространения расписания является целью написания программного обеспечения.

Для обеспечения работоспособности мобильного приложения необходимо разработать веб сервер, а для заполнения расписания админ-панель.

Задачами веб-сервера являются:

* хранение, добавление, редактирование, удаление преподавателей;
* хранение, добавление, редактирование, удаление аудиторий;
* хранение, добавление, редактирование, удаление групп;
* хранение, добавление, редактирование, удаление предметов;
* хранение, редактирование, создание расписания на основе существующих преподавателей, аудиторий, групп и предметов;
* хранение, добавление, редактирование, удаление пользователей;
* проверка прав доступа у пользователей при различных операциях;
* авторизация пользователей.

Задачами админ-панели являются:

* предоставление пользователю интерфейса для просмотра, добавления, редактирования, удаления преподавателей;
* предоставление пользователю интерфейса для просмотра, добавления, редактирования, удаления аудиторий;
* предоставление пользователю интерфейса для просмотра, добавления, редактирования, удаления групп;
* предоставление пользователю интерфейса для просмотра, добавления, редактирования, удаления предметов;
* предоставление пользователю интерфейса для просмотра, редактирования и создания расписания на основе существующих преподавателей, аудиторий, групп и предметов;
* предоставление администраторам интерфейса для редактирования, создания, удаления пользователей;
* авторизация пользователей.

Задачами мобильного приложения являются:

* получение и отображение расписания выбранной группы;
* получение и отображение информации о преподавателях;
* поиск по преподавателям;
* кэширование данных;
* создание условий для возможности просмотра расписания и информации о преподавателях оффлайн.

Админ панель должна иметь развитый и удобный пользовательский интерфейс. В ходе разработки программы следует использовать такие элементы, как таблицы для более информативного представления данных. Не следует пренебрегать такими элементами графического интерфейса как иконки и картинки. Это позволит повысить воспринимаемость данных.

В админ панели так же должен быть реализован лёгкий поиск данных. В каком бы окне программы не находился пользователь, он должен иметь возможность осуществить поиск интересующих его данных, набрав на клавиатуре последовательность символов, содержащихся в искомом элементе. Веб приложение должно выполнить поиск по набранной последовательности символов и выдавать результаты, подходящие под запрос поиска. Такой поиск должен быть контекстным, т.е. зависеть от того, в каком окне он выполняется и по каким реквизитам ведётся поиск. Поиск увеличит удобство работы пользователя с программой и повысит качество разрабатываемого приложения.

Мобильное приложение должно уметь работать без интернета, показывая пользователю последнюю полученную информацию. Так же приложение должно уведомлять пользователя при отсутствии соединения с интернетом.

Таким образом разработанное программное средство позволит любому учебному заведению без особых проблем внедрить процессы автоматизации создания и распространения программного обеспечения.

# 2. Проектирование задачи

## **2.1 Логическое моделирование**

В процессе проектирования программной средства были выбраны две диаграммы: диаграмма вариантов использования и диаграмма сущность-связь.

Диаграмма вариантов использования – диаграмма, отражающая отношения между актёрами и прецедентами и являющаяся составной частью модели прецедентов, позволяющей описать систему на концептуальном уровне.

Прецедент – возможность моделируемой системы (часть её функциональности), благодаря которой пользователь может получить конкретный, измеримый и нужный ему результат. Прецедент соответствует отдельному сервису системы, определяет один из вариантов её использования и описывает типичный способ взаимодействия пользователя с системой. Варианты использования обычно применяются для спецификации внешних требований к системе.

Диаграмма сущность-связь – модель данных, позволяющая описывать концептуальные схемы предметной области. Она используется при высокоуровневом проектировании баз данных. С её помощью можно выделить ключевые сущности и обозначить связи, которые могут устанавливаться между этими сущностями.

Диаграмма вариантов использования представлена в графической части на плакате ДП62.49307.305.002ПЛ. На этой диаграмме мы можем наблюдать трех актеров – учащегося, администратора и пользователя. Меньше всего возможностей у простого пользователя – он может просматривать только пары и преподавателей. Для выполнения этих действий авторизация не обязательна. Второй по значимости ролью является пользователь. Пользователь может просматривать полностью все расписание: пары, преподавателей, группы, аудитории, предметы, а также пользователь может редактировать их. Для работы, пользователь обязан пройти регистрацию. Ролью с самым большим количеством прав является администратор. Эта роль включает в себя те же возможности редактирования и просмотра расписания, что и роль «пользователь», однако администратор так же обладает возможностью просматривать и редактировать других пользователей и администраторов. Как и пользователь, администратор обязан пройти авторизацию.

Диаграмма сущность-связь представлена в графической части на плакате ДП62.49307.305.001ПЛ. На этой диаграмме видны связи межу сущностями. Самая главная сущность – пара. Эта сущность включает в себя время начала и время конца, а также ссылки на все остальные сущности. Этими сущностями являются: преподаватель, группа, аудитория и предмет. Все эти сущности связаны с сущностью пары связью один ко многим, что означает, что пару может вести только одни преподаватель, пара может вестись только у одной группы, пара может проходить только в одной аудитории и на паре может изучаться только один предмет.

## **2.2 Проектирование интерфейса**

Поскольку программное средство включает себя 3 части: админ панель, веб-сервер и мобильное приложение, необходимо спроектировать 2 интерфейса (веб-сервер в интерфейсе не нуждается).

2.2.1 Проектирование интерфейса веб-сервера.

Проектирование интерфейса начинается с того, что мы хотим отображать. В нашем случае это однотипные данные. Самый лучший способ отображение однотипных данных – таблицы. Поэтому для отображения всех данных будут использованы таблицы. Количество строк таблицы будет определятся количеством данных, которые мы хотим отобразить, а количество столбцов таблицы будет определятся тем, сколько полей будет у объекта, экземпляры которого мы хотим отобразить.

Для навигации будет использовано меню слева от главного рабочего окна.

В меню текущий пункт будет выделен жирным шрифтом.

Так же необходимо предусмотреть способ выйти из аккаунта в любой момент. Он будет реализован через выпадающее меню по иконке пользователя, находящееся в правом верхнем углу.

Кнопка добавления элементов будет располагаться справа, над таблицей с данными. Для удаления элементов необходимо будет нажать на чекбоксы напротив записей, которые необходимо удалить. Чекбоксы располагаются в самом левом столбце. Для редактирования данных необходимо нажать на кнопку «Изменить» напротив каждой записи. Кнопка «Изменить» будет располагаться в самом правом столбце таблицы.

Окно для ввода текста для фильтрации, слева над таблицей с данными.

Для экрана с добавлением данных необходимо отказаться от использования таблиц, поскольку необходимо показывать за раз максимум один элемент.

Поля, необходимые для заполнения будут расположены в столбик один под другим.

В самом низу слева будет кнопка сохранения записи с надписью: «Сохранить» и добавления ее в таблицу.

Итого, получим интерфейс просмотра записей, представленный на рисунке 2.1.

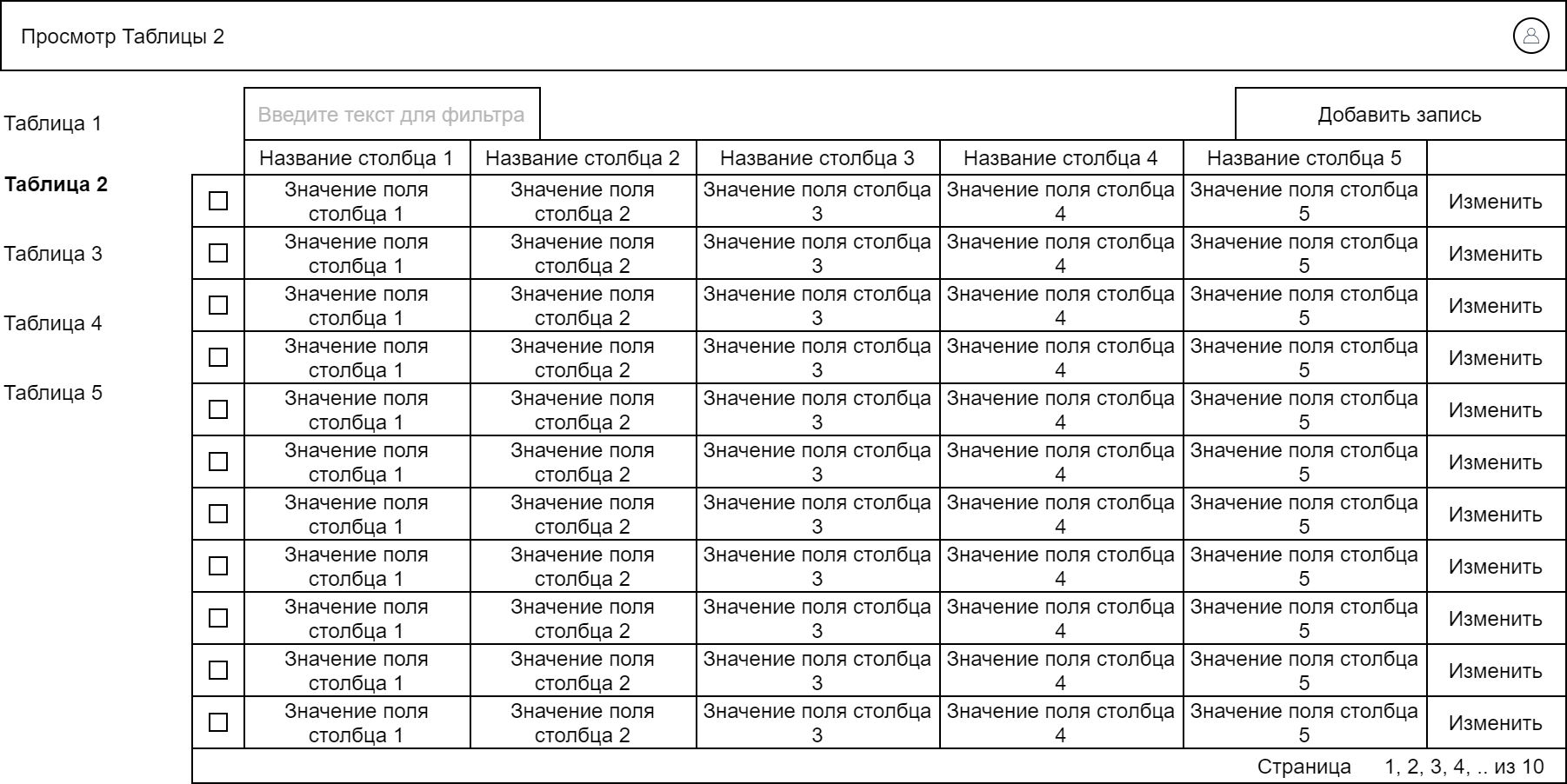


Рисунок 2.1 – Экран просмотра записей

Интерфейс создания записей представлен на рисунке 2.2

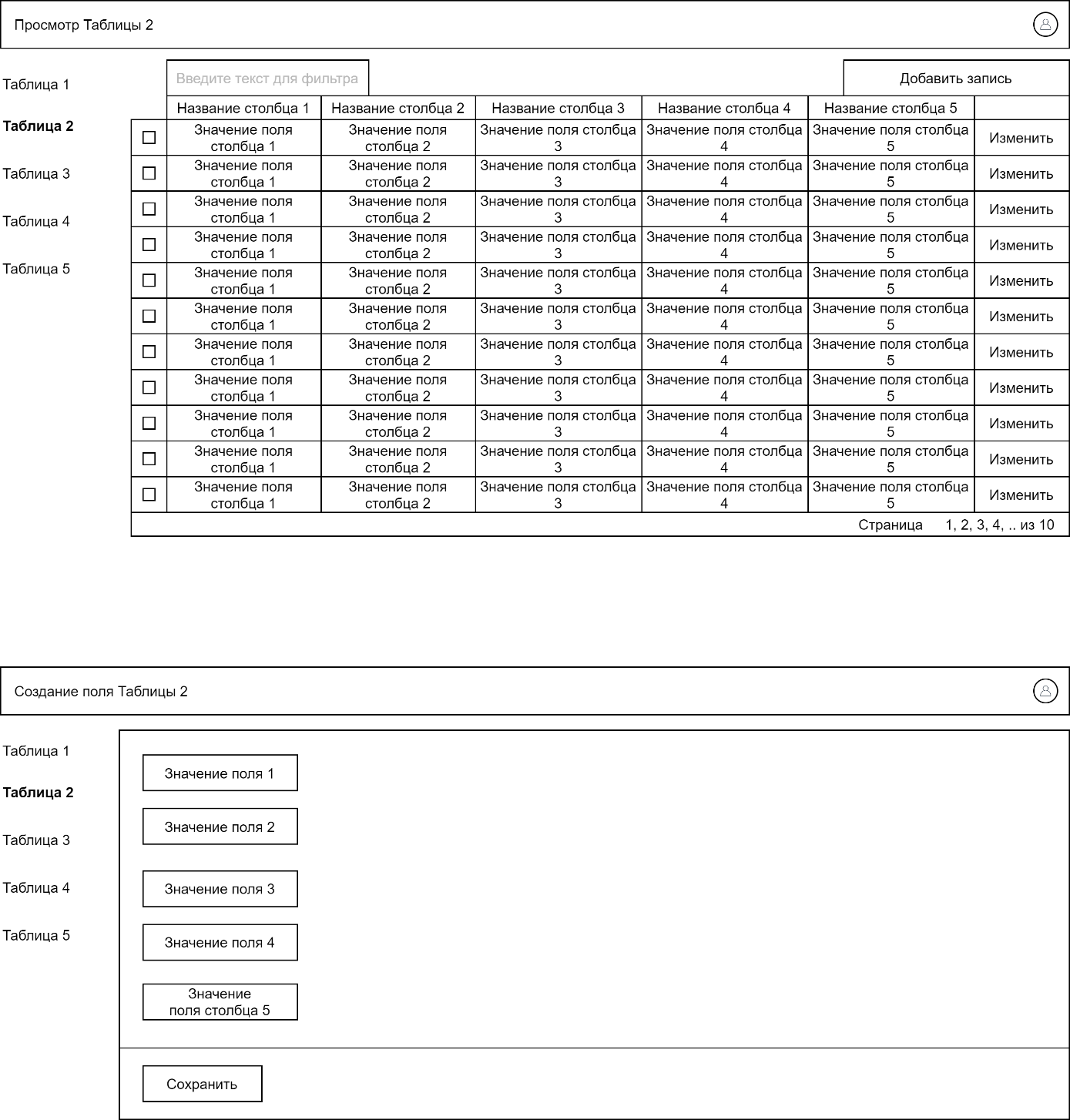


Рисунок 2.2 – Экран создания и редактирования записей

Экран редактирования записей ничем не будет отличатся от экрана добавления записей за исключением того, что поля при открытии экрана будут предварительно заполнены текущими значениями полей записи и надписи в заголовке экрана.

Таким образом был рассмотрен интерфейс админ панели, необходимый для полноценного использования приложением, а именно: навигация по таблицам, создание, просмотр, фильтрация, редактирование и удаление записей.

2.2.2 Проектирование интерфейса мобильного приложения.

В мобильном приложении будет два экрана – экран просмотра расписания и экран просмотра информации о преподавателях. Внизу каждого экрана будет находится меню, чтобы пользователь мог в любой момент переключиться на интересующий его экран.

Экран расписания представляет собой список подэкранов с расписанием на день. Между расписаниями на разные дни можно переключаться, используя свайп влево и свайп вправо.

В самом верху экрана находится индикатор того, расписание на какой день сейчас отображается.

Сам экран с расписанием на день представляет из себя набор окон с парами, расположенные в вертикальной ориентации. В окне с парой можно найти время начала, время конца, ФИО преподавателя, название предмета и номер аудитории. Макет экрана с расписанием представлен на рисунке 2.3.



Рисунок 2.3 – Макет экрана с расписанием

Экран с преподавателями выполнен в похожем стиле, он представляет из себя вертикальный список преподавателей.

В каждом элементе списка будет находится имя, фамилия, отчество, номер телефона и кратка информация о преподавателе.

Поскольку поле «Номер преподавателя» содержит в себе номер, на который предполагается звонить, необходимо, чтобы при нажатии на номер телефона открывался набор номера с введенным номера преподавателя. Чтобы позвонить по нему станется только нажать на кнопку вызова.

Предполагается, что преподаватель может указать в разделе краткой информации email адрес. Поэтому необходимо, чтобы в тексте находился email адреса и по нажатию на ним пользователя перебрасывало в приложение для отправки email сообщений с адресом, на который нажал пользователь в графе «получатель» у email сообщения.

Помимо этого, вверху экрана будет располагаться строка поиска по преподавателям.

Макет экрана с преподавателями представлен на рисунке 2.4.

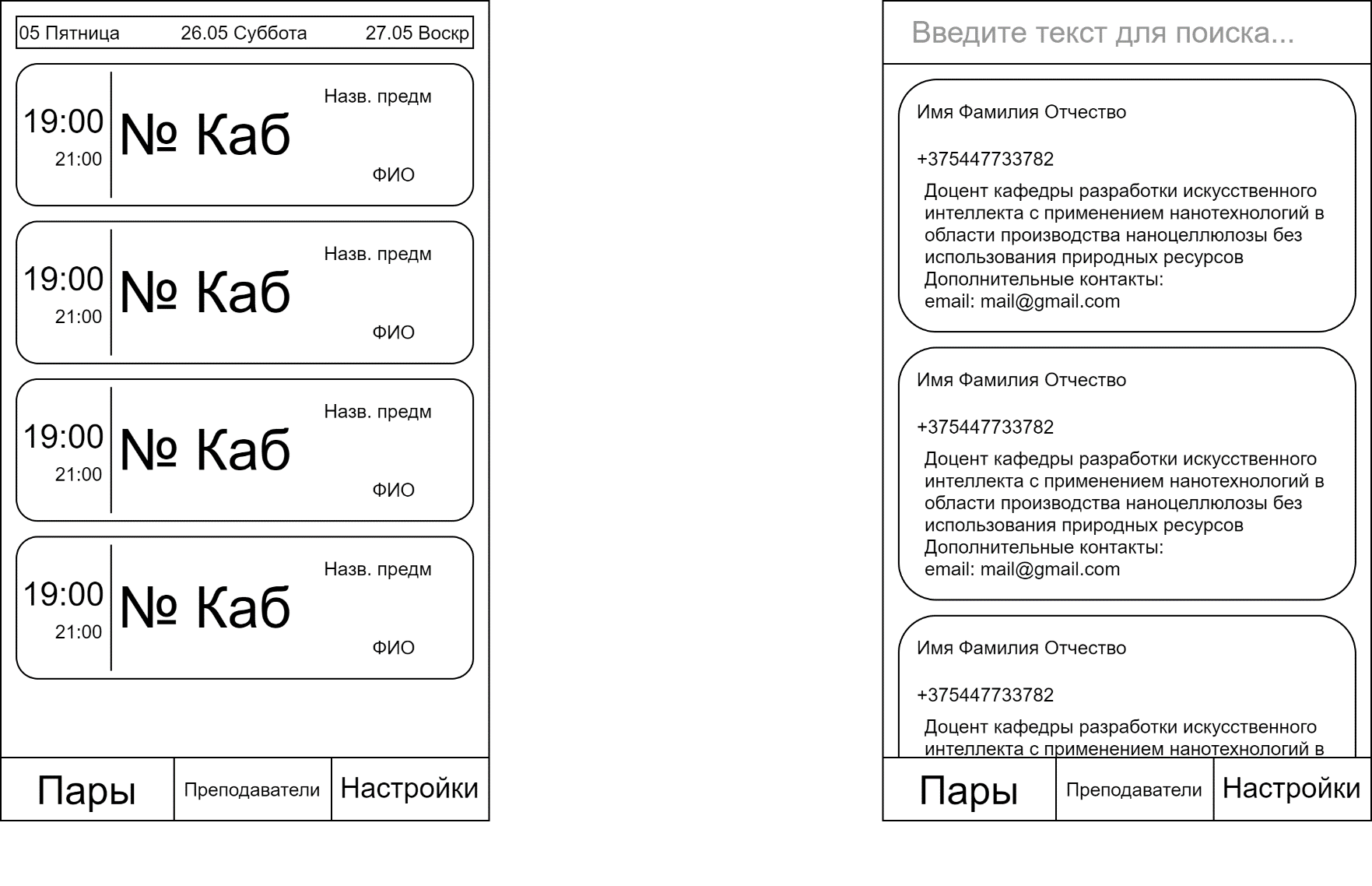


Рисунок 2.4 - Макет экрана с преподавателями

При нажатии на кнопку с настройками снизу вылезает окно с настройками. У каждой настройки есть свое действие. Макет экрана с открытыми настройками представлен на рисунке 2.5.

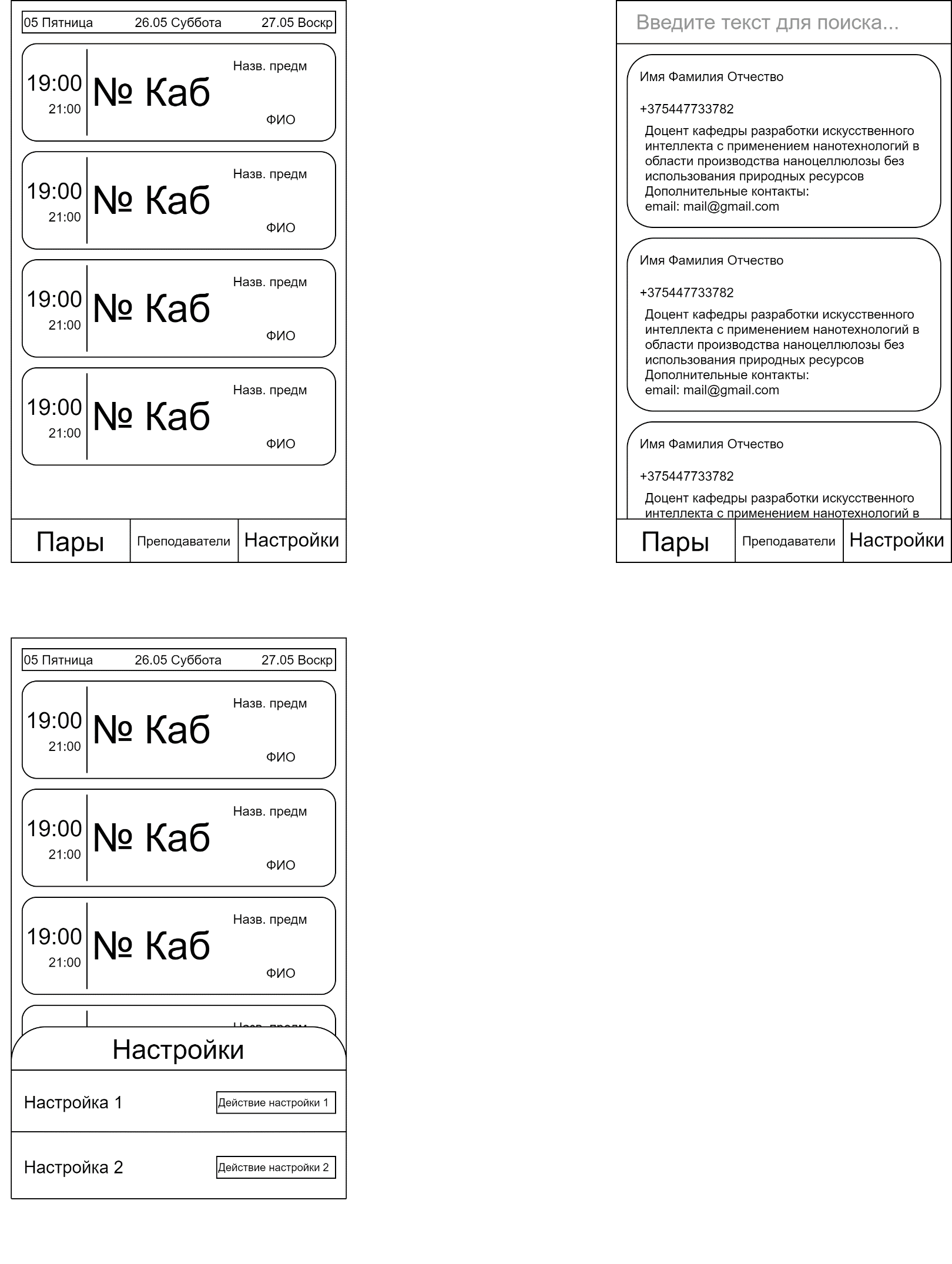


Рисунок 2.5 – Макет экрана с открытыми настройками

Таким образом был рассмотрен интерфейс мобильного приложения, необходимый для полноценного его использования, а именно: просмотр расписания, просмотр преподавателей и взаимодействие с настройками.

## **2.3 Выбор и обоснование инструментов разработки**

Разработка программного средства состоит из 3 частей: веб сервер, админ панель и мобильное приложение. Каждая часть представляет из себя программное обеспечение из различных областей, поэтому описывать выбор инструмента разработки стоит для каждой части по-отдельности.

2.3.1 Выбор и основание инструментов разработки веб-сервера

Для разработки веб-сервера был выбран язык программирования Kotlin. Выбор пал на Kotlin по следующим причинам:

* статическая типизация (не будет непредвиденных преобразований типов);
* язык компилируется в JVM байткод, что позволяет без проблем использовать все Java библиотеки;
* язык null-безопасен (если переменная может быть null, это надо явно указывать), что позволяет избегать NullPointerException;
* в языке много синтаксического сахара, что позволяет ускорить разработку;
* язык немногословный, что так же положительно влияет на скорость разработки.

Для языка Kotlin был выбран бэкэнд фреймворк Javalin из-за его простоты, легковесности, скорости, богатых встроенных возможностях и популярности.

В качестве базы данных была использована система управления базами данных PostgreSQL. Выбор сделан в сторону этой СУБД поскольку она имеет большое количество преимуществ над другими СУБД, такие как: большее количество типов данных, поддержка JSON и возможность создавать свои типы.

В качестве IDE была выбрана среда разработки IntellijIDEA. Эта IDE обладает функциями автозаполнения кода, встроенной отладки приложений, возможностью выполнять код во время отладки, проверки на синтаксические и логические ошибки, а также эта IDE обладает огромным количеством настроек, что позволяет настроить ее под себя.

2.3.2 Выбор и основание инструментов разработки админ панели

Админ панель написана на React.js с использованием фреймворка React Admin. React.js – библиотека для написания веб сайтов. К ее преимуществам перед чистым js можно отнести: переиспользование компонентов, декларативный подход написания интерфейсов, а также обновления компонентов в реальном времени.

Фреймворк React Admin позволяет создавать таблицы сданными, редактировать их, добавлять и удалять данные, поддерживает таблицы со связями, позволяет реализовать авторизацию, контролировать доступ к данным и многое другое, что идеально подходит под необходимый функционал админ панели.

В качестве IDE была выбрана Visual Studio Code. Эта IDE обладает функциями автозаполнения, проверки синтаксических ошибок, а также возможность производить отладку приложения.

2.3.3 Выбор и основание инструментов разработки мобильного приложения

Мобильное Android было написано на языке Kotlin. Помимо преимуществ этого языка, указанных выше, можно добавить нативную поддержку Android’ом Kotlin’a. Так же Google, компания, которая активно разрабатывает Android, призвала всех разработчиков переходить на Kotlin.

Интерфейс приложения был написан с помощью языка разметки XML. Android нативно поддерживает этот способ создания интерфейсов, что стало главной причиной в пользу выбора этого метода создания разметки экранов.

Для работы с сетью была использована библиотека OkHttp. Она обладает большим функционалом, высоким быстродействием и эту библиотек просто использовать.

Для хранения и кэширования данных была использована ORM библиотека Room DB. По сути эта библиотека является очень удобной надстройкой над встроенной в Andorid БД SQLite3. В отличие от SQLite3, Room работает не с сырыми запросами, а с Java объектами, представляющими собой ту или иную сущность базы данных. Это позволяет значительно сократить время разработки приложения.

Для асинхронной работы была использована библиотека RxJava в паре с билиотекой RxAndroid. Эта библиотека позволяет выполнять асинхронные действия, создавать потоки данных, управлять ими в функциональном стиле, код с этой библиотекой легко читается, что позволяет легко реализовать асинхронность в Android приложении.

В качестве IDE была выбрана среда разработки Android Studio. Эта среда разработки обладает встроенными Android эмуляторами, встроенной функцией отладки Android устройств, автозаполнением, большими возможностями в области кастомизации, а также большим сообществом.

Таким образом были описаны инструменты разработки всех 3 частей: веб сервера, админ панели и мобильного приложения.

# 3. Программная реализация

## **3.1 Физическая структура**

Разработка приложения состоит из 3 частей:

* разработка серверной части приложения;
* разработка админ-панели;
* разработка клиентского мобильного приложения.

Все вышеупомянутые части приложения должны между собой как-то взаимодействовать. Для этого была выбрана классическая клиент-серверная архитектура. На рисунке 3.1 представлена схема работы разработанной архитектуры [14].

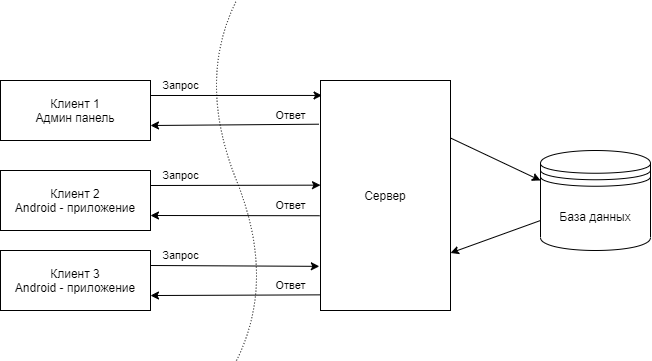


Рисунок 3.1 – Схема клиент-серверной архитектуры

Самая главная часть приложения – сервер. Только через него можно добавлять, изменять и читать расписания.

Общение с сервером происходит по протоколу HTTP. Для получения данных особых разрешений или авторизации не надо. Однако для запросов, которые подразумевают редактирование, добавление или удаление данных, необходимо быть авторизованным. А для редактирования других пользователей, активный пользователь должен обладать правами администратора.

При обращении к серверу, сервер делает запрос в базу данных, представляет данные в стандартизированном виде и отправляет их в ответ на запрос.

К серверу будут обращаться 2 вида клиентов – админ-панель и Android-приложения. Только админ панель будет располагать возможностями для редактирования расписания, следовательно, в админ-панели авторизация обязательна. Для получения данных авторизация не нужна, следовательно, в Android-предложении авторизации нет и им может пользоваться любой желающий без ограничений.

Сначала был разработан веб-сервер. Веб сервер содержит следующие папки:

* common – папка, содержащая классы, логика которых используется в других классах;
* login – папка, содержащая логику авторизации;
* response – папка, содержащая логику обработки ошибок;
* static – папка, содержащая логику отправки статических html страниц;
* utils – папка с утилитарными классами;
* user – папка с фалами для авторизации, работы с ролями и пользователями.

В остальных папках group, lesson, office, teacher, discipline лежат файлы, отвечающие за обработку запросов на получение, добавление, редактирование и удаление групп, пар, аудиторий, преподавателей и предметов соответственно.

Для хранения данных, веб-сервер использует базу данных. Схема спроектированной базы данных представлена на рисунке 3.2.

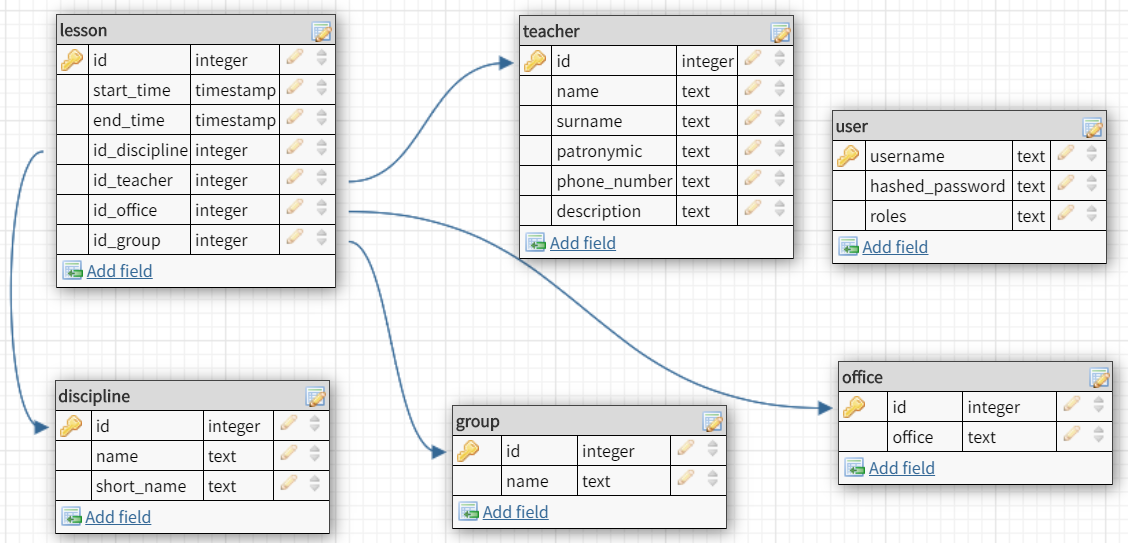


Рисунок 3.2 – Схема базы данных, используемая на сервере

В базе данных имеется таблица с пользователя и основная таблица, которая хранит непосредственно сами расписания и 4 остальныеместо для ния ая часть андроид приложений состоит мых важных из них.лицасписаниярограммного обеспечения.олв о оаооо двлаовгакоаа таблицы, которые необходимы для отсутствия дубликатов данных, в главной таблице используются только ссылки на индексы дополнительных таблиц. Стоит отметить, что в таблице пользователей пароли хранятся в захешированном виде. Это необходимо, чтобы не было возможности подсмотреть чужой пароль. Кончено, простое хеширование не идеальный вариант, т.к. если пользователь использует частые пароли, как 123456, то существуют таблицы, в которых прописаны хеши и их возможное изначальное значение, и таким образом возможно будет узнать пароль, однако в нашем случае хватит и простого хеширования.

Следующим этапом разработки стало написание админ-панели. Структура админ-панели состоит из 2-х папок: public и src. В папке public содержатся статические файлы, такие как html-файлы, картинки, текстовые файлы. В папке src содержатся сами исполняемые исходные файлы. Именно в этой папке лежат скрипты разработанной админ-панели.

Заключающим и самым важным этапом разработки стало написание клиентского мобильного приложения. Исходные файлы приложения распределены по 2 папкам: res и src. В корне проекта лежит файл manifest.xml в котором указаны основные компоненты приложения, чтобы Android знал, каким функционалом обладает приложение. В папке src содержатся 7 папок – teachers, di, schedule, settings, tools и view, где лежат логика для работы с преподавателями, файлы dependency injection, логика для работы с расписанием, логика настроек, инструментальные классы и кастомные view-элементы соответственно. В папке res лежат файлы, содержащие все ресурсы приложения, а также разметки всех экранов.

Таким образом была разработана система ПО, решающая проблему невозможности посмотреть расписание

## **3.2 Описание разработанных модулей**

Разработка приложения состоит из 3 частей:

* разработка серверной части приложения;
* разработка админ-панели;
* разработка клиентского мобильного приложения.

Сначала был разработан веб-сервер. Его эндпоиты для работы с админ-панелью реализуют собой полноценный REST API.

В текущей реализации REST API имеются следующие возможности: получение всех элементов выборки, добавление элемента в выборку, получение данных и конкретном элементе выборки по номеру, изменение данных конкретного элемента в выборке по номеру, удаление элемента из выборки по номеру.

За получение всех элементов отвечает HTTP метод GET на эндпоинте с названием типа объекта. В теле ответа будет находится JSON массив запрошенных объектов.

За добавление элемента отвечает HTTP метод POST на эндпоинте с названием типа объекта. В тело запроса необходимо поместить JSON объект, который необходимо добавить в выборку. В теле ответа будет находится JSON объект добавленного элемента. Это необходимо на случай, если сервер произведет какую-либо обработку данных и изменит их. Клиент должен об этом знать.

За получение элемента по id отвечает HTTP метод GET на эндпоинте с названием типа объекта и id элемента. В теле ответа будет находится JSON объект найденного элемента с индексом id.

За редактирование элемента по id отвечает HTTP метод PUT на эндпоинте с названием типа объекта и id элемента. В тело запроса необходимо поместить JSON объект, измененного элемента. В теле ответа будет находится JSON объект отредактированного элемента.

За удаление элемента по id отвечает HTTP метод DELTE на эндпоинте с названием типа объекта и id элемента. В теле ответа будет находится JSON объект удаленного элемента.

Для работы с пользователями, выполняя запрос необходимо иметь валидный токен в HTTP header «Authorization».

На рисунке 3.3 находится схематичное представление REST API преподавателей. Для других объектов, таких как предметы, группы, аудитории и пары схема идентичная, за исключением названия объекта в пути к эндпоинту. Так, для получения всех групп, путь будет «/groups», для получения всех аудиторий, путь будет «/offices» и так далее.

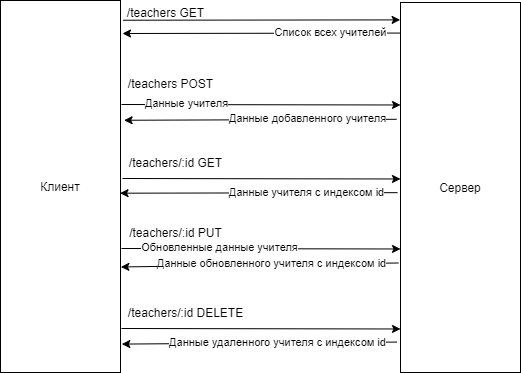


Рисунок 3.3 – Схематичное представление REST API преподавателей

Веб-сервер состоит из файлов, сгруппированных по логике работы, которую они исполняют.

Так, в папке common содержатся следующие файлы CanDeleteUtils.kt – файл с функциями-расширениями, Filter.kt – файл с общей логикой фильтрации, Interactor.kt – файл с абстрактным классом, содержащий логику работы с БД.

В папке login расположился файл LoginController,kt, отвечающий за авторизацию пользователей.

В папке response – файл ResponseController.kt, отвечающий за обработку ошибок.

В папке static – файл StaticController.kt, отвечающий за отправку статических html страниц.

В папке utils содержатся утилитарные классы AddHeader.kt, Enums.kt, GetAllParams.kt и Path.kt, которые используются по всему проекту.

В папке discipline лежит файл Discipline.kt, который выражает собой структуру объекта «предмет», файл DisciplineController.kt, который содержит маппинг эндпоинтов сервера на функции работы с БД и файл DisciplineInteractor.kt, который является реализацией абстрактного класса Interactor и служит для работы с БД.

В папке group лежит файл Group.kt, который выражает собой структуру объекта «группа», файл GroupController.kt, который содержит маппинг эндпоинтов сервера на функции работы с БД и файл GroupInteractor.kt, который является реализацией абстрактного класса Interactor и служит для работы с БД.

В папке lesson лежит файл Lesson.kt, который выражает собой структуру объекта «пара», файл LessonController.kt, который содержит маппинг эндпоинтов сервера на функции работы с БД и файл LessonInteractor.kt, который является реализацией абстрактного класса Interactor и служит для работы с БД.

В папке office лежит файл Office.kt, который выражает собой структуру объекта «аудитория», файл OfficeController.kt, который содержит маппинг эндпоинтов сервера на функции работы с БД и файл OfficeInteractor.kt, который является реализацией абстрактного класса Interactor и служит для работы с БД.

В папке teacher лежит файл Teacher.kt, который выражает собой структуру объекта «преподаватель», файл TeacherController.kt, который содержит маппинг эндпоинтов сервера на функции работы с БД и файл TeacherInteractor.kt, который является реализацией абстрактного класса Interactor и служит для работы с БД.

В папке user лежит файл User.kt, который выражает собой пользователя, файл UserController.kt, который содержит маппинг эндпоинтов сервера на функции работы с БД и файл UserInteractor.kt, который является реализацией абстрактного класса Interactor и служит для работы с БД. Эндпоинты в этой папке доступны только пользователям с правами «Администратор».

В корне проекта лежат 2 файла – Main.kt, содержащий логику инициализации веб-сервера и DB.kt, содержащий логику соединения с базой данных.

Следующим этапом разработки стало написание админ-панели.

Файл index.html является главной точкой входа в приложение. Файл favicon.ico представляет собой картинку, отображаемую на вкладке в браузере.

После загрузки html страницы первым делом загружается и запускается файл App.js. Он содержит в себе координирующую логику приложения. Он отвечает за то, что, как и где показывать.

Файл index.css содержит стили страницы.

Файл customBottomToolbar.js содержит кастомный компонент нижней панели на экране редактирования.

Файл customDeleteToolbar.js содержит кастомный компонент контекстного меню.

Файл customTextField.tsx содержит кастомное поля для отображения текста.

Файл deleteButton.tsx содержит компонент кнопки удаления.

Файл deleteButtonEdit.tsx содержит компонент кнопки удаления в контекстном меню.

Файл discipline.js содержит разметку для экранов просмотра, создания, изменения и редактирования предметов.

Файл group.js содержит разметку для экранов просмотра, создания, изменения и редактирования групп.

Файл lesson.js содержит разметку для экранов просмотра, создания, изменения и редактирования пар.

Файл office.js содержит разметку для экранов просмотра, создания, изменения и редактирования аудиторий.

Файл teacher.js содержит разметку для экранов просмотра, создания, изменения и редактирования преподавателей.

Файл authController.js содержит в себе логику авторизации и выхода из текущего пользователя.

Файл url.js хранит в себе ip адрес бэкэнд-сервера.

Заключающим и самым важным этапом разработки стало написание клиентского мобильного приложения.

Все экраны приложения выполнены на архитектуре MVP. MVP расшифровывается как Model-View-Presenter.

В разработанном приложении роль View выполняет Android компонент Fragment. View сообщает Presenter о пользовательских взаимодействиях, таких как нажатия кнопки, ввод текста.

Presenter слушает пользовательские взаимодействия и на их основе решает, как изменить Model. Model – это данные в приложении. Например, номер текущей группы.

Presenter также слушает обновления модели и на их основе решает, как обновить View.

Схема архитектуры MVP представлена на рисунке 3.4 [13].

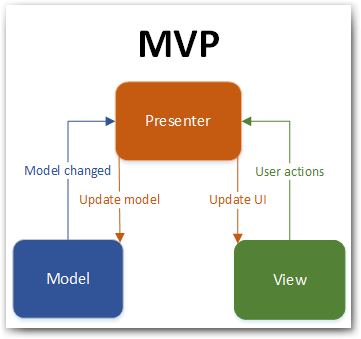


Рисунок 3.4 – Схема архитектуры MVP

В корне проекта лежат файлы MainActivity.kt, MainPresenter.kt, MainView.kt и MainDialogs.kt, которые в совокупности образуют главный экран приложения. MainView представляет из себя интерфейс, который создает контракт, по которому можно изменять визуальные компоненты. Класс MainPresenter.kt содержит логику работы главного экрана и через интерфейс MainView общается с классом MainActivity, который его реализует. Класс MainActivity, реализуя интерфейс MainView представляет из себя прослойку между Android и абстрактным экраном, который представлен через интерфейс MainView. MainDialogs.kt также является частью этой связки и отвечает за отображение диалогов.

Также в корне проекта лежит файл ScheduleApplication.kt, в котором настраивается DI во время запуска приложения.

Класс ApplicationModule, находящийся в папке di, отвечает за внедрение в DI зависимостей, которые зависимы от Android.

В папке tools находится интерфейс файл Extesnsions.kt, который содержит функции расширения, в частности для перевода «Density-independent Pixels» в обычные экранные пиксели.

В папке view находится файл CustomPagerTitleStrip.java, который представляет из себя кастомный компонент карусели.

В папке settings находятся файлы SettingsViewImpl.kt, SettingsPresenter.kt, SettingsView.kt, SettignsInretactor.kt и SettingsRepository.kt, которые в совокупности образуют нижнюю выдвигалку с настройками. SettingsView представляет из себя интерфейс, который создает контракт, по которому можно изменять визуальные компоненты. Класс SettingsPresenter.kt содержит логику работы настроек и через интерфейс SettingsView общается с классом SettingsViewImpl, который его реализует. Класс SettingsViewImpl, реализуя интерфейс SettingsView представляет из себя прослойку между Android и абстрактными настройками, которые представлены через интерфейс SettingsView. Классы SettignsInretactor и SettingsRepository служат для получения и сохранения настроек во внутреннее хранилище Android.

В папке schedule находятся файлы ScheduleFragment.kt, SchedulePresenter.kt, ScheduleView.kt и ScheduleRepository.kt, которые в совокупности образуют экран с расписанием. ScheduleView представляет из себя интерфейс который создает контракт по которому можно изменять визуальные компоненты. Класс SchedulePresenter.kt содержит логику работы экрана расписания и через интерфейс ScheduleView общается с классом ScheduleFragment, который его реализует. Класс ScheduleFragment, реализуя интерфейс ScheduleView представляет из себя прослойку между Android и абстрактным экраном, который представлен через интерфейс ScheduleView. Класс ScheduleRepository служит для получения расписания из интернета и кэша. Помимо этих классов тут так же лежит файл Schedule.kt, который представляет из себя данные о расписании, файлы ScheduleAdapter.kt и ScheduleViewHolder.kt, которые служат специальным адаптером для android-компонента RecyclerView, файл DayView, который предоставляет из себя часть экрана, показывающую один день из расписания, файлы SchedulePagerAdapter.kt и ScheduleViewHolder.kt, которые служат специальным адаптером для еще одного android-компонента ViewPager2.

В папке teacher находятся файлы TeacherFragment.kt, TeacherPresenter.kt, TeacherView.kt и TeacherRepository.kt, которые в совокупности образуют экран с преподавателями. TeacherView представляет из себя интерфейс, который создает контракт, по которому можно изменять визуальные компоненты. Класс TeacherPresenter.kt содержит логику работы экрана с преподавателями и через интерфейс TeacherView общается с классом TeacherFragment, который его реализует. Класс TeacherFragment, реализуя интерфейс TeacherView представляет из себя прослойку между Android и абстрактным экраном, который представлен через интерфейс TeacherView. Класс TeacherRepository служит для получения информации о преподавателях из интернета и кэша. Помимо этих классов тут так же лежит файл Teacher.kt, который представляет из себя данные о преподавателе, файлы TeacherAdapter.kt и TeacherViewHolder.kt, которые служат специальным адаптером для android-компонента RecyclerView.

В папке res находятся статические, неисполняемые файлы ресурсы приложения.

Файл activity\_main.xml содержит разметку экрана главного окна.

Файл bottom\_sheet\_settings.xml содержит разметку нижней выдвигающейся панели и настроек.

Файл fragment\_schedule.xml содержит разметку экрана с расписанием.

Файл fragment\_teachers.xml содержит разметку экрана с преподавателями.

Файл item\_lesson.xml содержит разметку одного дня расписания.

Файл item\_teacher.xml содержит разметку одного преподавателя.

Файл search.xml содержит разметку для строки поиска преподавателей.

Таким образом были рассмотрены структуры файлов веб-сервера, админ-панели и мобильного приложения, а также архитектура веб-сервера и мобильного приложения. Архитектура разработанных компонентов позволит их легко изменять, модифицировать, добавлять новые функции, а также поддерживать существующие.

Программный код разработанных модулей представлен в приложении А.

# 4. Тестирование

Тестирование – процесс исследования, испытания программного продукта, имеющий своей целью проверку соответствия между реальным поведением программы и её ожидаемым поведением на конечном наборе тестов, выбранных определенным образом.

Статическое тестирование производится без запуска программного кода продукта. Тестирование осуществляется путем анализа программного кода или скомпилированного кода. Анализ может производиться как вручную, так и с помощью специальных инструментальных средств. Целью анализа является раннее выявление ошибок и потенциальных проблем в продукте.

Примерами ошибок, которые потенциально можно выявить с помощью автоматического статического тестирования, могут быть:

* утечки ресурсов (утечки памяти, не освобождаемые файловые дескрипторы и так далее);
* возможность переполнения буфера;
* ситуации частичной (неполной) обработки ошибок.

В отличие от статического, динамическое тестирование производится путем запуска продукта и проверки его функционала. Проверка осуществляется с помощью ручного или автоматического выполнения заранее подготовленного набора тестов. Примеры динамического тестирования:

* модульное тестирование;
* интеграционное тестирование;
* приемочное тестирование.

Чтобы удостовериться в правильной работе программного продукта был проведен тест-кейс.

Тест-кейс — это профессиональная документация тестировщика, последовательность действий, направленная на проверку какого-либо функционала, описывающая как прийти к фактическому результату.

В таблице 4.1 представлено тестирование программы.

Таблица 4.1 – Тест-кейсы программного средства

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Шаги | Ожидаемый результат | Результат |
| Запуск мобильного приложения | 1. Тап по иконке приложения | 1. Приложение открылось | Пройдено |
| Продолжение таблицы 4.1 | |  |  |
| Название | Шаги | Ожидаемый результат | Результат |
| Первое указание группы в мобильном приложении | 1. Начать вводить в единственное доступное поле номер группы | 1. В поле ввода появляются введенные символы | Пройдено |
| 2. Дождаться появления подсказки с доступными номерами групп | 2. Появляется выпадающая подсказка с номерами групп |
| 3. Нажать на номер группы | 3. В поле ввода отображается полный номер группы |
| 4. Нажать на кнопку «Готово» | 4. Окно закрылось, на главном экране отображается расписание для введенной группы |
| Изменение группы в мобильном приложении | 1. Нажать на кнопку «Настройки» | 1. Снизу выдвигается окно с настройками | На главном экране отобра-жается распи-сание старой группы. Не пройдено. |
| 2. В появившемся окне нажать на кнопку «Изменить» напротив номера группы | 2. Открывается диалог с вводом номера группы |
| 3. Начать вводить номер группы в поле ввода | 3. В поле ввода появляются введенные символы |
| 4. Дождаться появления подсказки с доступными номерами групп | 4. Появляется выпадающая подсказка с номерами групп |
| 5. Нажать на номер группы | 5. В поле ввода отображается полный номер группы |
| 6. Нажать на кнопку «Готово» | 6. Окно закрылось, на главном экране отображается расписание для введенной группы |
| Продолжение таблицы 4.1 | |  |  |
| Название | Шаги | Ожидаемый результат | Результат |
| Просмотр расписания в мобильном приложении | 1. Ввести номер группы, расписание для которой заведомо известно | 1. Группа выбрана | Пройдено |
| 2. Просмотреть расписание этой группы | 2. Расписание совпадает с заранее известным расписанием |
| Просмотр преподавателя в мобильном приложении | 1. Перейти на экран с преподавателями | 1. Отображается страница с преподавателями | Пройдено |
| 2. Найти преподавателя, информация о котором заранее известна и тапнуть по нему. | 2. Станица с преподавателями открыта на преподавателе, информация о котором известна. |
| 2. Просмотреть информацию о преподавателе | 2. Информация о преподавателе совпадает с заранее известной информацией |
| Поиск преподавателя в мобильном приложении | 1. Перейти на экран с преподавателями | 1. Отображается страница с преподавателями | Пройдено |
| 2. Ввести часть имени, фамилии, отчества, номера телефона или описания преподавателя | 2. В списке преподавателей остались только те преподаватели, в имени, фамилии, отчестве, номере телефона или описании содержатся введённые символы |
| Продолжение таблицы 4.1 | |  |  |
| Название | Шаги | Ожидаемый результат | Результат |
| Ввод номера несущес- твующей группы в мобильном приложении | 1. Начать вводить в поле ввода номера группы номер несуществующей группы | 1. В поле ввода появляются введенные символы | Пройдено |
| 2. Убедится, что подсказок не появляется | 2. Подсказок не появляется |
| 3. Нажать на кнопку «Готово» | 3. Окно закрылось, на главном экране отображается надпись, что для текущей группы расписания нет |
| Добавление группы в админ-панели | 1. Перейти на страницу с группами | 1. Открывается страница с группами | Пройдено |
| 2. Нажать на кнопку «Создать» | 2. Открывается страница с добавлением групп |
| 3. Ввести данные группы | 3. Введенные данные группы отображаются |
| 4. Нажать на кнопку «Сохранить» | 4. Добавленная группа видна в списке всех групп |
| Редактирование группы в админ-панели | 1. Перейти на страницу с группами | 1. Открывается страница с группами | Пройдено |
| 2. Нажать на кнопку «Редактировать» напротив группы, которую необходимо отредактировать | 2. Открывается страница редактирования выбранной группы |
| 3. Ввести новые данные группы | 3. Введенные данные группы отображаются |
| 4. Нажать на кнопку «Сохранить» | 4. Отредактированная группа видна в списке всех групп |
| Продолжение таблицы 4.1 | | | |
| Название | Шаги | Ожидаемый результат | Результат |
| Удаление группы в админ-панели | 1. Перейти на страницу с группами | 1. Открывается страница с группами | Пройдено |
| 2. Выбрать группы, которые необходимо удалить | 2. Напротив выбранных групп появляется галочка |
| 3. Нажать на кнопку «Удалить» | 3. Выбранные группы больше не отображаются в списке всех групп |
| Поиск по группам в админ-панели | 1. Перейти на страницу с группами | 1. Открывается страница с группами | Пройдено |
| 2. Ввести часть номера группы в поле поиска | 2. В списке отображаются только те группы, в которых содержится введенная в поле поиска строка. |
| Добавление преподавателей в админ-панели | 1. Перейти на страницу с преподавателями | 1. Открывается страница с преподавателями | Пройдено |
| 2. Нажать на кнопку «Создать» | 2. Открывается страница с добавлением преподавателей |
| 3. Ввести данные преподавателя | 3. Введенные данные преподавателя отображаются |
| 4. Нажать на кнопку «Сохранить» | 4. Добавленный преподаватель виден в списке всех преподавателей |
| Продолжение таблицы 4.1 | | | |
| Название | Шаги | Ожидаемый результат | Результат |
| Редактирование преподавателей в админ-панели | 1. Перейти на страницу с преподавателями | 1. Открывается страница с преподавателями | Пройдено |
| 2. Нажать на кнопку «Редактировать» напротив преподавателя, которого необходимо отредактировать | 2. Открывается страница редактирования выбранного преподавателя |
| 3. Ввести новые данные преподавателя | 3. Введенные данные преподавателя отображаются |
| 4. Нажать на кнопку «Сохранить» | 4. Отредактированный преподаватель виден в списке всех преподавателей |
| Удаление преподавателей в админ-панели | 1. Перейти на страницу с преподавателями | 1. Открывается страница с преподавателями | Пройдено |
| 2. Выбрать преподавателей, которые необходимо удалить | 2. Напротив выбранных преподавателей появляется галочка |
| 3. Нажать на кнопку «Удалить» | 3. Выбранные преподаватели больше не отображаются в списке всех преподавателей |
| Поиск по преподавателям в админ-панели | 1. Перейти на страницу с преподавателями | 1. Открывается страница с преподавателями |  |
| Продолжение таблицы 4.1 | | | |
| Название | Шаги | Ожидаемый результат | Результат |
| Поиск по преподавателям в админ-панели | 2. Ввести часть ФИО преподавателя в поле поиска | 2. В списке отображаются только те преподаватели, в которых содержится введенная в поле поиска строка. | Пройдено |
| Добавление предметов в админ-панели | 1. Перейти на страницу с предметами | 1. Открывается страница с предметами | Пройдено |
| 2. Нажать на кнопку «Создать» | 2. Открывается страница с добавлением предметов |
| 3. Ввести данные предмета | 3. Введенные данные предмета отображаются |
| 4. Нажать на кнопку «Сохранить» | 4. Добавленный предмет виден в списке всех предметов |
| Редактирование предметов в админ-панели | 1. Перейти на страницу с предметами | 1. Открывается страница с предметами | Пройдено |
| 2. Нажать на кнопку «Редактировать» напротив предмета, который необходимо отредактировать | 2. Открывается страница редактирования выбранного предмета |
| 3. Ввести новые данные предмета | 3. Введенные данные предмета отображаются |
| 4. Нажать на кнопку «Сохранить» | 4. Отредактированный предмета виден в списке всех предметов |
| Удаление предметов в админ-панели | 1. Перейти на страницу с предметами | 1. Открывается страница с предметами |  |
| 2. Выбрать предметы, которые необходимо удалить | 2. Напротив выбранных предметов появляется галочка |
| Продолжение таблицы 4.1 | | | |
| Название | Шаги | Ожидаемый результат | Результат |
| Удаление предметов в админ-панели | 3. Нажать на кнопку «Удалить» | 3. Выбранные предметы больше не отображаются в списке всех предметов | Пройдено |
| Поиск по предметам в админ-панели | 1. Перейти на страницу с предметами | 1. Открывается страница с предметами | Пройдено |
| 2. Ввести часть номера предмета в поле поиска | 2. В списке отображаются только те предметы, в которых содержится введенная в поле поиска строка. |
| Добавление аудиторий в админ-панели | 1. Перейти на страницу с аудиториями | 1. Открывается страница с аудиториями | Пройдено |
| 2. Нажать на кнопку «Создать» | 2. Открывается страница с добавлением аудиторий |
| 3. Ввести данные аудитории | 3. Введенные данные аудитории отображаются |
| 4. Нажать на кнопку «Сохранить» | 4. Добавленная аудитория видна в списке всех аудиторий |
| Редактирование аудиторий в админ-панели | 1. Перейти на страницу с аудиториями | 1. Открывается страница с аудиториями | Пройдено |
| 2. Нажать на кнопку «Редактировать» напротив аудитории, которую необходимо отредактировать | 2. Открывается страница редактирования выбранной аудитории |
| 3. Ввести новые данные аудитории | 3. Введенные данные аудитории отображаются |
| 4. Нажать на кнопку «Сохранить» | 4. Отредактированная аудитория видна в списке всех аудиторий |
| Продолжение таблицы 4.1 | | | |
| Название | Шаги | Ожидаемый результат | Результат |
| Удаление аудиторий в админ-панели | 1. Перейти на страницу с аудиториями | 1. Открывается страница с аудиториями | Пройдено |
| 2. Выбрать аудитории, которые необходимо удалить | 2. Напротив выбранных аудиторий появляется галочка |
| 3. Нажать на кнопку «Удалить» | 3. Выбранные аудитории больше не отображаются в списке всех предметов |
| Поиск по аудиториям в админ-панели | 1. Перейти на страницу с аудиториями | 1. Открывается страница с аудиториями | Пройдено |
| 2. Ввести часть номера аудитории в поле поиска | 2. В списке отображаются только те аудитории, в которых содержится введенная в поле поиска строка. |
| Добавление пар в админ-панели | 1. Перейти на страницу с парами | 1. Открывается страница с парами | Пройдено |
| 2. Нажать на кнопку «Создать» | 2. Открывается страница с добавлением пар |
| 3. Ввести данные пары | 3. Введенные данные пары отображаются |
| 4. Нажать на кнопку «Сохранить» | 4. Добавленная пара видна в списке всех пар |
| Редактирование пар в админ-панели | 1. Перейти на страницу с парами | 1. Открывается страница с парами |  |
| 2. Нажать на кнопку «Редактировать» напротив пары, которую необходимо отредактировать | 2. Открывается страница редактирования выбранной пары |
| 3. Ввести новые данные пары | 3. Введенные данные пары отображаются |
| Продолжение таблицы 4.1 | | | |
| Название | Шаги | Ожидаемый результат | Результат |
| Редактирование пар в админ-панели | 4. Нажать на кнопку «Сохранить» | 4. Отредактированная пара видна в списке всех аудиторий | Пройдено |
| Удаление пар в админ-панели | 1. Перейти на страницу с парами | 1. Открывается страница с парами | Пройдено |
| 2. Выбрать пары, которые необходимо удалить | 2. Напротив выбранных пар появляется галочка |
| 3. Нажать на кнопку «Удалить» | 3. Выбранные пары больше не отображаются в списке всех пар |
| Фильтрация по парам в админ-панели | 1. Перейти на страницу с парами | 1. Открывается страница с парами | При выборе несколь-ких критериев фильтра-ции, применя-ется только первый устано-вленный. Не пройдено |
| 2. Нажать на кнопку фильтрации | 2. Из кнопки выпадает список критериев фильтрации |
| 3. Выбрать один или несколько критериев фильтрации | 3. Появляются поля, в которых можно выбрать объекты фильтрации |
| 4. Выбрать объекты фильтрации | 4. В списке отображается только записи с выбранными объектами фильтрации |

В ходе тестирования мобильного приложения было выявлено, что после изменения номера группы, расписание не обновляется автоматически и все еще показывает расписание на старую группу. Этот баг был сразу же исправлен.

В ходе тестирования админ панели было выявлено, что при выборе нескольких критериев фильтрации в разделе пар применяется только фильтрация по первому установленному критерию. В ходе разработки этот баг был устранен.

В результате тестирования остальных мест приложений, ошибок выявлено не было.

# 5. Применение

## **5.1 Описание процесса установки и запуска приложения**

Админ панель установке не подлежит, поскольку это сайт.

Для запуска админ панели необходимо перейти на сайт, на котором она расположена.

Веб-сервер в установке не нуждается, он представляет из себя один jar файл. Для его запуска необходимо, чтобы на компьютере была уставлена java 9 или новее. Для непосредственно запуска необходимо использовать команду «java –jar {имя jar-файла}».

Для установки приложения на Android устройство, необходимо скачать apk файл, полученный в результате сборки проекта на целевое устройство. В настройках телефона необходимо включить установку из неизвестных источников. После этого достаточно запустить apk файл. Откроется экран установки, в котором надо согласится с установкой приложения.

Для запуска приложения необходимо найти его в списке установленных приложений и нажать на него.

## **5.2 Руководство пользователя**

Веб-сервер не имеет графического интерфейса, поэтому пользоваться им напрямую невозможно. Однако мобильное приложение и админ-панель не будут без него работать. Так что даже несмотря на то, что мы не видим работу сервера, он должен быть запущен в обязательном порядке.

Админ-панель представляет из себя инструмент для работников учебных заведений, ответственных за составление расписания. В ней они могут редактировать, добавлять, удалять преподавателей, предметы, группы и наконец составлять расписания, а пользователями с правами администратора могут удалять, добавлять, изменять права других пользователей.

При запуске админ-панели после установки сервера пользователя будет встречать окно авторизации.

В нем пользователь должен ввести свои логин и пароль. Сам пользователь зарегистрироваться не может, иначе любой мог бы просматривать и редактировать расписание. Пользователя может зарегистрировать только администратор.

Если админ панель запускается в первый раз и никаких пользователей еще не было создано, следует авторизоваться под единственным доступным аккаунтом login: admin, password: admin. Этот аккаунт обладает правами администратора, это значит, что у него есть возможность создавать, редактировать и удалять пользователей. Сразу после авторизации, следует изменить свой пароль, чтобы никто не авторизовался, используя эти логин и пароль. Окно авторизации представлено на рисунке 5.1.

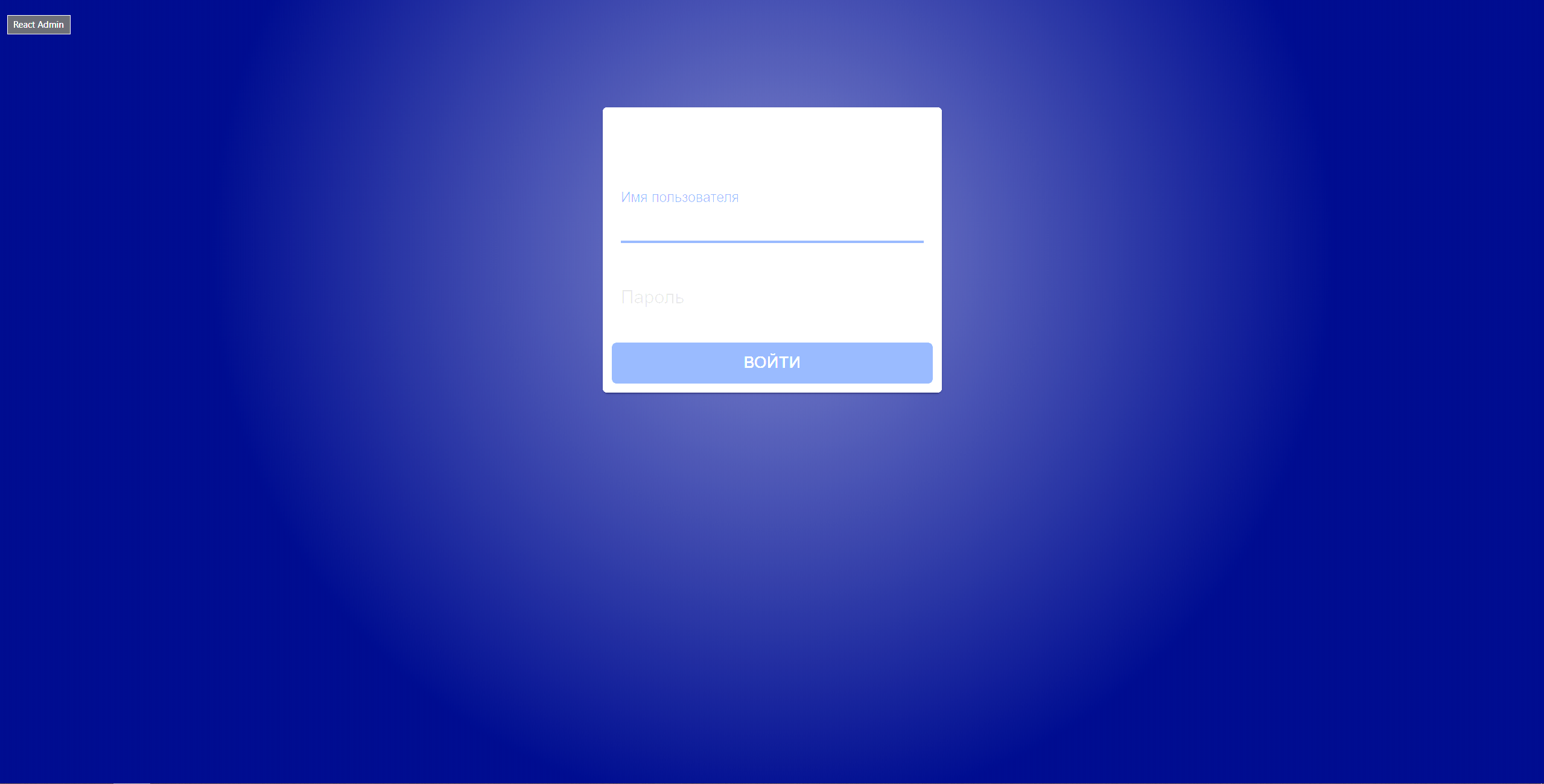


Рисунок 5.1 – Окно авторизации

В случае, если был введен неверный пароль, появится уведомление с соответствующей ошибкой. Окно с сообщением об ошибке представлено на рисунке 5.2.

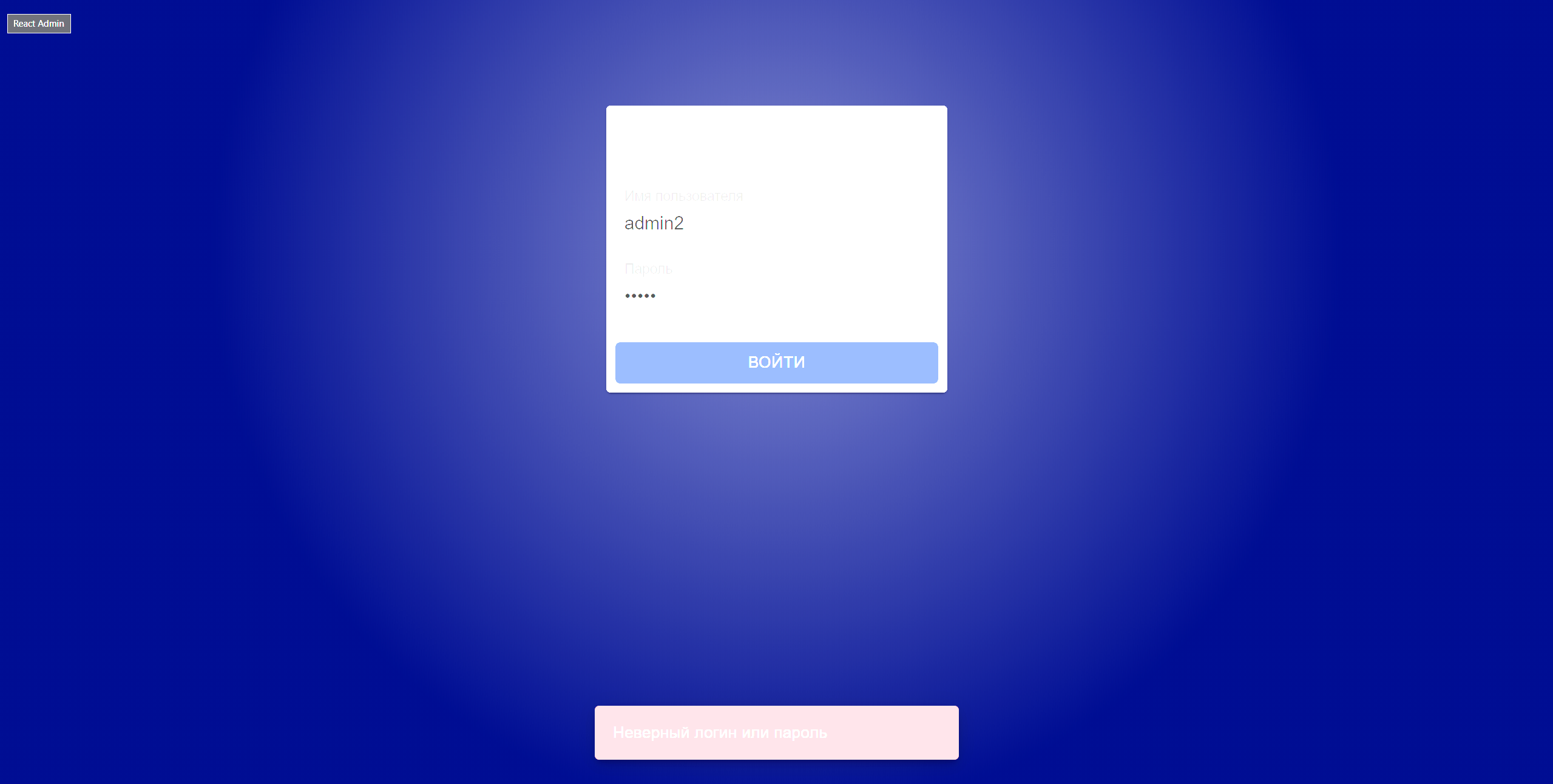


Рисунок 5.2 – Окно авторизации с сообщением об неверном логином или паролем

Для изменения пароля авторизованному пользователю, необходимо нажать на иконку пользователя в правом верхнем углу. Тогда появится контекстное меню, в котором следует выбрать пункт «Изменить пароль». Так же в этом контекстном меню можно просмотреть логин, под которым авторизирован пользователь. Контекстное меню изображено на рисунке 5.3.

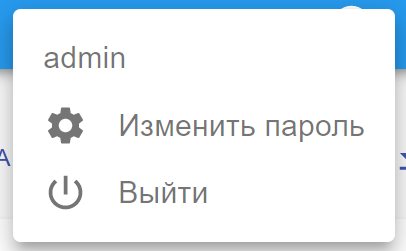


Рисунок 5.3 – Контекстное меню пользователя

После нажатия на эту кнопку, пользователь попадет на страницу изменения пароля. Форма изменения пароля представлена на рисунке 5.4.

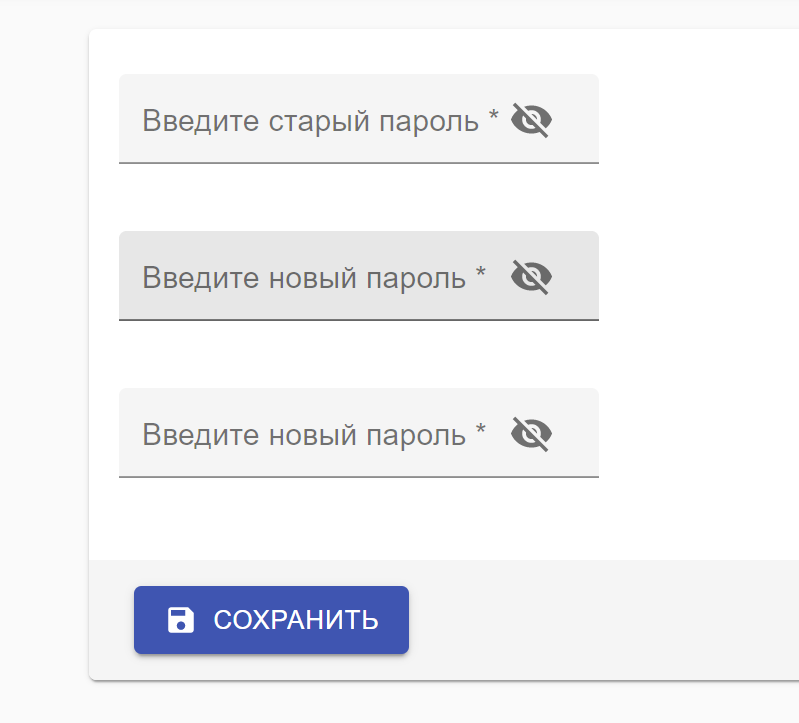


Рисунок 5.4 – Форма изменения пароля.

В форме необходимо указать текущий пароль, а также дважды ввести свой новый пароль. Это необходимо для отсутствия опечаток. После нажатия на кнопку «Сохранить», пароль будет изменен. Необходимость в повторной авторизации после изменения пароля отсутствует.

Если у пользователя есть права администратора, он увидит следующее меню, представленное на рисунке 5.5.

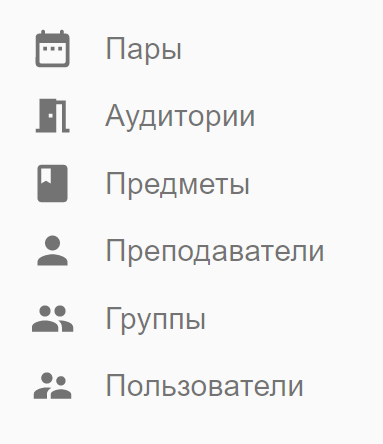


Рисунок 5.5 – Меню администратора.

Стоит обратить внимание на пункт «Пользователи». Он доступен только пользователям с правами администратора. Об этом пункте сейчас и пойдет речь.

Если до этого в систему не было добавлено пользователей, то на странице пользователей будет только один стандартный пользователь, представленный на рисунке 5.6.

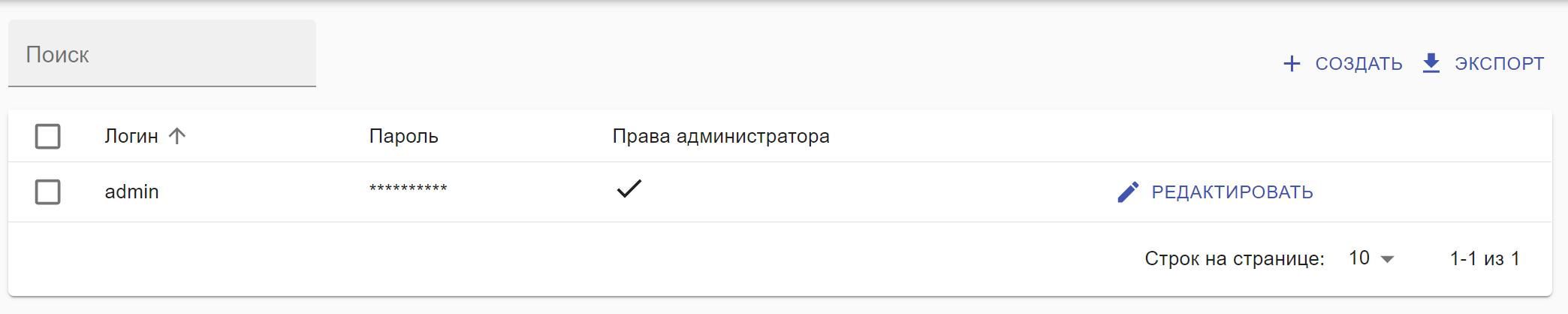


Рисунок 5.6 – Страница пользователей с одним стандартным пользователем

В случае, если надо выдать доступ к расписанию коллеге, не нужно давать ему логин и пароль своего аккаунта, необходимо создать для него пользователя. Для этого необходимо нажать на кнопку «Создать» в правом верхнем углу.

Перед нами появится страница создания пользователя. На этой странице находится 3 поля – логин, пароль и переключатель прав администратора. Логин может быть любым, хоть именем фамилией и отчеством, написанным через пробел, хоть просто набором английских букв. В поле пароль следует прописать временный пароль для пользователя. После авторизации, человек, для которого мы создаем пользователя, сам изменит пароль на удобный для него. Выдавать пользователю права администратора или нет, решает создающий пользователя. По стандарту он выключен и лучше без доверия к человеку не выдавать ему права администратора. Форма создания пользователя изображена на рисунке 5.7.

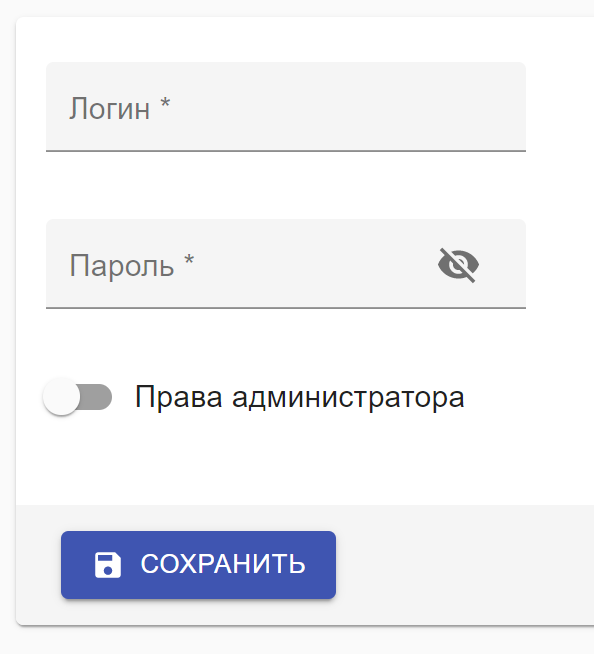


Рисунок 5.7 – Форма создания пользователя

После создания пользователя, он отобразится в списке всех пользователей.

В случае, если пользователь забыл пароль, администратор может его сбросить. Для этого администратору необходимо нажать на кнопку «Редактировать» напротив пользователя, чей пароль необходимо изменить.

Так, администратор попадет на страницу редактирования пользователя. Страница редактирования пользователя представлена на рисунке 5.8.

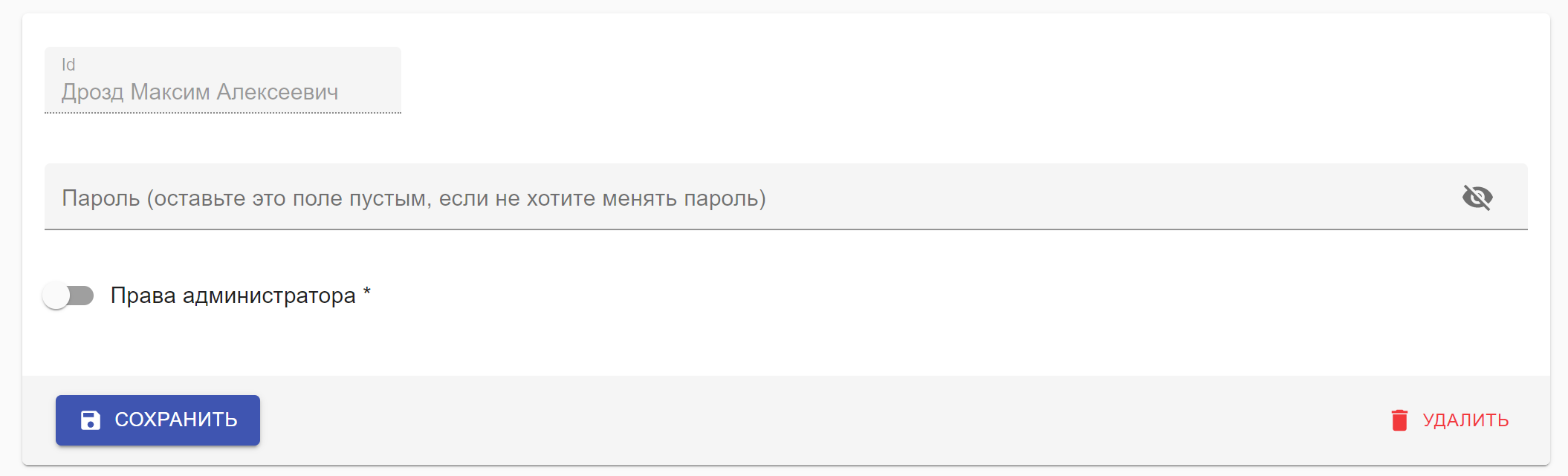


Рисунок 5.8 – Форма редактирования пользователя.

Для сброса пароля, необходимо ввести новый временный пароль в поле «Пароль», который пользователь потом заменит на свой. Так же на этом экране можно выдавать и забирать права администратора, а также можно удалить пользователя.

Стоит отметить, что невозможно изменить собственный пароль через это меню, нельзя забрать у себя права администратора, а также невозможно удалить свой собственный аккаунт.

Переходим к вещам, которые доступны всем пользователям, а не только администратору.

Если сервер был запущен в первый раз, то после авторизации, пользователя будет встречать сообщение о том, что расписания нет. Так как это только что установленный сервер, его там быть и не должно.

Однако перед тем, чтобы создать расписание сначала необходимо добавить преподавателей, группы, предметы и аудитории.

Без преподавателей, групп, предметов и аудиторий расписание составить просто невозможно, так как расписание состоит из всех этих компонентов в обязательном порядке.

Для начала заполнения всех необходимых данных, необходимо выбрать тип данных, которые мы хотим заполнить. Начнем с групп. Окно с пустым списком групп представлено на рисунке 5.9.

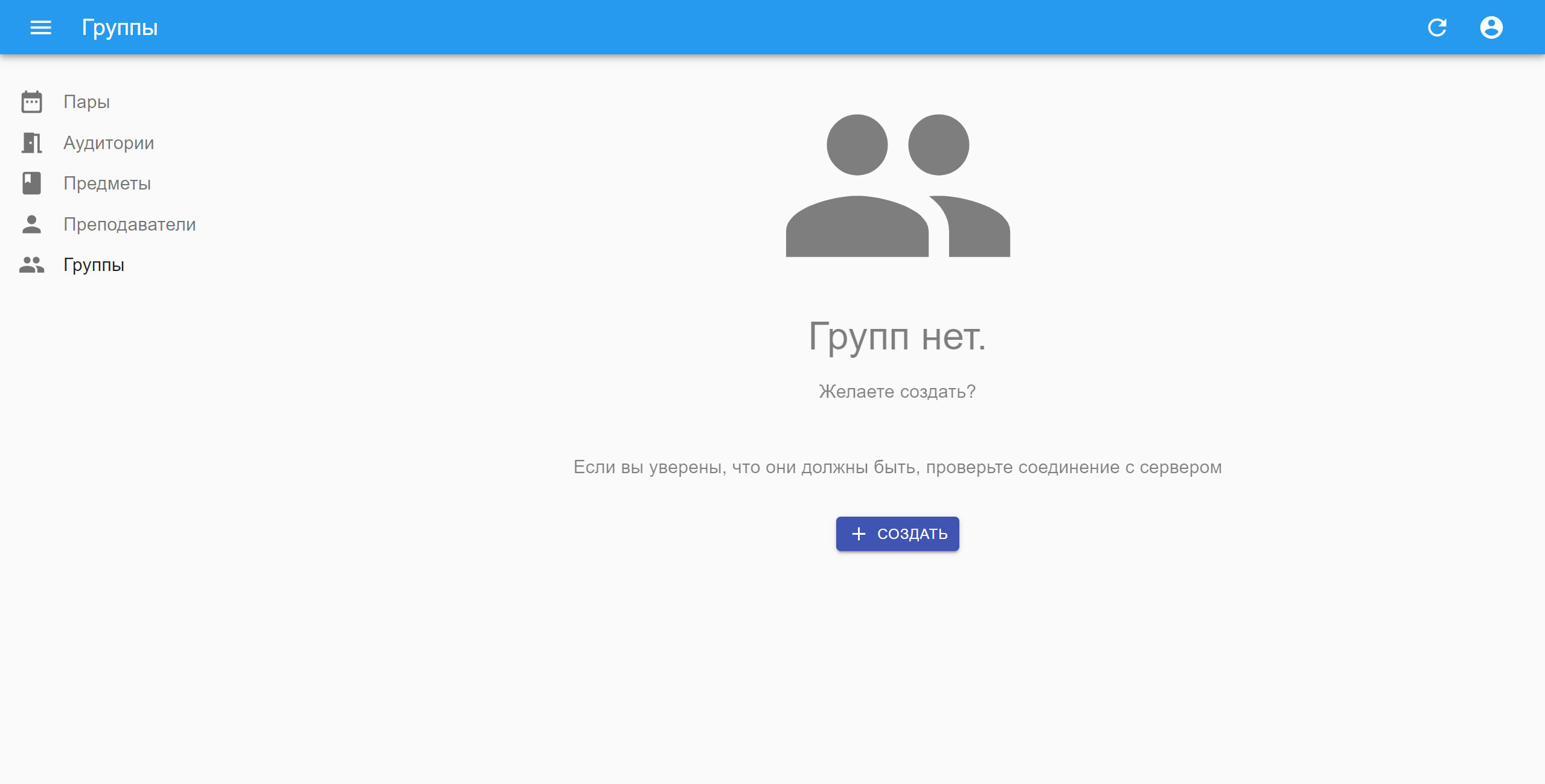


Рисунок 5.9 – Страница групп с пустым списком

Для добавления группы, необходимо нажать на кнопку «Создать». Тогда мы перейдем на страницу добавления групп. Эта страница представлена на рисунке 5.10.

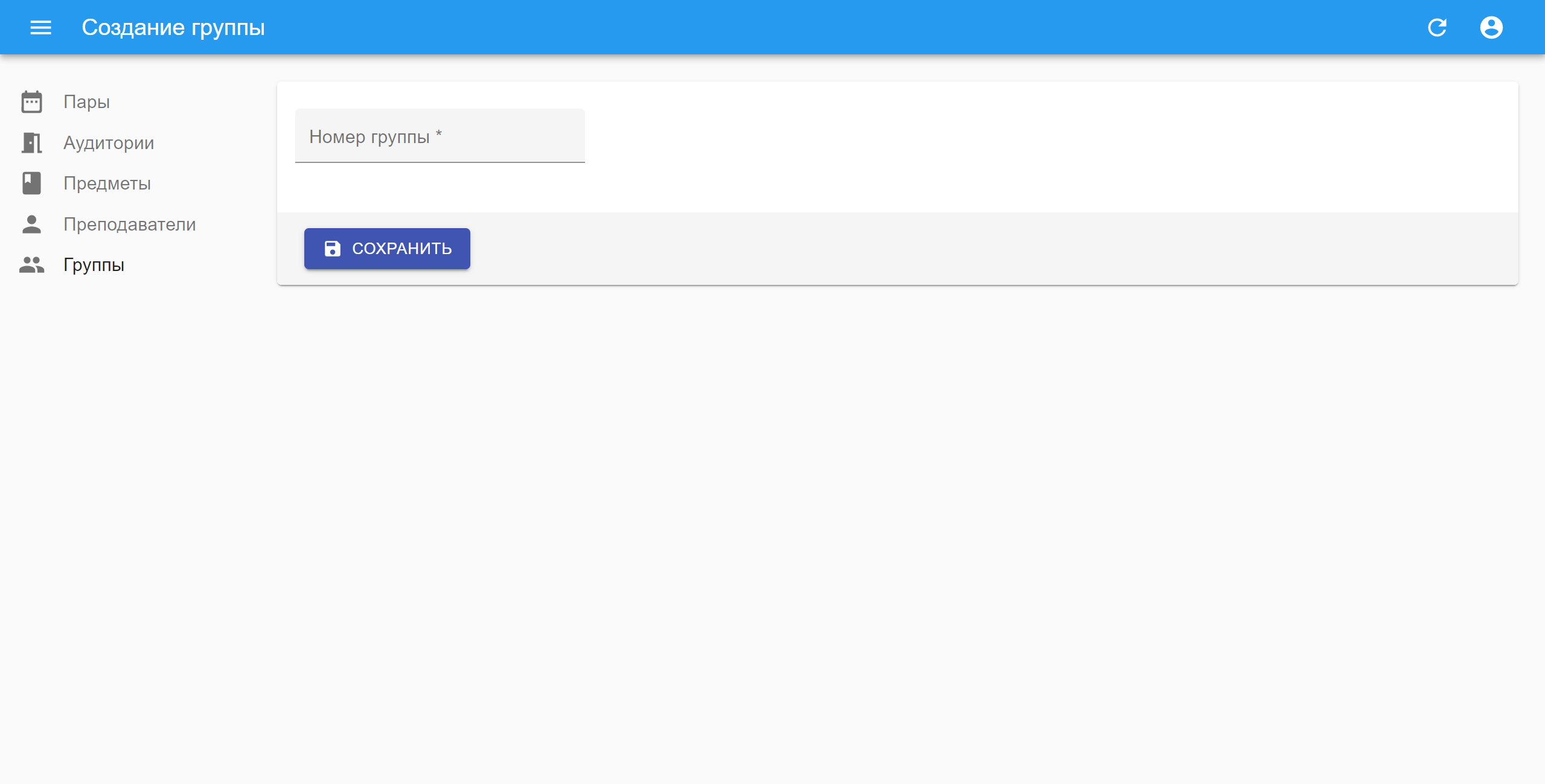


Рисунок 5.10 – Страница добавления групп

В данном окне нам необходимо только ввести номер группы и жать на кнопку «Сохранить». Нас перебросит на страницу просмотра группы, там мы сможем отредактировать нашу группу. Окно просмотра группы представлено на рисунке 5.11.

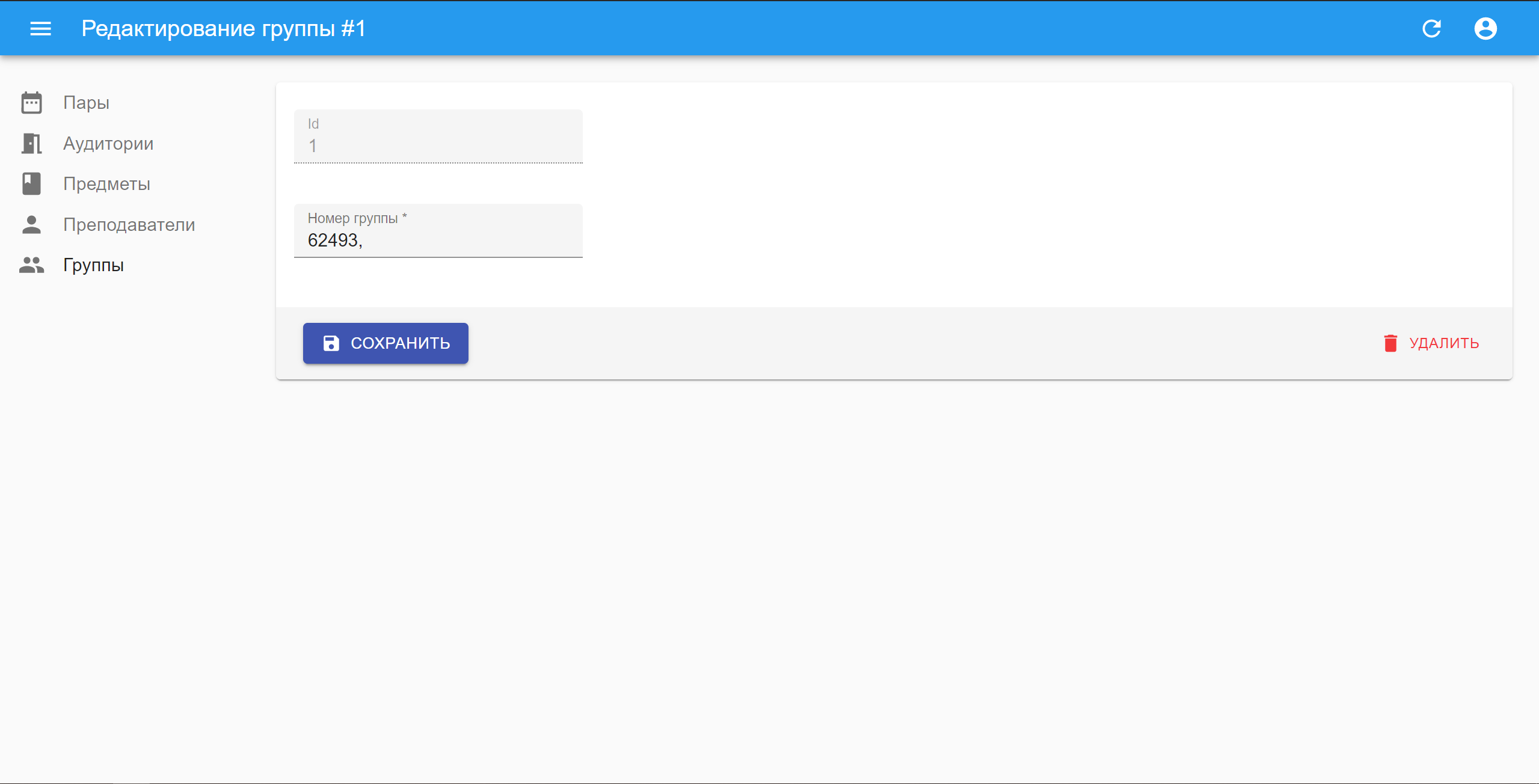


Рисунок 5.11 – Страница редактирования группы

Как мы можем заметить, в номере группы была совершена опечатка. Для исправления этой опечатки, необходимо нажать на кнопку «Сохранить».

После добавления нескольких групп, на странице просмотра групп мы увидим их все. Пример приведен на рисунке 5.12.

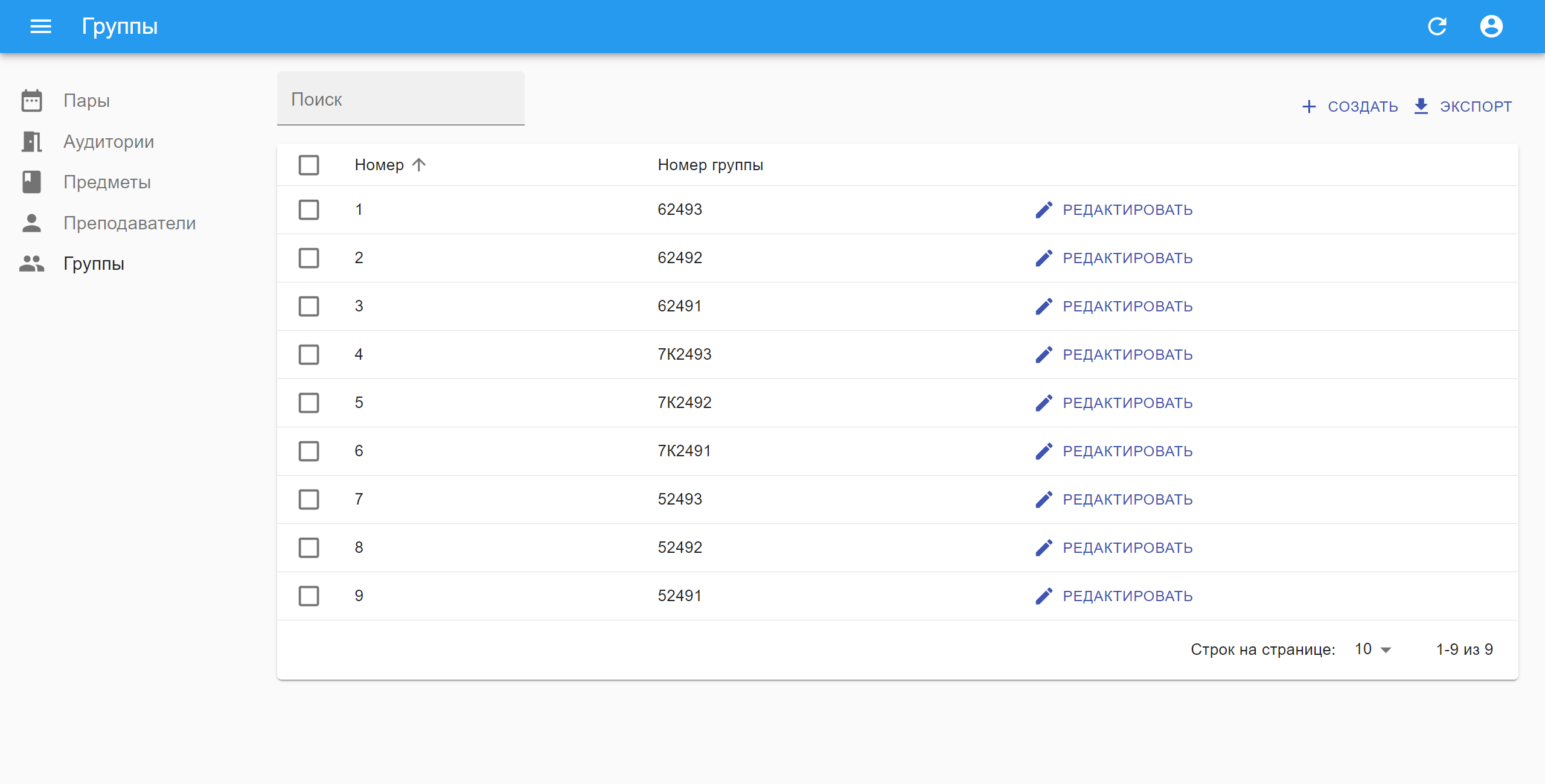


Рисунок 5.12 – Пример заполнения страницы с группами

Далее заполним преподавателей. Сейчас их нет, поэтому зайдем на страницу преподавателей и нажмем на кнопку «Создать». Перед нами появится окно, идентичное тому, что мы видели при добавлении групп, однако полей будет побольше. Экран добавления преподавателей представлен на рисунке 5.13.

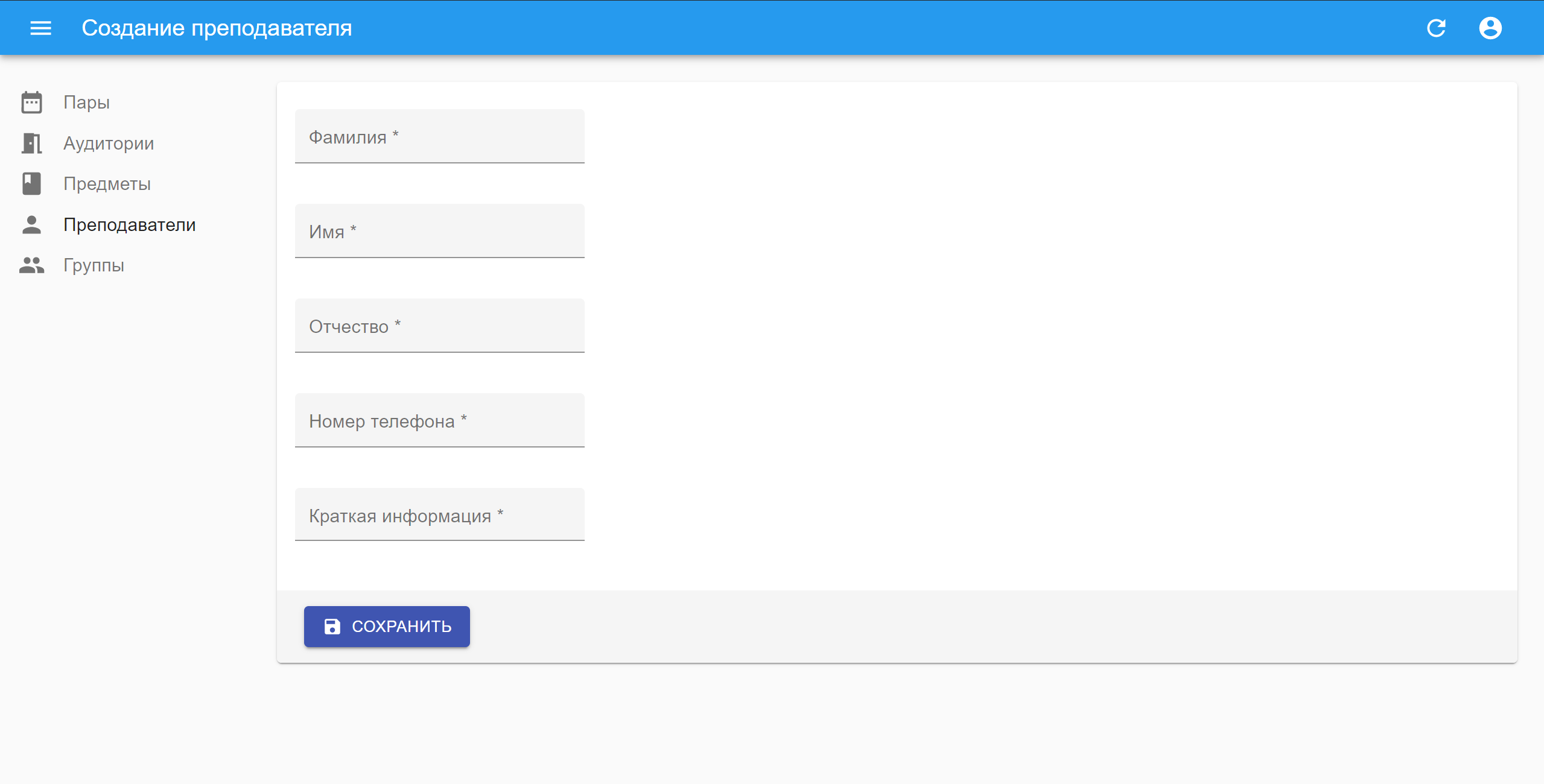


Рисунок 5.13 – Экран добавления преподавателей.

С именем, фамилией, отчеством и номером телефона все и так понятно. Поле краткой информации может содержать в себе краткую биографию преподавателя, которая может быть полезна учащимся, а также дополнительные контакты, такие как Telegram, Viber, VK, WhatsApp, второй номер телефона и другие. Какие именно, решает сам преподаватель. Он может написать сюда все что угодно.

Пример заполнения страницы преподавателей представлен на рисунке 5.14.

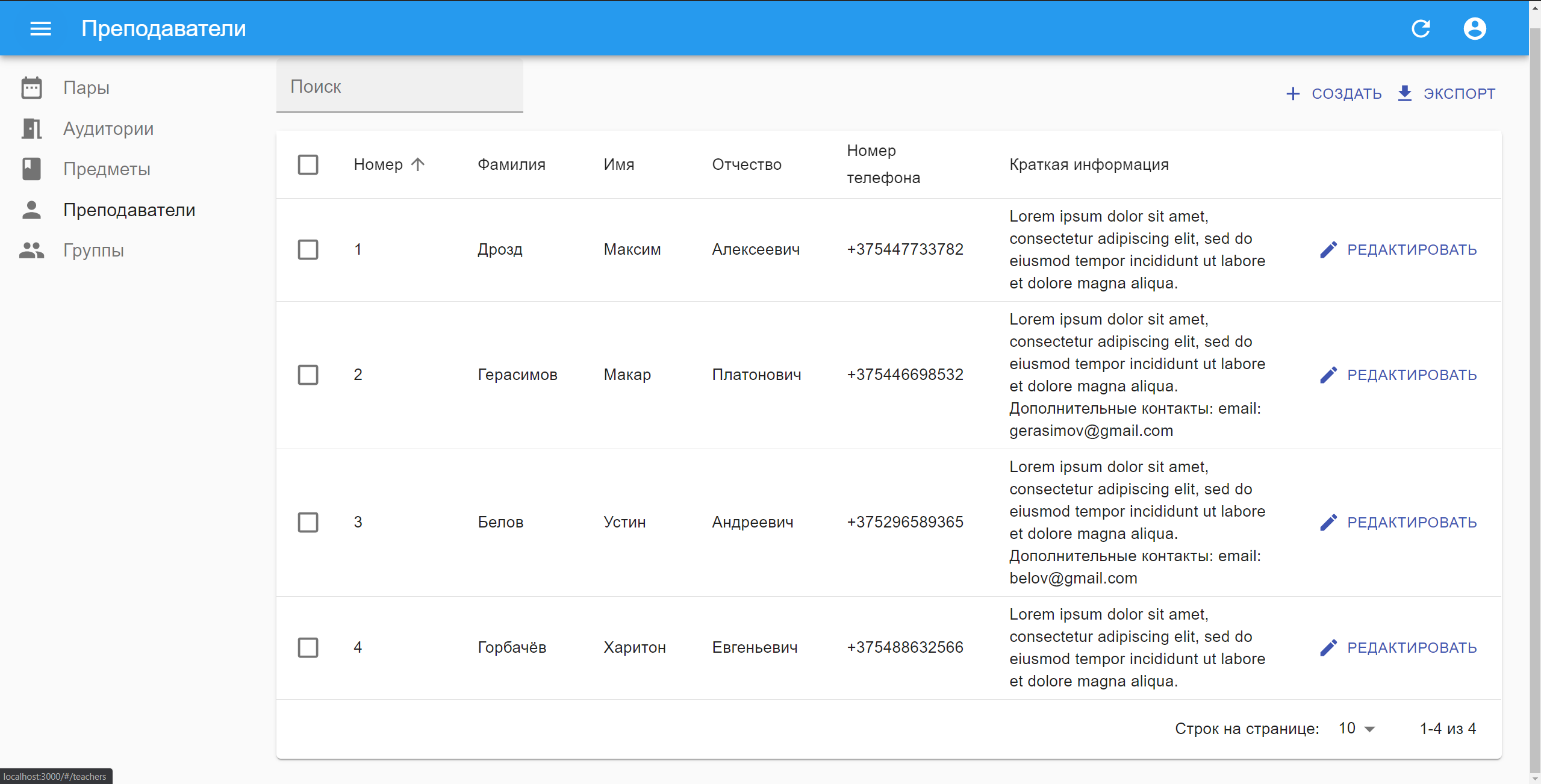


Рисунок 5.14 – Экран с преподавателями.

Преподавателей, так же, как и группы можно редактировать. Для этого необходимо открыть преподавателя, отредактировать его и нажать на кнопку «Сохранить». Экран редактирования преподавателей представлен на рисунке 5.15.

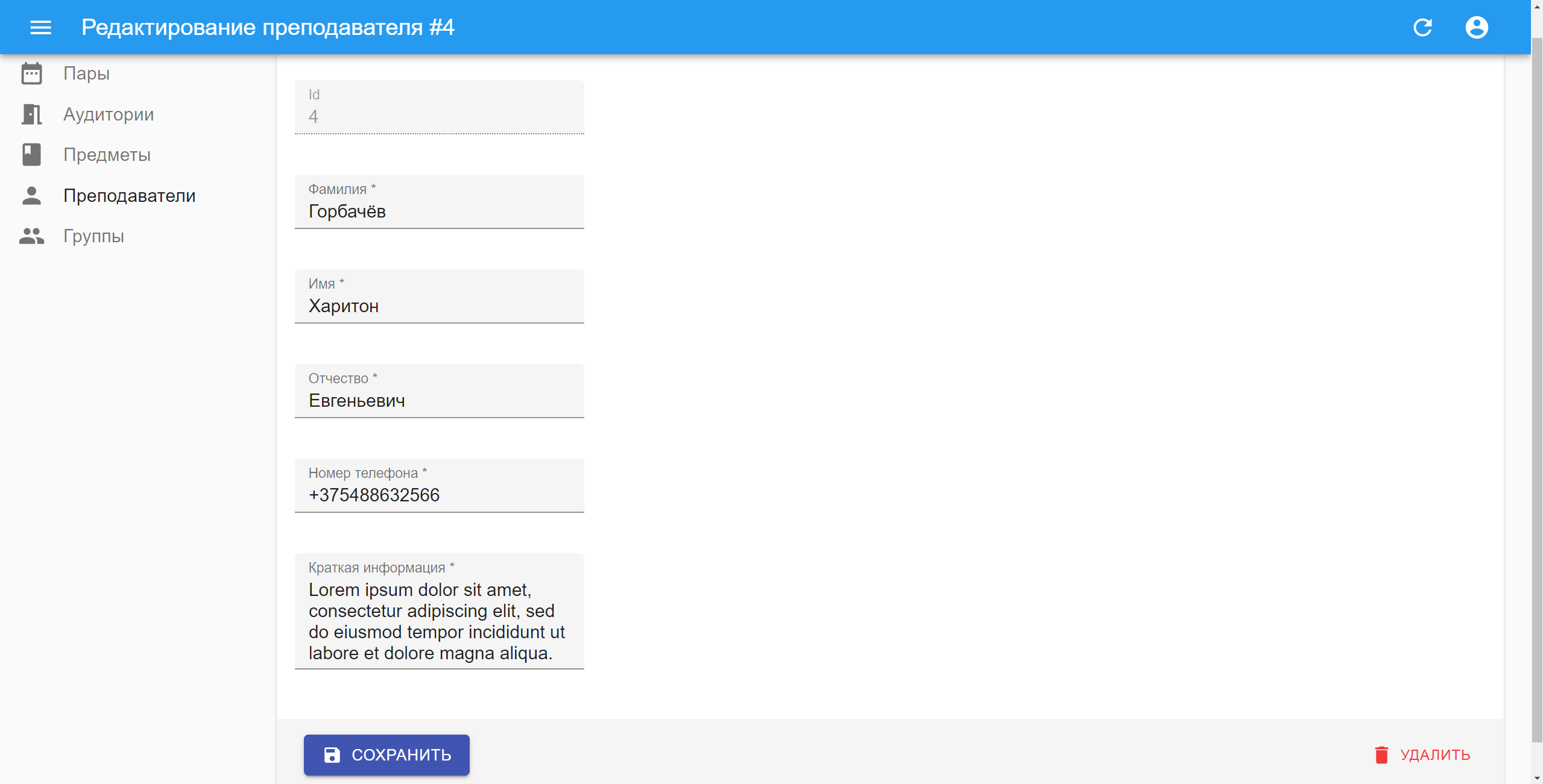


Рисунок 5.15 – Экран редактирования преподавателей

Экран добавления предметов и аудиторий точно такой же, как и групп, однако он имеет два поля – полное и краткое название предмета. Краткое название предмета для отображения расписания в маленьком окне в мобильном приложении. Краткое название предмета не должно быть больше 10 символов. Пользоваться ими можно точно так же, как и экраном добавления групп. Форма добавления предметов представлен на рисунке 5.16.

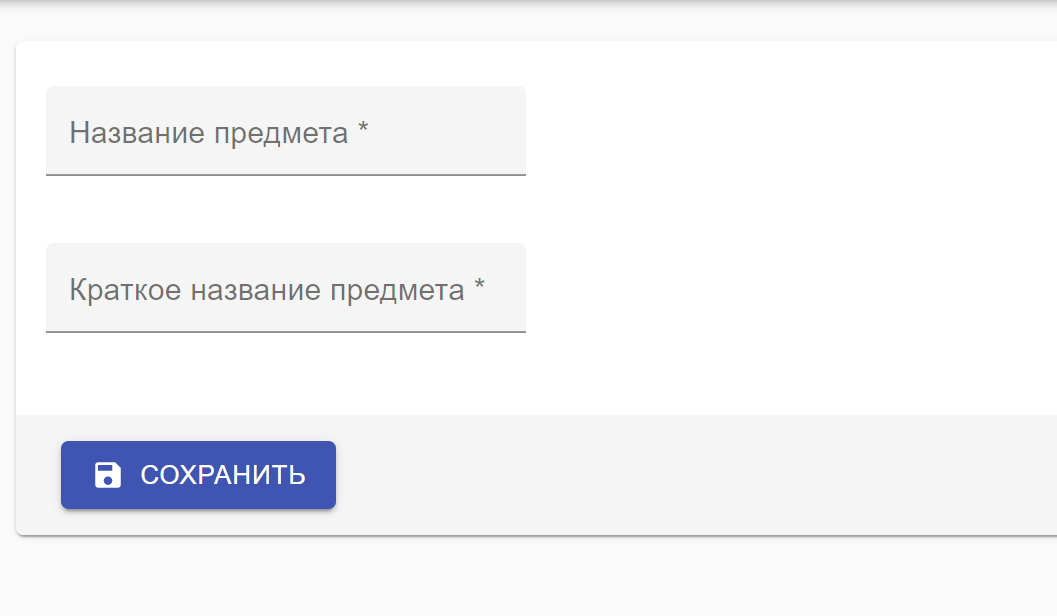


Рисунок 5.16 – Форма добавления предметов

Экран добавления аудиторий представлен на рисунке 5.17.

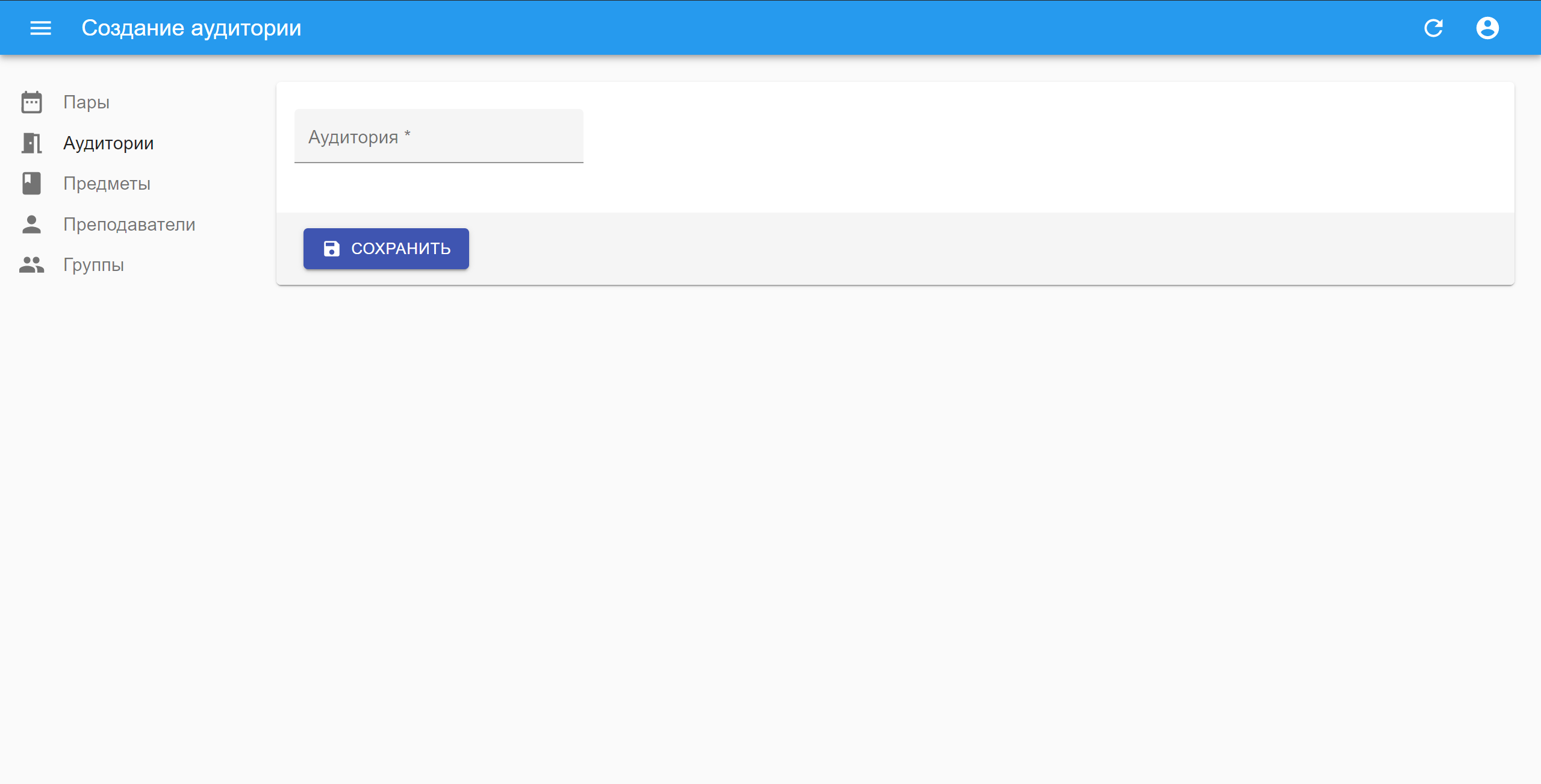


Рисунок 5.17 – Экран добавления аудиторий

Редактировать аудитории и предметы также, как и группы – выбрать аудиторию или предмет, изменить данные и нажать на кнопку «Сохранить».

Пример экрана со списком с добавленными предметами представлен на рисунке 5.18.

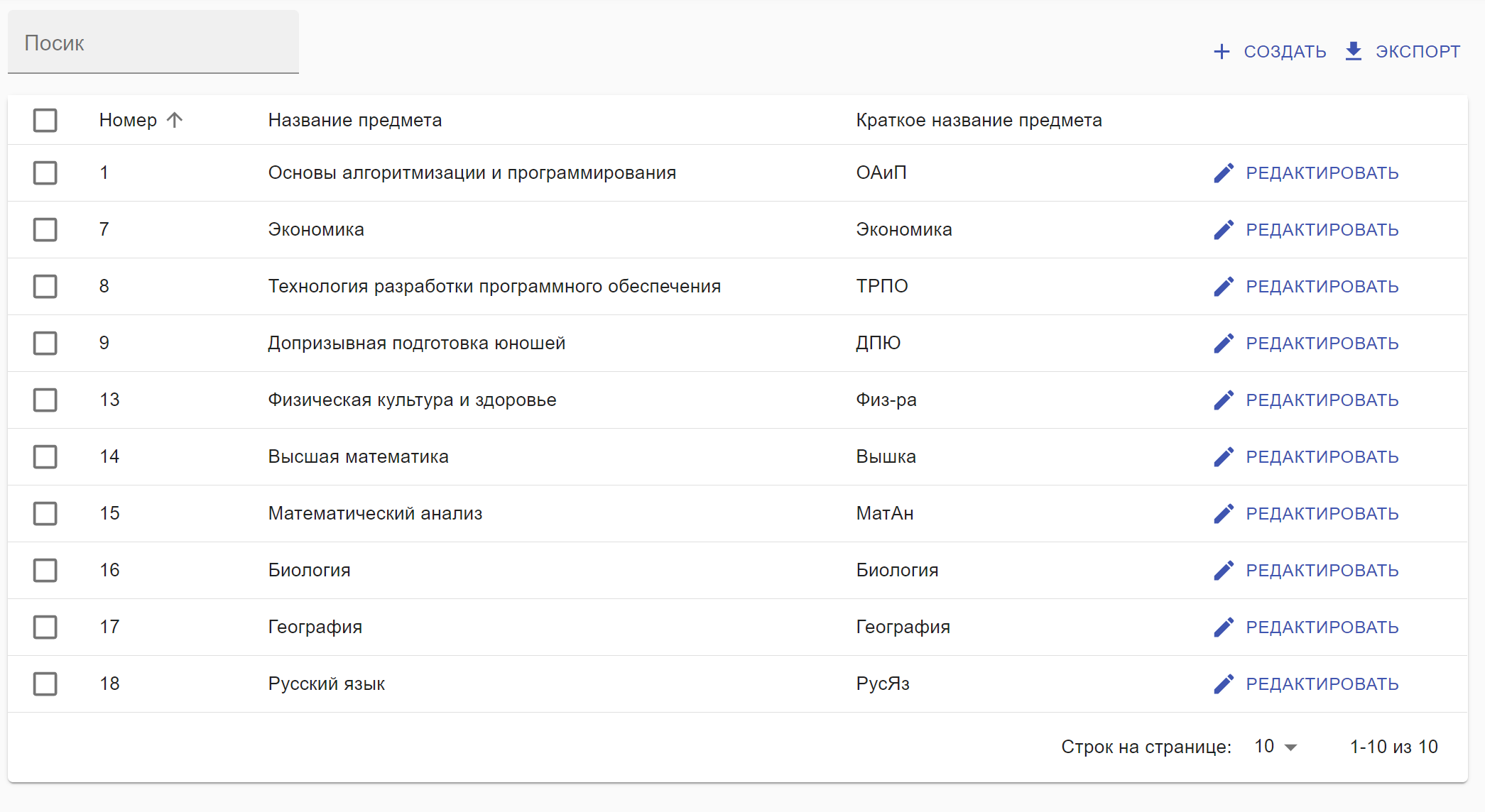


Рисунок 5.18 – Пример заполнения экрана с предметами

Пример списка с добавленными предметами представлен на рисунке 5.19.

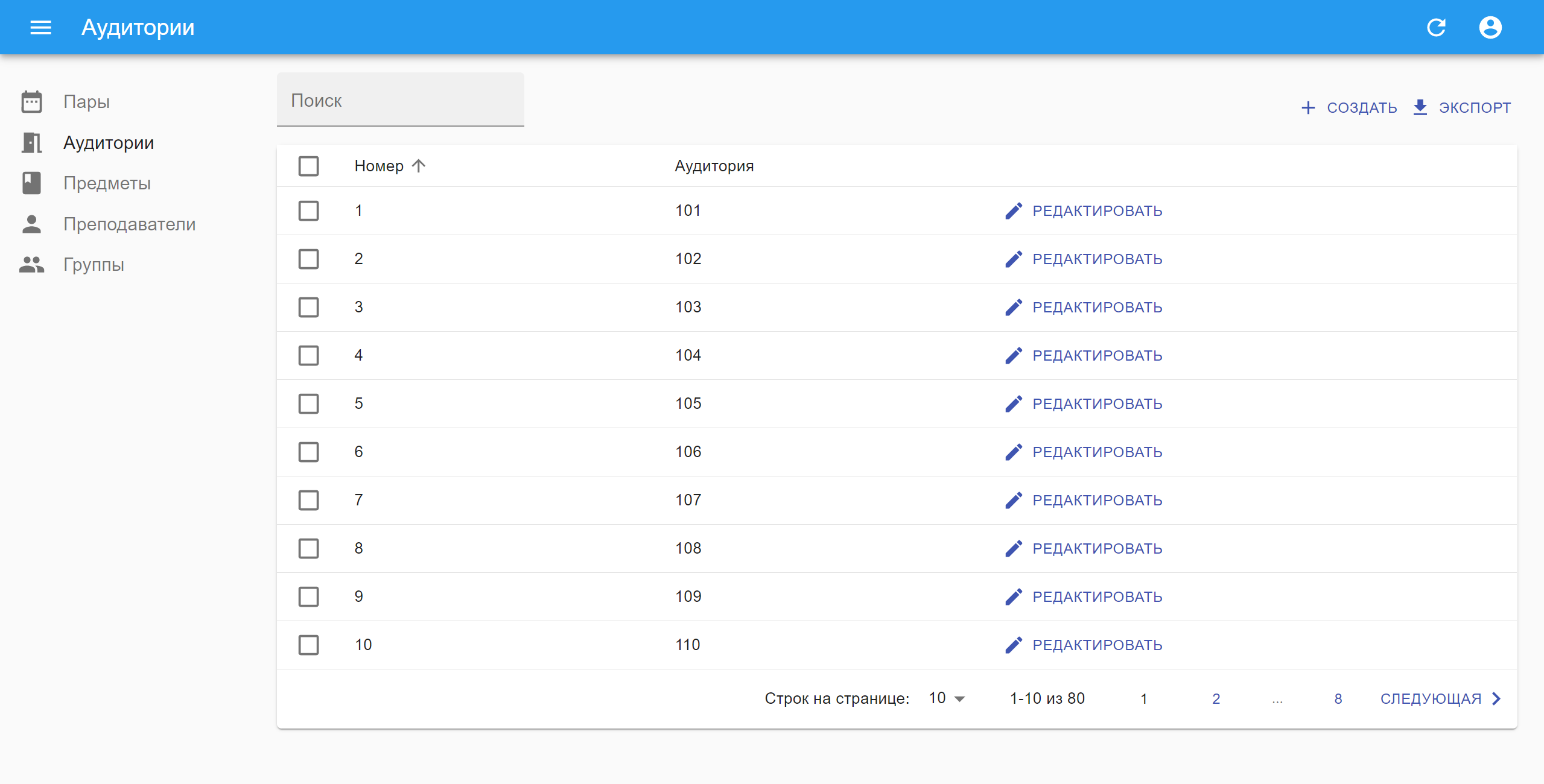


Рисунок 5.19 – Пример заполнения экрана с аудиториями.

Теперь можно начать составлять расписания. Экран с добавлением пары в расписание представлен на рисунке 5.20.

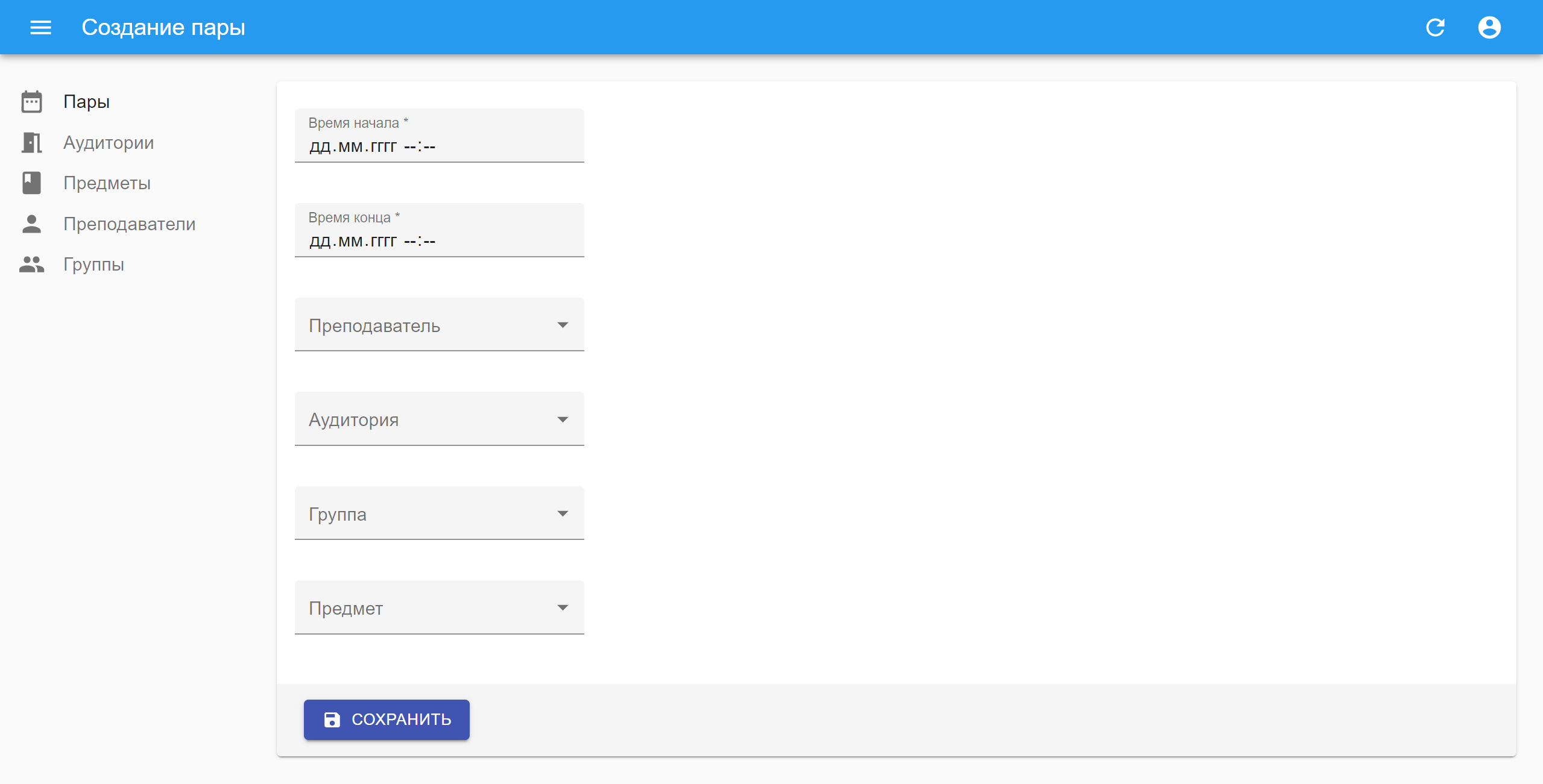


Рисунок 5.20 – Экран с добавлением пар в расписание.

Для добавления пары в расписание требуется указать время ее начала и время ее конца. Ввод производится с клавиатуры или с помощью мыши. Затем следует выбрать преподавателя, аудиторию, группу и предмет. Если нажать на эти поля, то можно заметить, что это не просто поле ввода текста, это поле выбора из уже существующих вариантов. Так, например, если мы нажмем на список преподавателей, то появится список преподавателей, состоящий только из преподавателей, которые мы добавили ранее. Выпадающий список преподавателей представлен на рисунке 5.21.

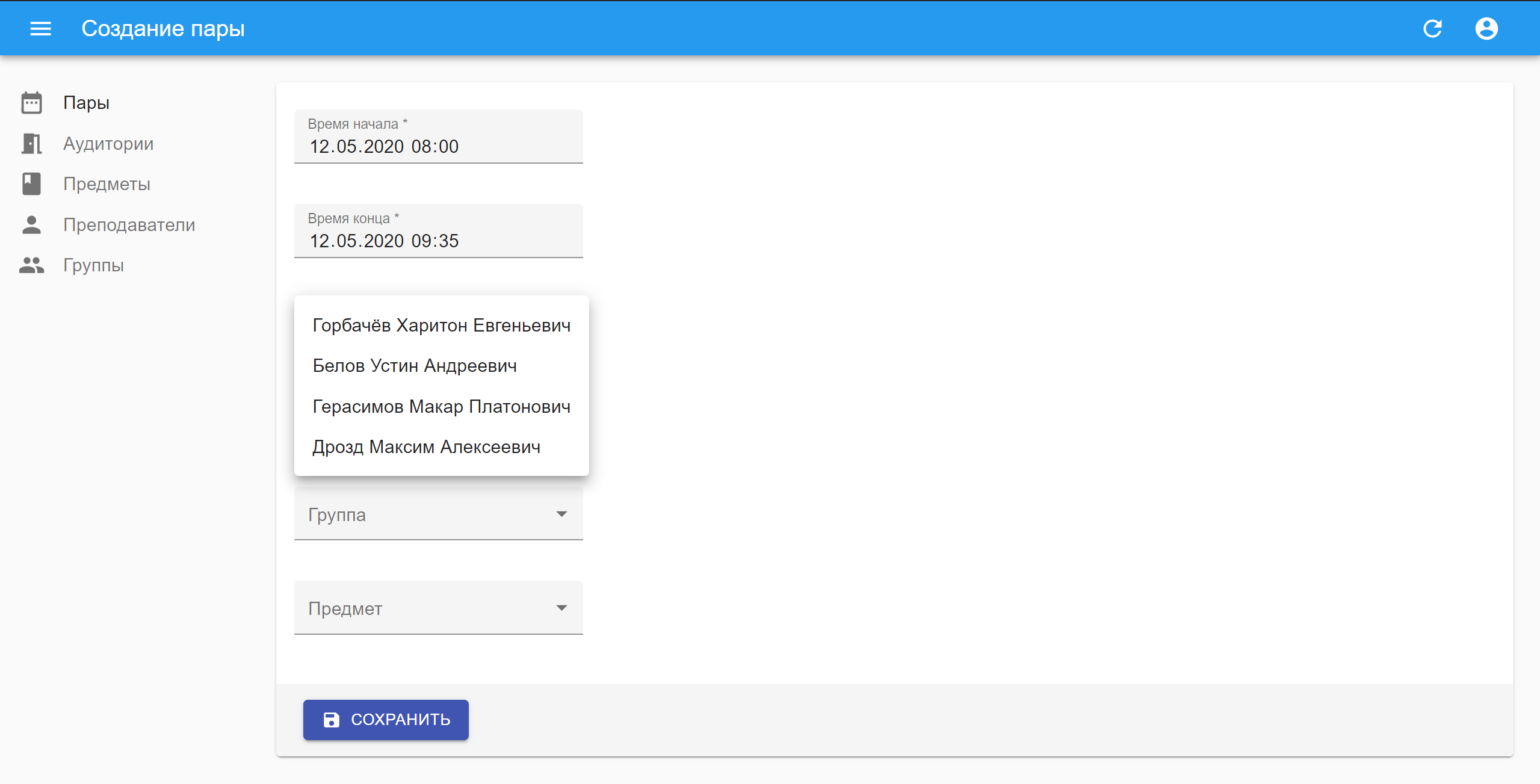


Рисунок 5.21 – Cписок преподавателей на странице с добавлением пар.

Таким образом заполняются поля преподавателя, аудитории, группы и предмета. После их заполнения, можно нажать на кнопку «Сохранить». Далее это расписание можно найти на странице со списком всех пар. На рисунке 5.22 представлен пример заполнения страницы с парами.

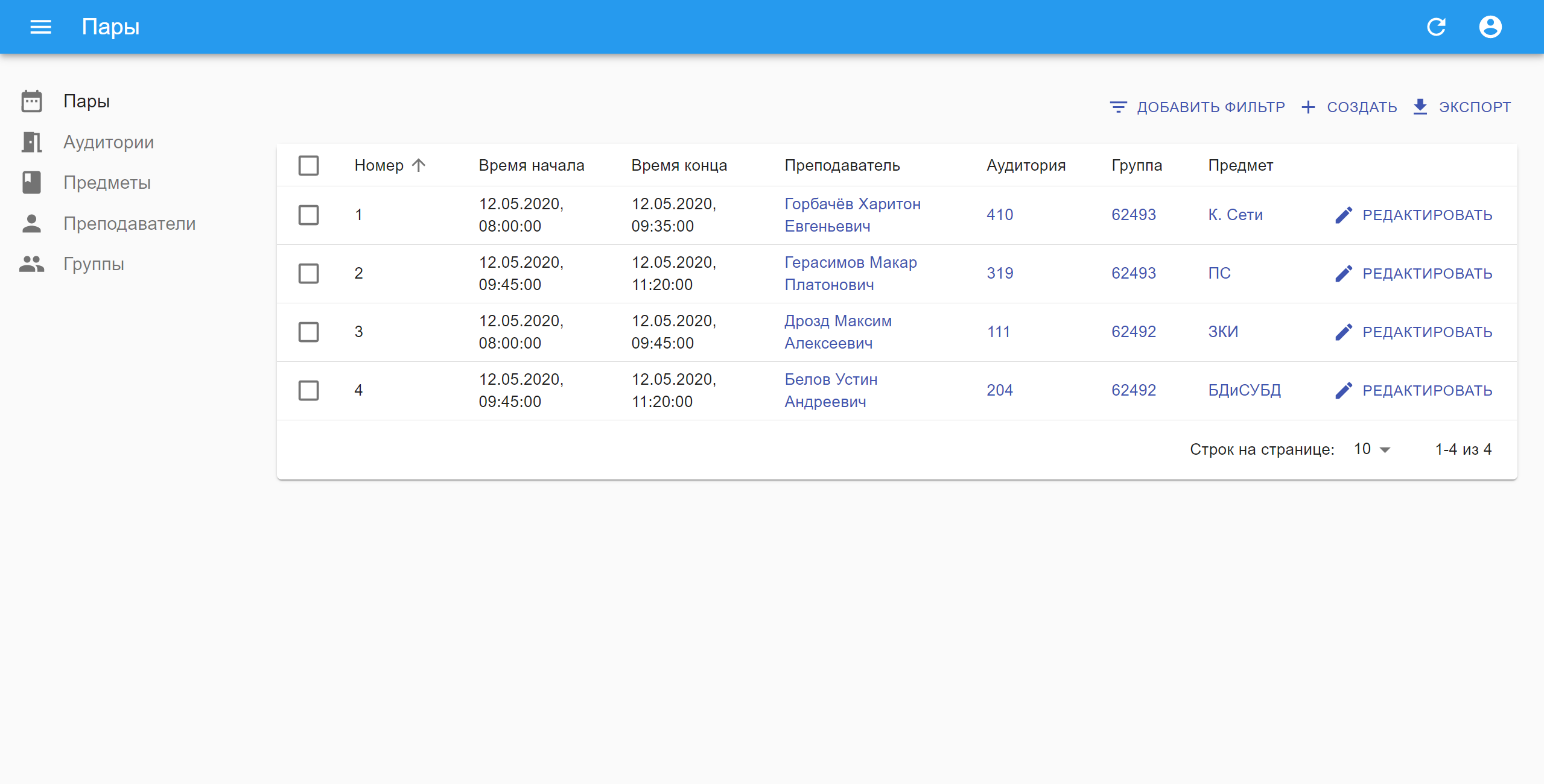


Рисунок 5.22 – Пример заполнения страницы с парами

На каждой странице есть поиск. На станицах аудиторий, предметов, преподавателей и групп он работает по одному принципу – введенный текст вводится в поле ввода и записи, в которых содержится введенный текст, отображаются. На рисунке 5.23 представлен пример поиска по преподавателям.

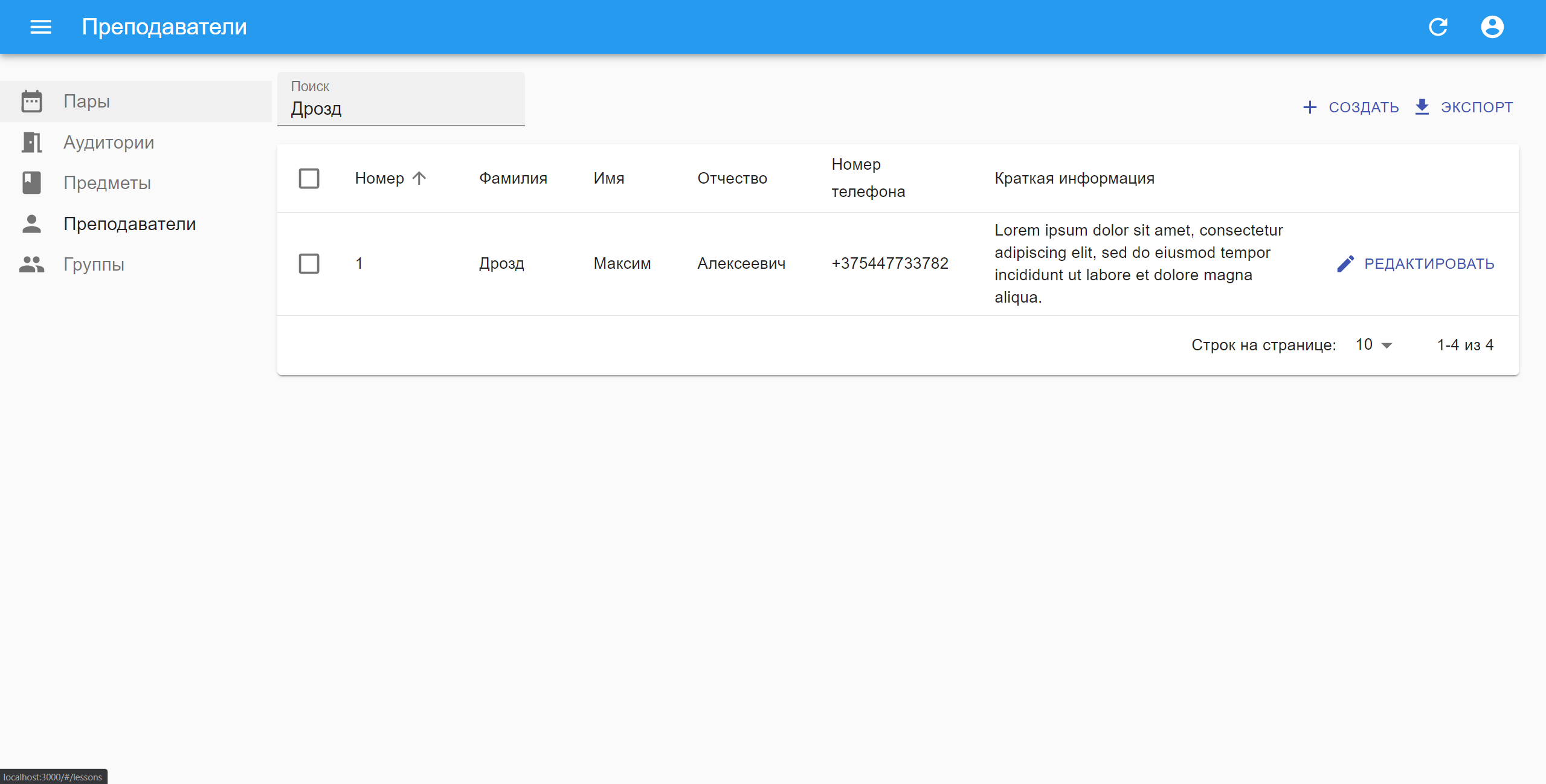


Рисунок 5.23 – Пример поиска по преподавателям

Поиск по парам немного отличится от всех остальных поисков. У поиска есть категории. Можно искать пары с конкретным преподавателем, у конкретной группы и в конкретной аудитории. Для включения поиска необходимо нажать на кнопку «Добавить фильтр». Появится выпадающее окно с категориями фильтрации. Оно представлено на рисунке 5.24.

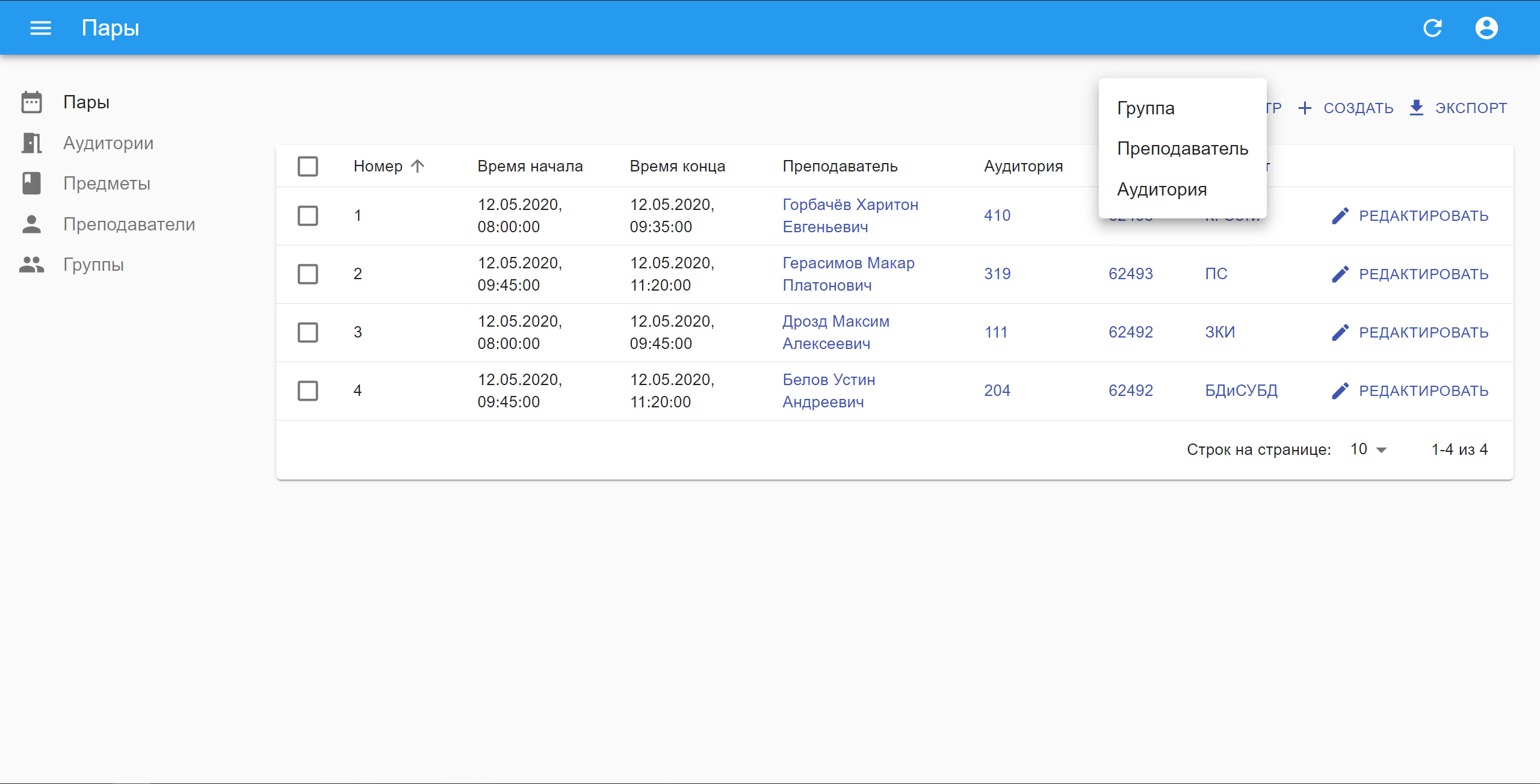


Рисунок 5.24 – Выпадающее окно с категориями фильтрации на экране с парами

При выборе любого варианта, на главном окне появится раздел фильтрации с выбранной категорией. Одновременно можно добавлять несколько фильтров.

Пример с категорией «Преподаватель» показан на рисунке 5.25.

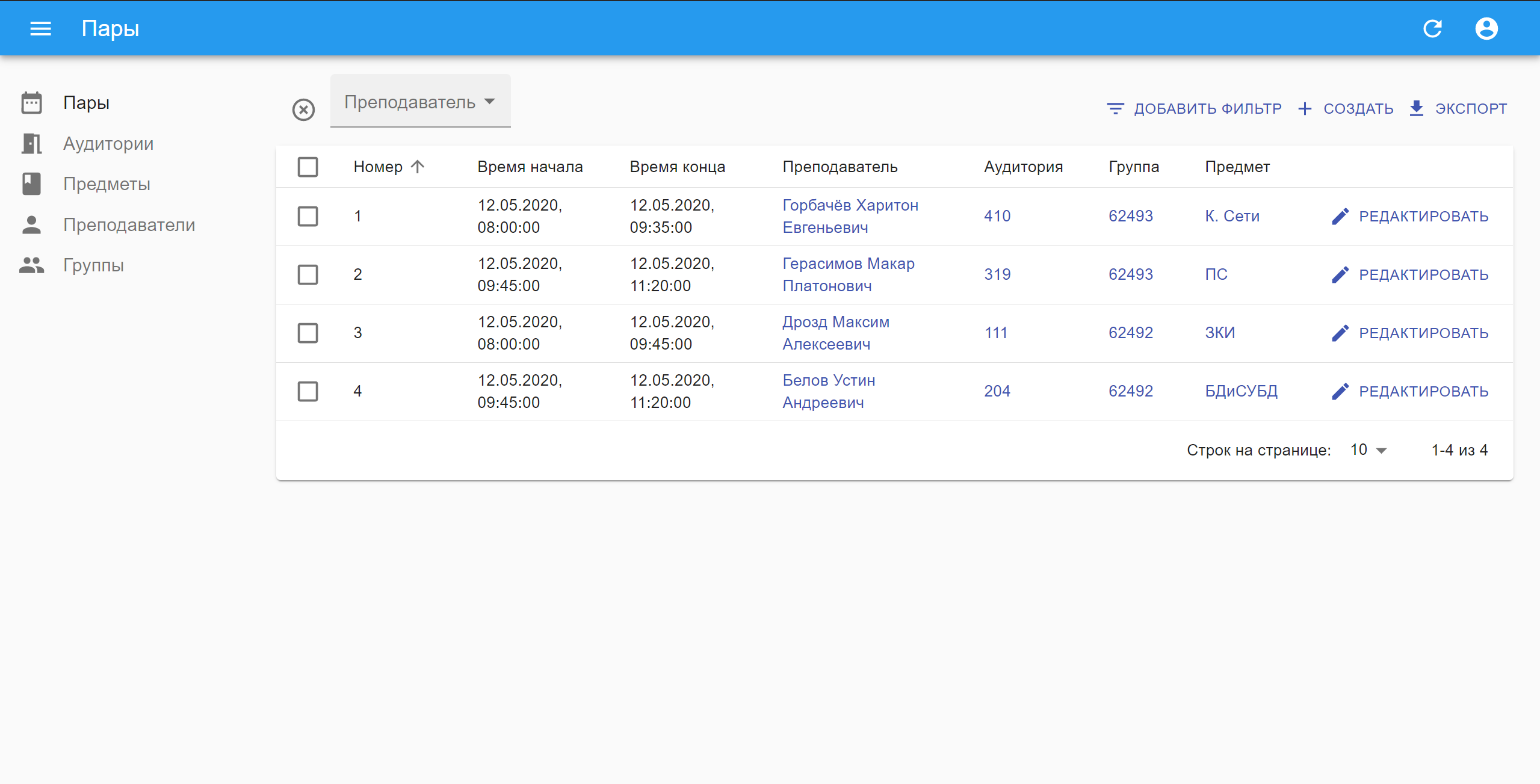


Рисунок 5.25 – Окно с категорией фильтрации «Преподаватель»

После этого можно нажать на этот фильтр и выбрать преподавателя, который нам интересен. Пример фильтрации приведен на рисунке 5.26.

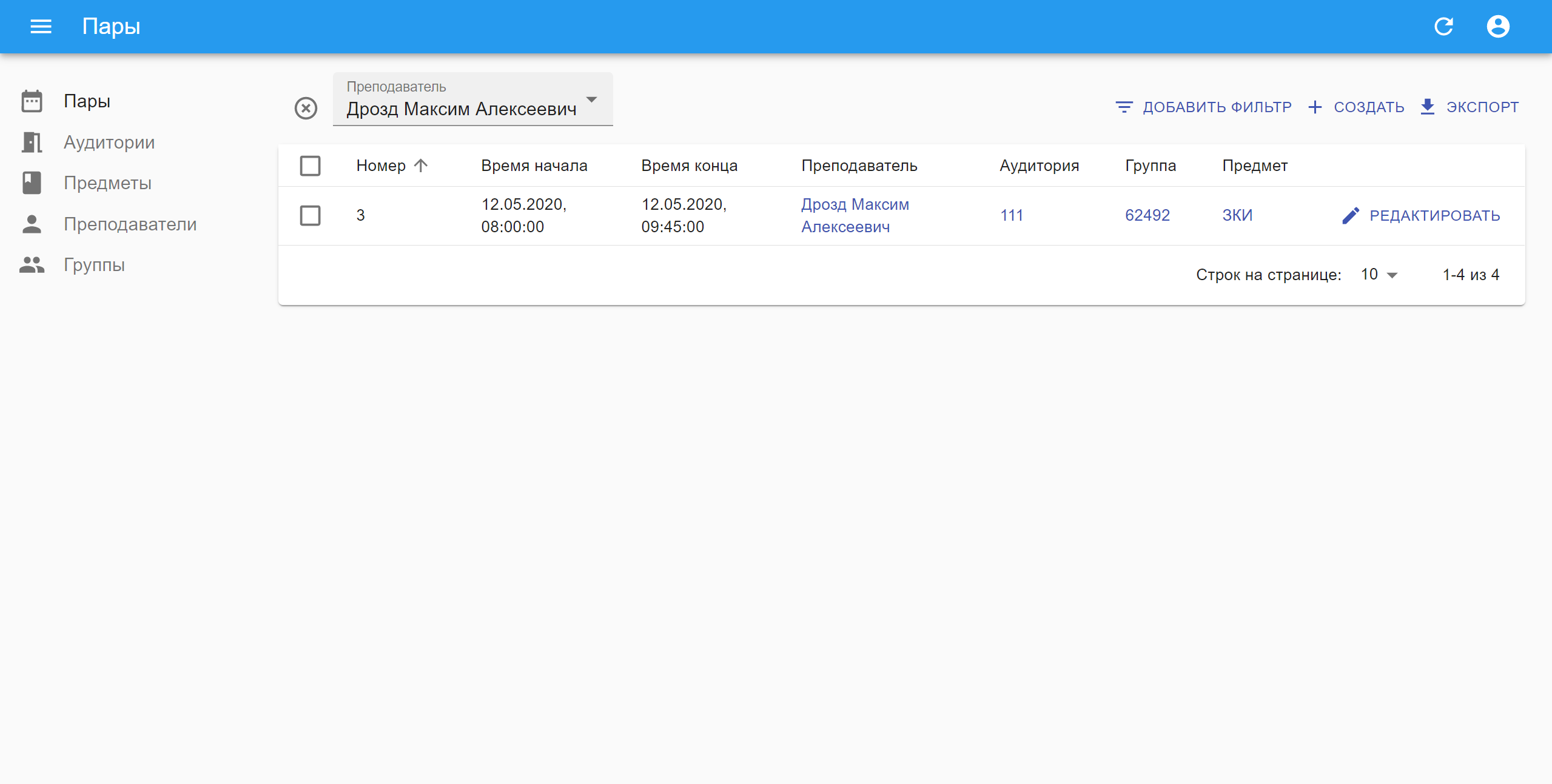


Рисунок 5.26 – Главное окно с примененной фильтрацией по преподавателю

Все записи во всех разделах можно удалить. Для этого необходимо нажать на квадратики напротив записей, которые мы хотим удалить. Пример с выбранными записями к удалению приведен на рисунке 5.27.

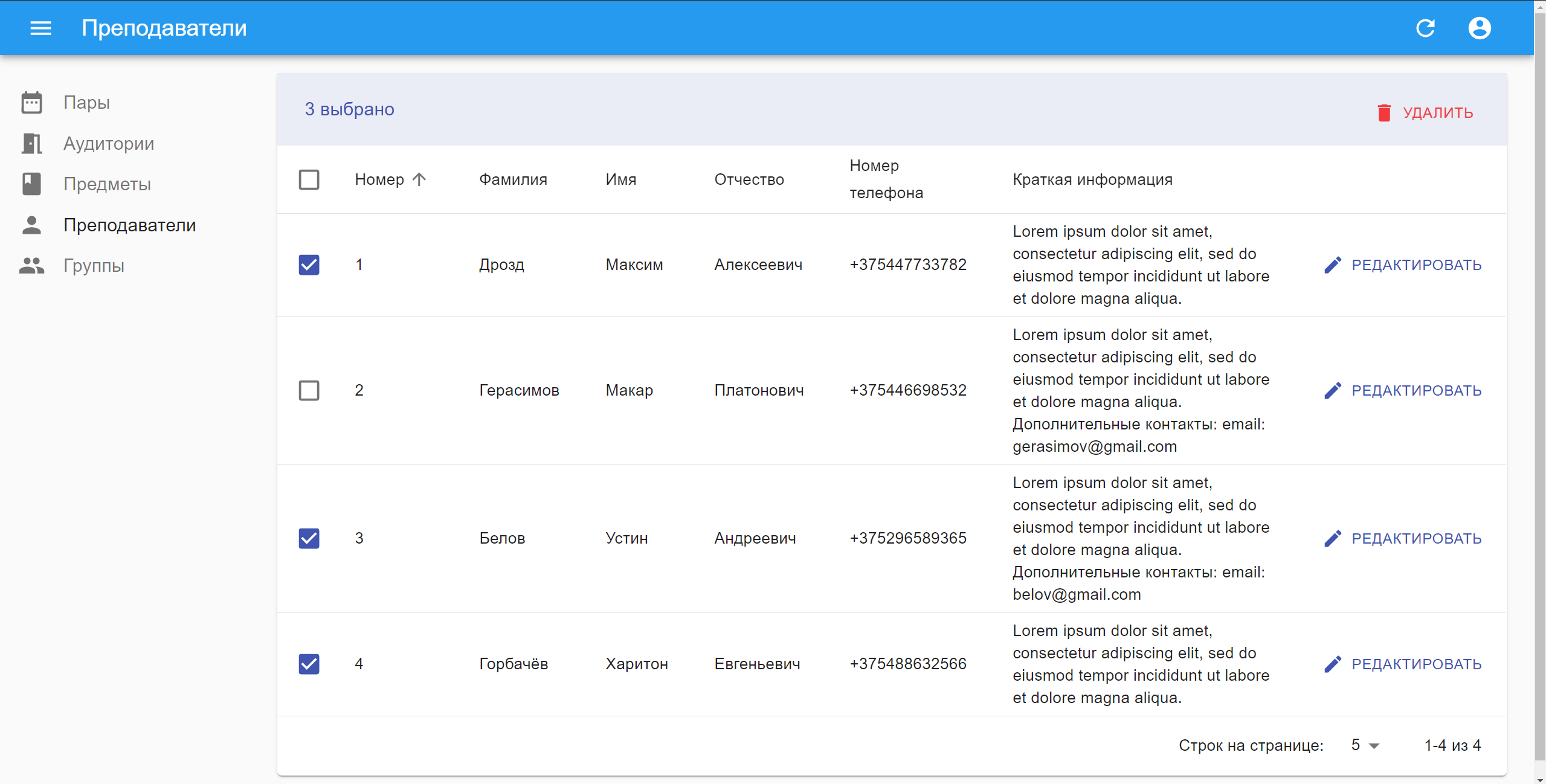


Рисунок 5.27 – Выделенные для удаления записи

Для их удаления осталось только нажать на кнопку «Удалить», однако одна из выбранных записей может быть использована в другой таблице.

Тогда появится сообщение ошибке с описанием проблемы. Сообщение приведено на рисунке 5.28.

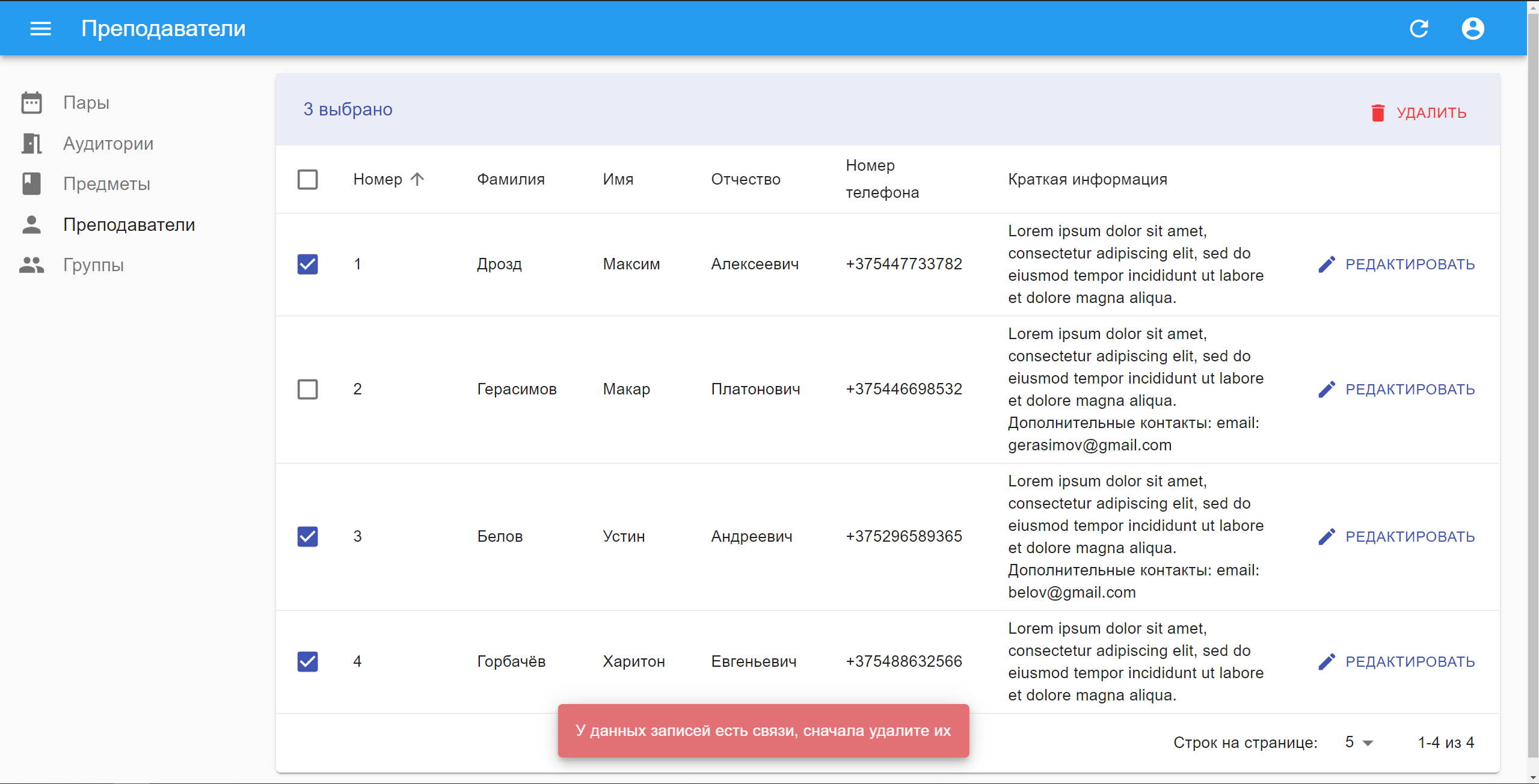


Рисунок 5.28 – Сообщение об ошибке при попытке удаления данных, на которые есть ссылки в других таблицах.

В случае, если ссылок на записи нет, то появится сообщение о том, что все успешно удалено, на котором будет кнопка отмены. То есть если кто-то случайно удалил запись, и он сразу-же это заметил, то он может успеть отменить операцию. Это сообщение представлено на рисунке 5.29.

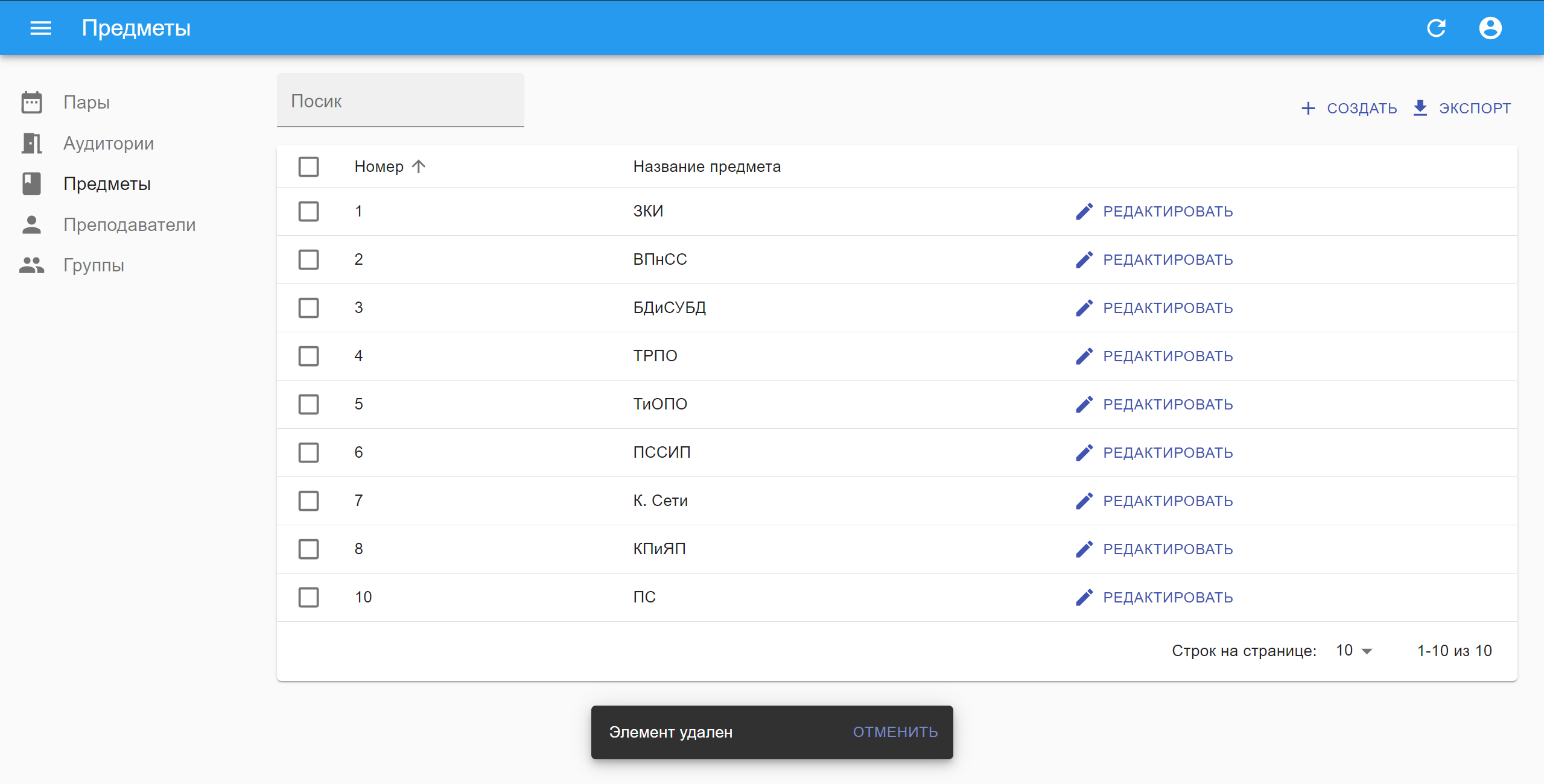


Рисунок 5.29 – Сообщение об успешном удалении записи с возможностью отмены удаления

Помимо этого, кнопка удаления записи есть в окне просмотра любой записи. Кнопку удаления можно заметить на рисунке 5.30.

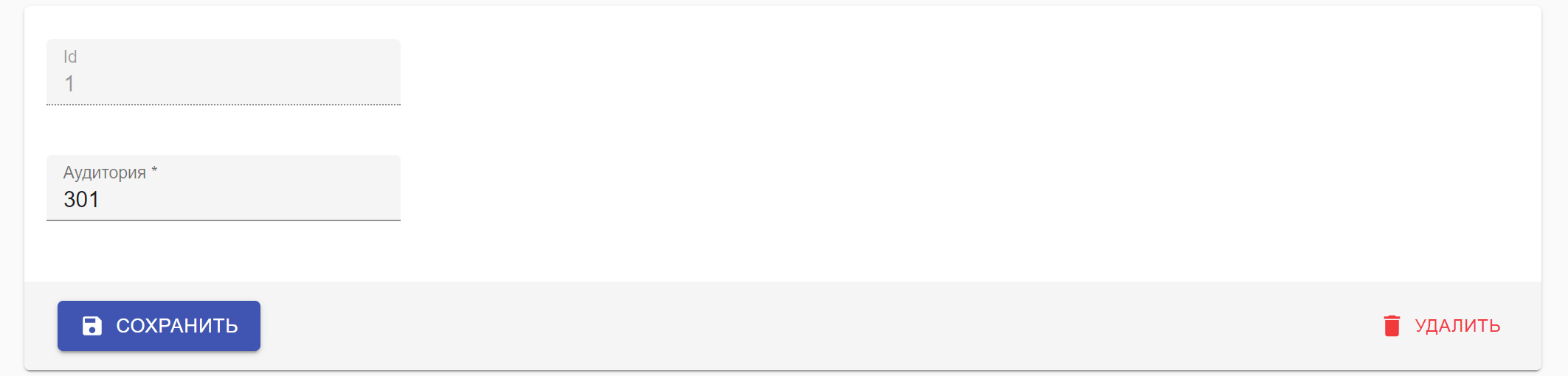


Рисунок 5.30 – Кнопка удаления на экране просмотра записи

Как и все другие кнопки удаления, при нажатии на эту кнопку удаления, будет выполнена проверка на связи, а затем будет произведено удаление с возможностью отмены. Сообщение об ошибке при удалении предмета из экрана редактирования показано на рисунке 5.31.

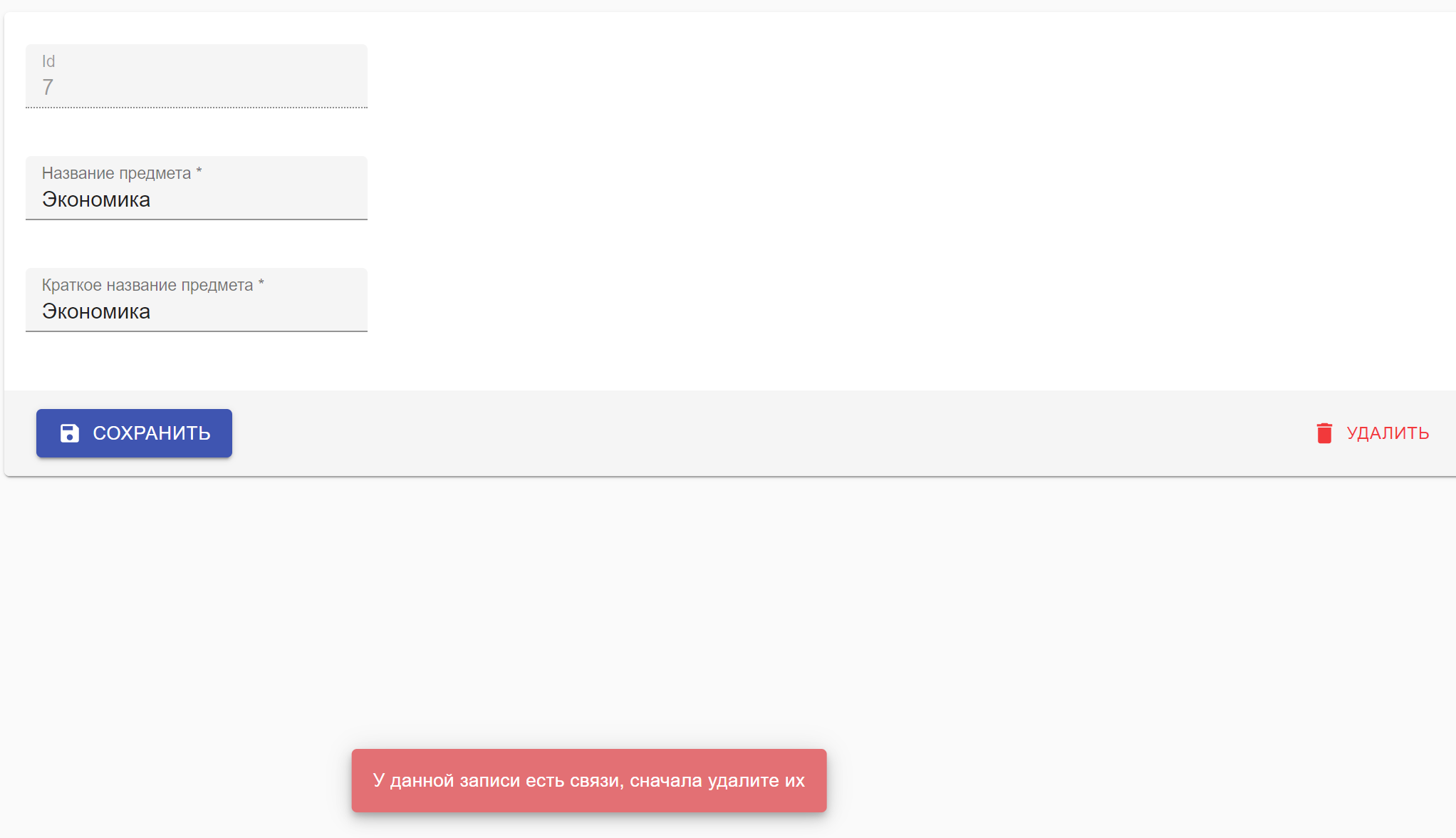


Рисунок 5.31 – Сообщение об ошибке удаления предмета на экране редактирования

Таким образом был рассмотрен весь функционал админ-панели. Ее возможности могут позволить полноценно вести расписание учебного заведения.

Теперь перейдем к инструкции по использованию Android-приложения.

При первом запуске приложения необходимо ввести номер группы в окно, представленное на рисунке 5.32. Это необходимо для того, чтобы знать к какой группе принадлежит пользователь и показывать ему расписание только его группы. Вы потом сможете изменить номер группы в настройках в любой момент.

По ходу ввода номера группы выпадет подсказка с номерами групп, доступными на данный момент. Эти номера групп берутся с сервера. То есть если ваша группа не появилась в подсказке, значит для вашей группы пока что еще не добавили расписание.



Рисунок 5.32 – Окно ввода номера группы

После ввода номера группы откроется окно с расписанием на ближайший день, или сегодня, если расписание на сегодня доступно. Само расписание состоит из пар. Каждая пара имеет время начала и время конца, аудиторию, название предмета и преподавателя. На этом экране используется сокращенное название предмета, чтобы он вместился. Помимо этого, используются сокращенные имена и отчества преподавателей.

Так же есть возможность просмотреть расписание на вчера, или позавчера, если вчера было воскресенье. В верху экрана отображается, на какой день вы сейчас смотрите расписание.

Свайпами влево-вправо можно переключаться межу расписаниями на разные дни. Окно представлено на рисунке 5.33.

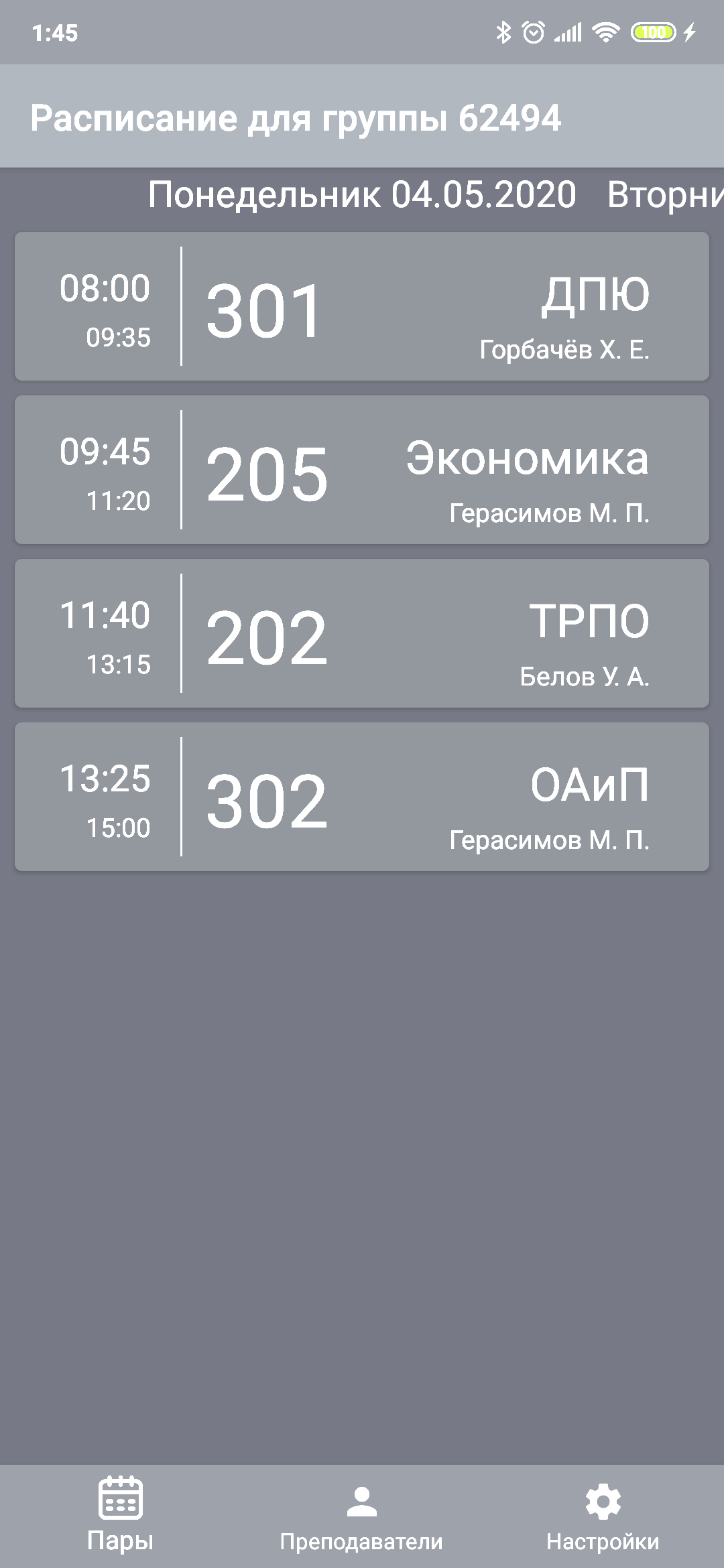


Рисунок 5.33 – Окно с расписанием

Если надо узнать полное название предмета или полное ФИО преподавателя, необходимо лишь нажать на интересующую Вас пару.

После этого откроется диалог, в котором можно посмотреть полное название предмета, полное ФИО преподавателя, время начала пары, время конца пары и кабинет в котором эта пара пройдет. Если необходимо узнать дополнительную информацию о преподавателе, можно запомнить его полное имя и фамилию и затем воспользоваться поиском на экране преподавателей. Как пользоваться этим поиском, будет описано позже. Для того чтобы выйти из диалога, достаточно тапнуть вне его содержимого. Диалог представлен на рисунке 5.34.

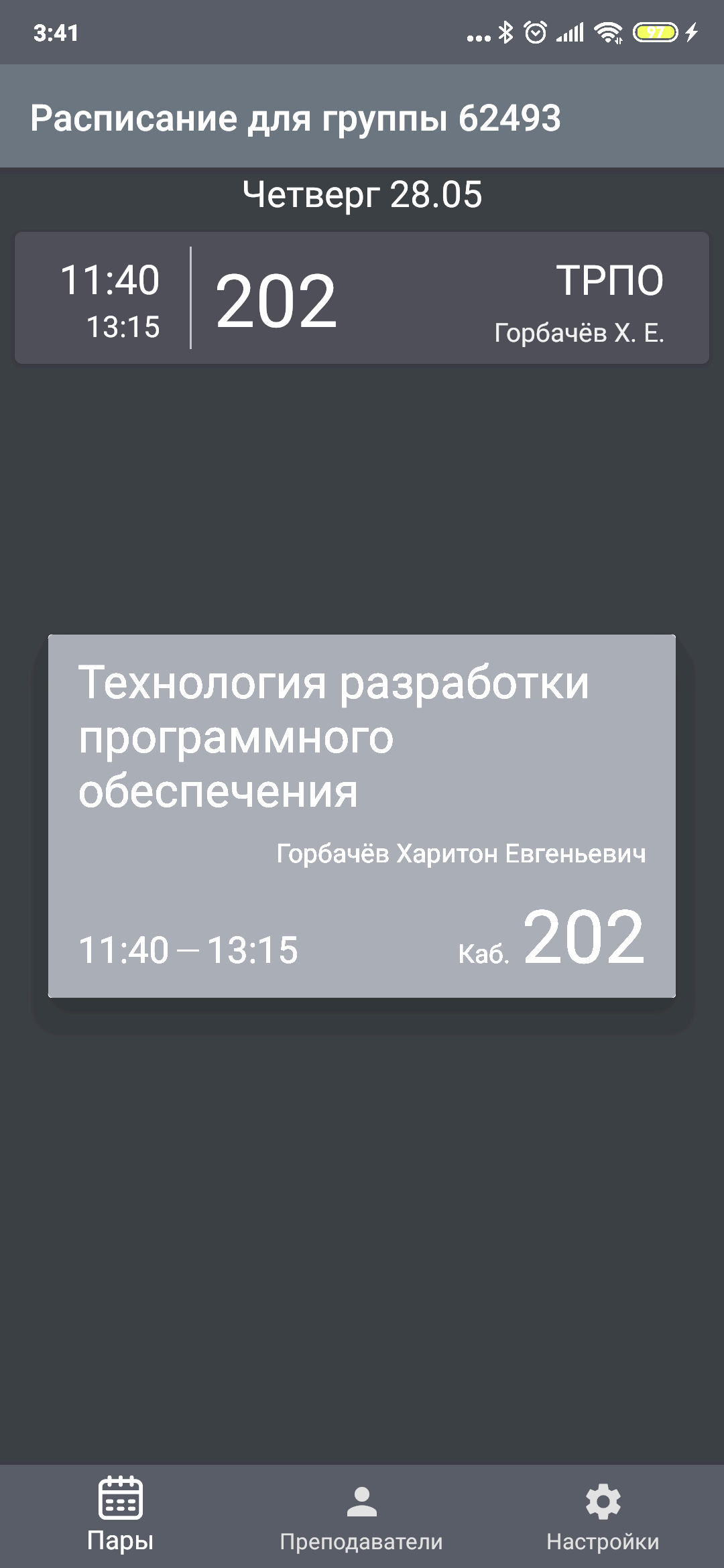


Рисунок 5.34 – Диалог с полными ФИО преподавателя и названием предмета.

В случае если пользователь ошибся при вводе номера группы, или если он захочет поменять ее, то он может это сделать нажатием по кнопке настроек. После нажатия на кнопку, выедет окно, в котором можно посмотреть текущий номер группы и нажать на кнопку изменения номера группы.

Настройки хранятся на специальной области памяти, которой Android сам выделяет место, поэтому после переустановки приложения, или после очистки файлов приложения настройки не сохранятся, и их придется вводить заново при первом открытии приложения.

Экран с окном настроек представлен на рисунке 5.35.

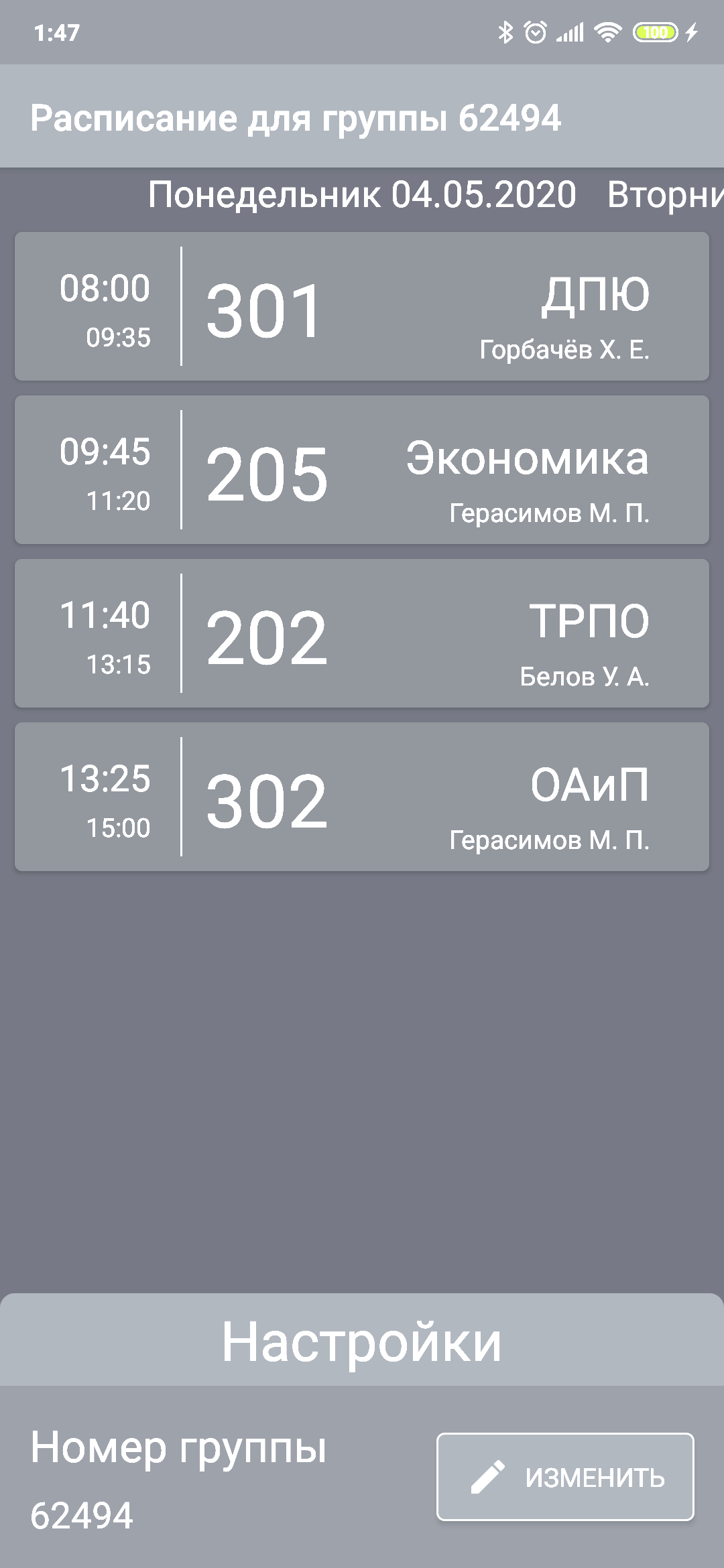


Рисунок 5.35 – Экран с окном настроек

При нажатии на кнопку редактирования появляется окно ввода номера группы. При заполнении нового номера группы будут появляться подсказки возможных номеров групп. Они формируются из списка всех групп, аналогично тому, как формируются подсказки при вводе номера группы в самый первый раз.

Принципиальное отличие этих двух диалогов заключается в том, что диалог, который пользователь видит в первый раз невозможно отменить. Пользователь просто обязан ввести номер группы, и, если он введет придуманный номер группы, он всегда его может изменить в настройках.

Окно редактирования с выпадающими подсказами представлено на рисунке 5.36.

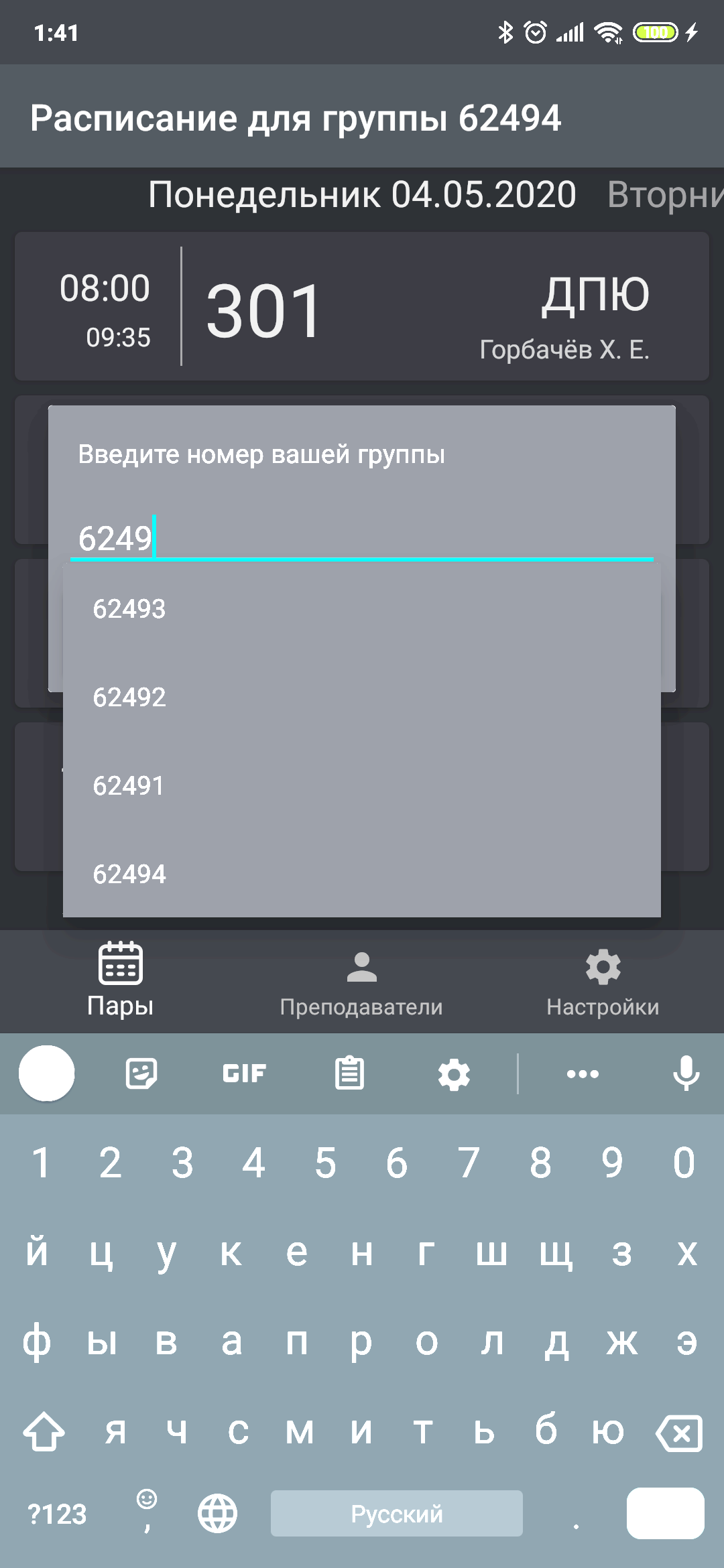


Рисунок 5.36 – Окно ввода номера группы с подсказками

После ввода нового номера группы, расписание обновится и будет актуальным на сегодняшний день для текущей группы.

В процессе обучения может понадобится необходимость в личном общении с преподавателями. Для этого необходимо знать их контактные данные. Специально для этого существует вкладка преподавателей. В ней расположен список всех преподавателей, их номер телефона и краткая информация о преподавателе, можно сказать, краткая биография, а также дополнительные средства связи с преподавателем. Это может быть Telegram, VK, Facebook, WhatsApp или второй номер телефона. Преподаватель сам предоставляет информацию для этого раздела в свободной форме и может оставить там контактные данные в любой форме. А может и не оставить.

Список преподавателей изображен на рисунке 5.37.

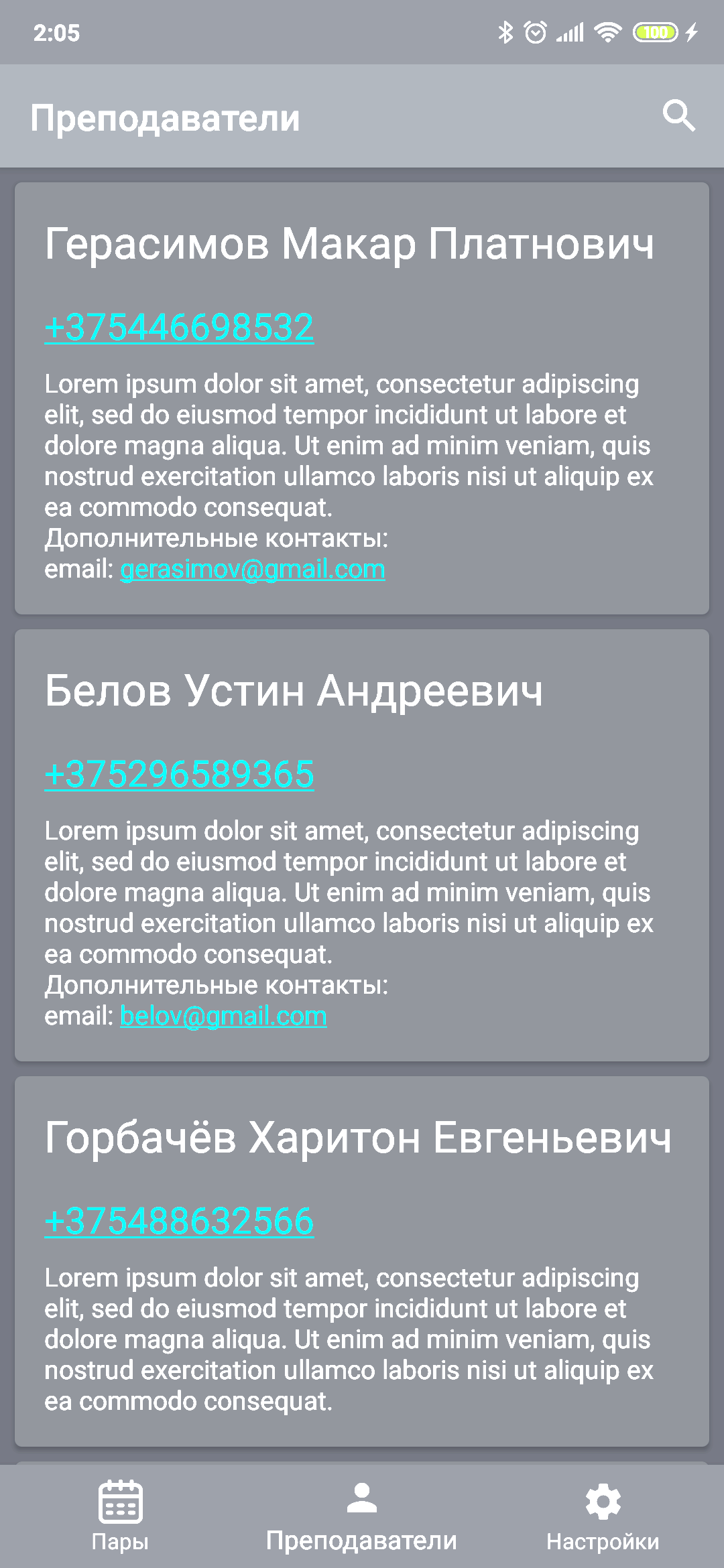


Рисунок 5.37 – Окно со списком преподавателей

Если необходимо быстро найти конкретного преподавателя, можно воспользоваться поиском. Для этого необходимо нажать на иконку лупы в правом верхнем углу экрана. В результате появится строка поиска в которую можно водить имеющуюся информацию. Это может имя, фамилия, отчество, номер телефона или даже отрывок из его краткой информации или его дополнительные контакты для связи. Алгоритм поиска просматривает все поля преподавателя и показывает только тех преподавателей, в полях которых было найдено совпадение по содержанию. Поиск не чувствителен к регистру, так что нет разницы, набирать текст для поиска преподавателя строчными или прописными буквами.

Пример пользования поиском представлен на рисунке 5.38.

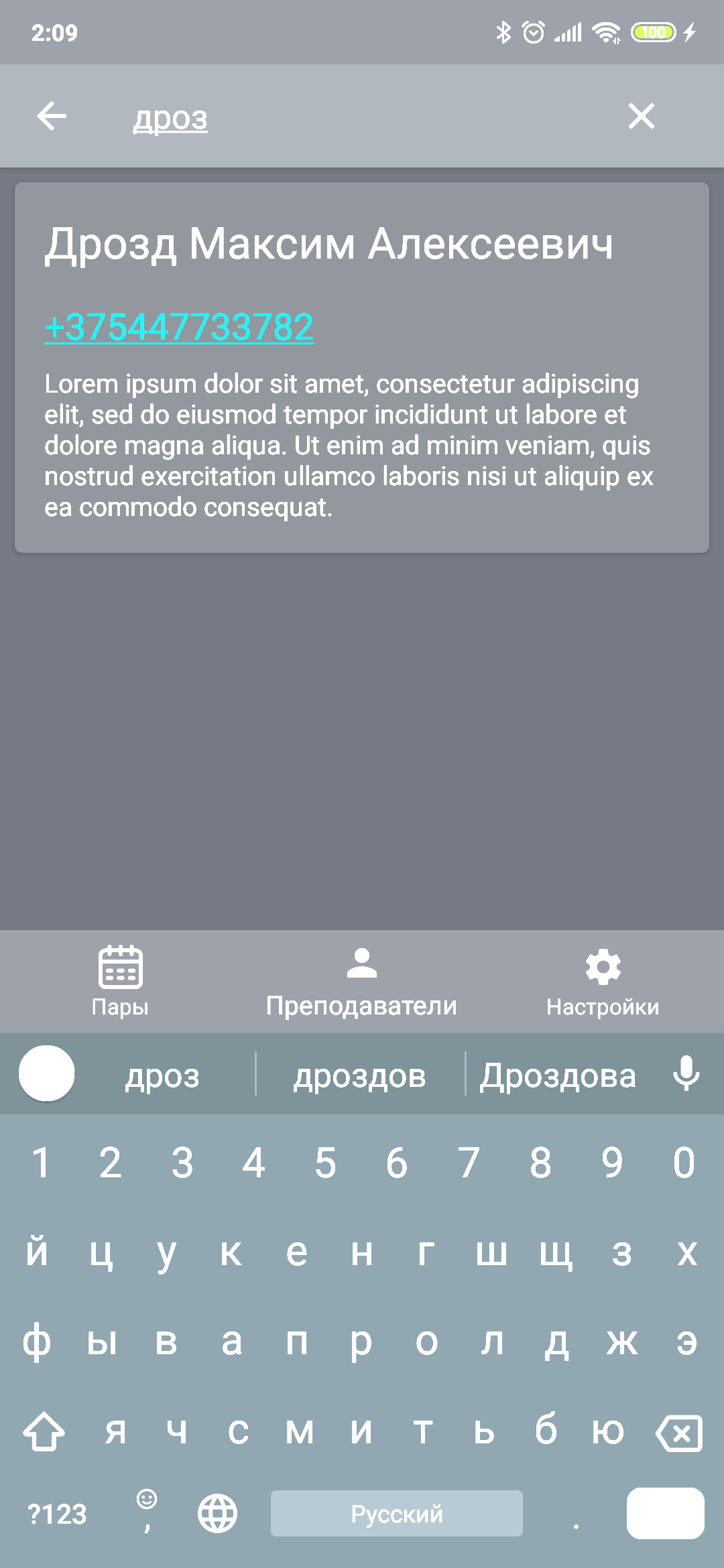


Рисунок 5.38 – Пример пользования поиском

Также на экране преподавателей, как и на экране с расписанием можно обновить содержимое страницы путем оттягивания верхней части экрана вниз. Тут реализована так называемая логика Pull To Refresh.

Таким образом был рассмотрен весь функционал разрабатываемого Android-приложения.

# 6. Экономический раздел

## **6.1 Технико-экономическое обоснование проекта разработки мобильного приложения на платформе Android «Расписание учебного заведения»**

Задачей экономического расчёта является определение с экономической точки зрения целесообразности разработки, внедрения и использования данного программного средства, относительно его платных аналогов.

Эффективность программного модуля определяется его качеством и эффективностью процесса разработки и сопровождения. Качество программного изделия определяется тремя составляющими:

* с точки зрения специалиста – пользователя данного программного продукта;
* с позиции использования ресурсов и их оценки;
* по выполнению требований на программное изделие.

Программное изделие должно быть разработано так, чтобы оно выполняло свои функции без лишних затрат ресурсов.

Содержание технико-экономического обоснования разработки программы заключается в следующем:

* расчёт технико-экономических показателей и выбор базы сравнения;
* определение трудоёмкости и стоимости программного обеспечения.

В задании на дипломное проектирование содержатся основные данные для выполнения технико-экономического обоснования результатов разработки. Тема дипломного проекта, связанная с разработкой программы, предполагает включение отдельных задач, составляющих содержание технико-экономического обоснования разработки, непосредственно включённые в технические разделы диплома.

Приведённые в данном разделе дипломного проекта результирующие таблицы, перечни данных, позволяют сопоставить результаты разработки и затраты на неё, чтобы сделать вывод об эффективности проекта.

6.1.1 Исходные данные

Таблица 6.1 – Исходные данные

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование показателя | Единица измерения | Условное обозначение | Норматив |
| 1 Коэффициент изменения скорости обработки информации | ед. |  |  |
| 2 Численность разработчиков | чел. |  |  |
| Продолжение таблицы 6.1 |  |  |  |
| Наименование показателя | Единица измерения | Условное обозначение | Норматив |
| 3 Базовая ставка в организации | руб. |  |  |
| 4 Тарифный коэффициент | ед. |  |  |
| 5 Фонд рабочего времени | ч. | ФРВ | 169,33 |
| 6 Коэффициент естественных потерь рабочего времени | ед. |  |  |
| 7 Коэффициент премирования | ед. |  |  |
| 8 Норматив дополнительной заработной платы | % |  |  |
| 9 Ставка отчислений в Фонд социальной защиты населения | % |  |  |
| 10 Ставка отчислений по обязательному страхованию от несчастных случаев на производстве и профессиональных заболеваний | % |  |  |
| 11 Цена одного машино-часа | руб. |  |  |
| 12 Норматив прочих затрат | % |  |  |
| 13 Норматив накладных расходов | % |  |  |
| 14 Норматив расходов на сопровождение и адаптацию | % |  |  |
| 15 Уровень рентабельности | % |  |  |
| 16 Ставка НДС | % |  |  |
| 17 Норматив расходов на освоение ПС | % |  |  |
| 18 Норматив расходов на пополнение оборотных средств в связи с использованием нового ПС | % |  |  |
| 19 Ставка налога на прибыль | % |  |  |
| 20 Норматив приведения разновременных затрат | ед. |  |  |

6.1.2 Расчёт цены и прибыли на программное средство

В современных рыночных экономических условиях ПС выступает преимущественно в виде продукции организаций, представляющей собой функционально завершённые и имеющие товарный вид ПС ВТ, реализуемые покупателям по рыночным отпускным ценам. Все завершённые разработки ПС ВТ являются научно-технической продукцией.

Широкое применение ВТ требует постоянного обновления и совершенствования ПС. Выбор эффективных проектов ПС связан с их экономической оценкой и расчётом экономического эффекта, который может определяться как у разработчика, так и у пользователя.

У разработчика экономический эффект выступает в виде чистой прибыли от реализации ПС, остающейся в распоряжении организации, а у пользователя – в виде экономии трудовых, материальных и финансовых ресурсов, получаемой за счёт:

* снижения трудоёмкости расчётов и алгоритмизации программирования и отладки программ за счёт использования ПС в процессе разработки автоматизированных систем обработки данных;
* сокращения расходов на оплату машинного времени и других ресурсов на отладку программ;
* снижения расходов на материалы (магнитные ленты, магнитные диски и прочие материалы);
* ускорение ввода в эксплуатацию новых систем;
* улучшения показателей основной деятельности в результате использования ПС.

Стоимостная оценка ПС у разработчиков предполагает определение затрат, что включает следующие статьи:

* заработная плата исполнителей - основная и дополнительная;
* отчисления в фонд социальной защиты населения;
* отчисления по обязательному страхованию от несчастных случаев на производстве и профессиональных заболеваний;
* расходы на материалы и комплектующие;
* расходы на спецоборудование;
* расходы на оплату машинного времени;
* прочие прямые затраты;
* накладные расходы.

На основании затрат рассчитывается себестоимость и отпускная цена ПС.

6.1.2.1 Общая характеристика разрабатываемого ПС ВТ

Программный продукт: Приложение на платформе Android «Расписание учебного заведения».

В программном продукте пользователю (учащемуся) предоставляется возможность просмотра текущих расписаний, получения информации о преподавателях, а администратору расписания (сотруднику учебного заведения) предоставляется возможность создания расписания и заполнения информации о преподавателях.

Тип программного средства: программное средство общего назначения.

Среды разработки приложения: Android Studio, Visual Studio code, Intellij IDEA.

Данный программный продукт относится ко второй группе сложности, так как он не обладает средствами высокой машинной графики, многомашинных комплексов и не имеет реализации особо сложных инженерных и научных расчетов.

Степень новизны программного продукта относится к группе «В», так как аналоги данного программного средства существуют, но все же эти программы имеют много недочетов и несоответствий с требованиями учебных заведений.

6.1.2.2 Определение объёма ПС ВТ

Объём ПС определяем путём подбора аналогов на основании классификации типов ПС, каталога функций ПС и аналогов ПС в разрезе функций, которые постоянно обновляются и утверждаются в установленном порядке.

На основании информации о функциях разрабатываемого ПС по каталогу функций определяется объем функций. Общий объем рассчитываем по формуле:

|  |  |
| --- | --- |
|  | (6.1) |

где, – общий объем ПС, условные машино-команды;

n – общее число функций;

– объем i-й функции ПС, условные машино-команды.

Содержание и объём функций разрабатываемого ПС ВТ представлены в таблице 6.2.

Таблица 6.2 – Содержание и объём функций на разрабатываемое ПС ВТ

|  |  |
| --- | --- |
| Содержание функции | Объём, условная машино-команда |
| Организация ввода информации | 11 |
| Продолжение таблицы 6.2 |  |
| Содержание функции | Объём, условная машино-команда |
| Контроль, предварительная обработка и ввод информации | 55 |
| Обработка входного заказа и формирование таблиц | 90 |
| Организация ввода/вывода информации в интерактивном режиме | 22 |
| Генерация структуры базы данных | 395 |
| Формирование баз данных | 175 |
| Обработка файлов | 80 |
| Формирование файла | 108 |
| Отладка прикладных программ в интерактивном режиме | 470 |
| Управление внешней памятью | 21 |
| Итого | 1 427 |

В связи с быстрым изменением ВТ определяем скорректированный объём функций по формуле:

|  |  |
| --- | --- |
|  | (6.2) |

где, – скорректированный объем ПС, условных машино-команд;

– общий объем ПС, условные машино-команды;

– коэффициент изменения скорости обработки информации

условных машино-команд

6.1.2.3 Расчёт трудоёмкости выполняемой работы

При определении трудоёмкости ПС учитываем объем ПС, объём документации, новизна и сложность ПС, язык программирования, степень использования типовых программ.

На основании общего объёма ПС определяем нормативную трудоёмкость по таблицам, устанавливается с учётом сложности ПС. А также устанавливается дополнительный коэффициент сложности ПС.

С учётом дополнительного коэффициента сложности рассчитываем общую трудоёмкость ПС:

|  |  |
| --- | --- |
|  | (6.3) |

где, – общая трудоемкость ПС, чел.-дн.;

– нормативная трудоемкость ПС, чел.-дн.;

– дополнительный коэффициент сложности ПС.

6.1.2.4 Расчёт основной заработной платы

В соответствии с «Рекомендациями по применению «Единой тарифной сетки работников Республики Беларусь»» и тарифными разрядами и коэффициентами должностей каждому исполнителю устанавливаем разряд и тарифный коэффициент, соответствующий установленному разряду:

|  |  |
| --- | --- |
|  | (6.4) |

где, – базовая ставка в организации за месяц, руб.;

– тарифный коэффициент.

Основную заработную плату исполнителей на конкретное ПС рассчитываем по формуле:

|  |  |
| --- | --- |
|  | (6.5) |

где, – основная заработная плата, руб.;

– базовая ставка в организации за день (Сзм разделить на 21,25), руб.;

– общая трудоемкость ПС, чел.-дн.;

– коэффициент естественных потерь рабочего времени, ед.;

– коэффициент премирования.

6.1.2.5 Расчёт дополнительной заработной платы

Дополнительная заработная плата на конкретное ПС включает выплаты, предусмотренные законодательством о труде (оплата отпусков, льготных часов, времени выполнения государственных обязанностей и других выплат, не связанных с основной деятельностью исполнителей), и определяется по нормативу в процентах к основной заработной плате:

|  |  |
| --- | --- |
|  | (6.6) |

где, – дополнительная заработная плата на конкретное ПС, руб.;

– основная заработная плата, руб.;

– норматив дополнительной заработной платы, %.

6.1.2.6 Расчёт отчислений в Фонд социальной защиты населения

Отчисления в Фонд социальной защиты населения (ФСЗН) определяются в соответствии с действующими законодательными актами по нормативу в процентном соотношении к фонду основной и дополнительной заработной платы исполнителей:

|  |  |
| --- | --- |
|  | (6.7) |

где, – сумма отчислений в Фонд социальной защиты населения, руб.;

– основная заработная плата, руб.;

– дополнительная заработная плата на конкретное ПС, руб.;

– норматив отчислений в Фонд социальной защиты населения, %.

6.1.2.7 Расчёт отчислений по обязательному страхованию от несчастных случаев на производстве и профессиональных заболеваний

Отчисление по обязательному страхованию от несчастных случаев на производстве и профессиональных заболеваний определяем в соответствии с действующими законодательными актами в зависимости от уровня риска отрасли, к которой относится организация-разработчик:

|  |  |
| --- | --- |
|  | (6.8) |

где, – сумма отчислений по обязательному страхованию от несчастных случаев на производстве и профессиональных заболеваний, руб.;

– основная заработная плата, руб.;

– дополнительная заработная плата на конкретное ПС, руб.;

– норматив отчислений по обязательному страхованию от несчастных случаев на производстве и профессиональных заболеваний, %.

6.1.2.8 Расчёт расходов на материалы

Расход на материалы определяются с учётом действующих нормативов. По данной статье отражаются расходы на магнитные носители, перфокарты, бумагу, красящие ленты и другие материалы, необходимые для разработки ПС. Нормы расхода материалов в суммарном выражении определяются в расчете на 100 строк исходного кода. Сумма затрат материалов рассчитывается по формуле:

|  |  |
| --- | --- |
|  | (6.9) |

где, – сумма расходов на материалы, руб.;

– норма расхода материалов в расчете на 100 строк исходного кода ПС, руб.;

– скорректированный объем ПС, условных машино-команд;

6.1.2.9 Расчёт расходов на оплату машинного времени

Расходы включают оплату машинного времени, необходимого для разработки и отладки ПС, которое определяется по нормативам (в машино-часах) на 100 строк исходного кода машинного времени в зависимости от характера решаемых задач и типа ПЭВМ:

|  |  |
| --- | --- |
|  | (6.10) |

где, – сумма расходов на оплату машинного времени, руб.;

– цена одного машино-ч, руб.;

– скорректированный объем ПС, условных машино-команд;

– норматив расхода машинного времени на отладку 100 строк исходного кода, машино-ч.

6.1.2.10 Расчёт прочих прямых затрат

Расходы на конкретное ПС включают затраты на приобретение и подготовку специальной научно-технической информации и специальной литературы. Мы определяем по нормативу в процентах к основной заработной плате:

|  |  |
| --- | --- |
|  | (6.11) |

где, – сумма прочих затрат, руб.;

– основная заработная плата, руб.;

– норматив прочих затрат в целом по организации, %.

6.1.2.11 Расчёт накладных расходов

Данные затраты, связанные с необходимостью содержания аппарата управления, а также с расходами на общехозяйственные нужды, относятся на конкретное ПС по нормативу в процентном отношении к основной заработной плате исполнителей.

|  |  |
| --- | --- |
|  | (6.12) |

где – сумма накладных расходов, руб.;

– основная заработная плата, руб.;

– норматив накладных расходов в целом по организации, %.

6.1.2.12 Расчёт суммы расходов на разработку ПС ВТ

|  |  |
| --- | --- |
|  | (6.13) |

где, – сумма расходов на разработку ПС ВТ, руб.;

6.1.2.13 Расчёт расходов на сопровождение и адаптацию

Кроме всего вышеперечисленного, организация-разработчик осуществляет затраты на сопровождение и адаптацию ПС, которые определяем по нормативу:

|  |  |
| --- | --- |
|  | (6.14) |

где, – сумма расходов на сопровождение и адаптацию ПС ВТ, руб.;

– норматив расходов на сопровождение и адаптацию, %.

6.1.2.14 Расчёт полной себестоимости разработки ПС ВТ

Общая сумма расходов на разработку (с затратами на сопровождение и адаптацию) – полная себестоимость ПС – определяем по формуле:

|  |  |
| --- | --- |
|  | (6.15) |

где, – полная себестоимость ПС ВТ, руб.

6.1.2.15 Определение отпускной цены на ПС ВТ

Отпускная цена определяется на основании цены разработчика, которая формируется на основе показателя рентабельности продукции. Рентабельность и прибыль по созданному ПС определяем исходя из результатов анализа рыночных условий, переговоров с заказчиком (потребителем) и согласования с ним отпускной цены, включающей дополнительно налог на добавленную стоимость и отчисления в целевые бюджетные фонды из выручки от реализации продукции. Прибыль рассчитывается по формуле:

|  |  |
| --- | --- |
|  | (6.16) |

где – прибыль от реализации ПС, руб.;

– уровень рентабельности ПС, %.

Прогнозируемая цена разработчика ПС без налогов рассчитываем по формуле:

|  |  |
| --- | --- |
|  | (6.17) |

где, – прогнозируемая цена разработчика ПС, руб.

Сумма налога на добавленную стоимость рассчитываем по формуле:

|  |  |
| --- | --- |
|  | (6.18) |

где НДС – сумма налога на добавленную стоимость, руб.;

– ставка НДС, %.

Прогнозируемую отпускную цену рассчитываем по формуле:

|  |  |
| --- | --- |
|  | (6.19) |

где – прогнозируемая отпускная цена, руб.

6.1.3 Расчёт экономического эффекта от применения программного средства у пользователя

Таблица 6.3 – Исходные данные для сравнения по базовому и новому вариантам

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Наименование показателя | Обозначение | Единица измерения | Значения показателя в | | Наименование источника информации |
| базовом варианте | новом варианте |
| 1. Средняя трудоемкость работ в расчете на 100 КБ |  | человеко-час на 100 КБ | 3,95 | 1,41 | По данным пользователя |
|  |
| 2.Средний расход машинного времени в расчете на 100 КБ |  | машино-час на 100 КБ | 7,6 | 7,2 | По данным пользователя |
|  |
| 3.Средний расход материалов в расчете на 100 КБ |  | руб. на 100 КБ | 0,03 | 0,02 | По данным пользователя |
|  |

Объем работ в зависимости от функциональной группы и назначения ПС определяется по формуле:

|  |  |
| --- | --- |
|  | (6.20) |

где – скорректированный объем ПС, условных машино-команд;

– коэффициент применения ПС.

6.1.3.2 Расчёт капитальных затрат

Общие капитальные вложения заказчика (потребителя), связанные с приобретением, внедрением и использованием ПС, рассчитываются по формуле

|  |  |
| --- | --- |
|  | (6.21) |

где, – затраты пользователя на приобретение ПС по отпускной цене разработчика с учетом стоимости услуг по эксплуатации и сопровождению, руб.;

– затраты пользователя на освоение ПС, руб.;

– затраты на доукомплектацию ВТ техническими средствами в связи с внедрением нового ПС, руб.;

– затраты на пополнение оборотных средств в связи с использованием нового ПС, руб.

Затраты на освоение ПС и на пополнение оборотных средств рекомендуется рассчитывать по формулам:

|  |  |
| --- | --- |
|  | (6.22) |

|  |  |
| --- | --- |
|  | (6.23) |

6.1.3.3 Расчёт экономии основных видов ресурсов в связи с использованием нового ПС

Экономия затрат на заработную плату при использовании нового ПС в расчете на объем выполненных работ:

|  |  |
| --- | --- |
|  | (6.24) |

где, – экономия затрат на заработную плату при решении задач с использованием нового ПС, руб.;

– экономия затрат на заработную плату при решении задач с использованием нового ПС в расчете на 100 КБ, руб.;

А – объем выполненных работ с использованием нового ПС, 100 КБ.

Экономия затрат на заработную плату в расчете на 100 КБ:

|  |  |
| --- | --- |
|  | (6.25) |

где, – среднемесячная заработная плата одного программиста, руб.;

, – трудоемкость работ в расчете на 100 строк кода при базовом и новом варианте соответственно, чел.-ч;

ФРВ – фонд рабочего времени за месяц, ч.

|  |  |
| --- | --- |
|  | (6.26) |

Экономия начислений на заработную плату при использовании нового ПС в расчете на объем выполненных работ:

|  |  |
| --- | --- |
|  | (6.27) |

где, – экономия начислений на заработную плату при решении задач с использованием нового ПС, руб.;

– коэффициент начислений на заработную плату, ед.

|  |  |
| --- | --- |
|  | (6.28) |

Экономия затрат на оплату машинного времени в расчете на выполненный объем работ в результате применения нового ПС:

|  |  |
| --- | --- |
|  | (6.29) |

где, – экономия затрат на оплату машинного времени при решении задач с использованием нового ПС, руб.;

– экономия затрат на оплату машинного времени при решении задач с использованием нового ПС в расчете на 100 КБ, руб.

Экономия затрат на оплату машинного времени в расчете на 100 КБ определяется по формуле:

|  |  |
| --- | --- |
|  | (6.30) |

где, , – средний расход машинного времени в расчете на 100 КБ при применении базового и нового варианта ПС соответственно, машино-ч (приложение Х).

Экономия затрат на материалы при использовании нового ПС в расчете на объем выполненных работ определяется по формуле:

|  |  |
| --- | --- |
|  | (6.31) |

где, – экономия затрат на материалы при использовании нового ПС, руб.;

– экономия затрат на материалы в расчете на 100 КБ при использовании нового ПС, руб.

|  |  |
| --- | --- |
|  | (6.32) |

где, , – средний расход материалов у пользователя в расчете на 100 КБ при использовании базового и нового варианта ПС соответственно, руб.

Общая годовая экономия текущих затрат, связанных с использованием нового ПС, определяется по формуле:

|  |  |
| --- | --- |
|  | (6.33) |

6.1.3.4 Расчёт экономического эффекта

Внедрение нового ПС позволит пользователю сэкономить на текущих затратах, то есть практически получить на эту сумму дополнительную прибыль. Для пользователя в качестве экономического эффекта выступает лишь чистая прибыль – дополнительная прибыль, остающаяся в его распоряжении.

|  |  |
| --- | --- |
|  | (6.34) |

где, – прирост прибыли, руб.;  
 – ставка налога на прибыль, %.

Коэффициент привидения (), который рассчитывается по формуле:

|  |  |
| --- | --- |
|  | (6.35) |

где, – норматив приведения разновременных затрат и результатов;

– расчетный год, = 1;

t – номер года, результаты и затраты которого приводятся к расчетному.

Таблица 6.4 – Расчёт экономического эффекта от использования нового ПС

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Показатель | Ед. изм. | Методика расчёта | Год | | | |
| 2020 | 2021 | 2022 | 2023 |
| Результаты: | | | | | | |
| Прирост прибыли за счет экономии затрат | руб. | ∆ПЧ | 3 062,06 | 3 062,06 | 3 062,06 | 3 062,06 |
| Сумма прибыли с учетом фактора времени | руб. | ∆ПЧ ALFAT | 3 062,06 | 2 786,47 | 2 572,13 | 2 357,78 |
| Затраты: | | | | | | |
| Затраты на приобретение ПС | руб. |  | 2 854,17 | - | - | - |
| Затраты на освоение ПС | руб. |  | 28,54 | - | - | - |
| Затраты на пополнение оборотных средств | руб. |  | 57,08 | 57,08 | 57,08 | 57,08 |
| Сумма затрат | руб. |  | 2 939,79 | 57,08 | 57,08 | 57,08 |
| Сумма затрат с учётом фактора времени | руб. |  | 2 939,79 | 51,94 | 47,94 | 43,95 |
| Экономический эффект | руб. |  | 122,27 | 2 734,53 | 2 524,19 | 2 313,83 |
| Экономический эффект с нарастающим итогом | руб. |  | 122,27 | 2 856,80 | 5 380,99 | 7 694,82 |
| Коэффициент приведения | ед. |  | 1,00 | 0,91 | 0,84 | 0,77 |

6.1.4 Вывод

В ходе проведения расчётов затрат на разработку программного средства учитывались все виды расходов: затраты на приобретение ПС, затраты на освоение ПС, а также затраты на пополнение оборотных средств. В итоге было высчитано, что пик затрат приходится на первый год – 2 939,79 руб. В последующие года затраты будут составлять одну и ту же сумму – 57,08 руб. Сумма затрат с учетом времени не сильно отличается от суммы затрат без учета времени: за первый год – 2 939,79 руб., второй – 51,94 руб., третий – 47,94 руб., четвертый – 43,95 руб.

Экономический эффект в каждый из расчётных периодов составил соответственно 122,27 руб., 2 734,53 руб., 2 524,19 руб. и 2 313,83 руб.

В результате расчётов было выявлено, что вложенные средства в течении первого года окупились, экономический эффект с нарастающим итогом составил 122,27. В последующих периодах окупаемость растёт и составляет: во втором году – 2 856,80 руб., в третьем – 5 380,99 руб., а в четвёртом – 7 694,82 руб.

На основании выполненных расчётов можно сделать вывод, что создание и внедрение программного средства является целесообразным и затраты на разработку окупятся уже через год. Более того, через год данное программное обеспечение позволит существенно сократить производственные расходы.

# 7. Охрана труда

## **7.1 Проектирование и расчет искусственного освещения светотехнической среды для работы с ПЭВМ при использовании мобильного приложения на платформе Android «Расписание учебного заведения»**

7.1.1 Характеристика помещения и выполняемых работ на проектируемом предприятии.

Качество производственного освещения в значительной мере сказывается на безопасности и производительности труда человека. При плохом освещении человек быстро устает, работает менее продуктивно, возникает потенциальная опасность ошибочных действий и несчастных случаев. Кроме того, плохое освещение может привести к профессиональным заболеваниям, например, таким, как рабочая миопия (близорукость), спазм аккомодации и др.

Гигиенические требования к производственному освещению, основанные на психофизических особенностях восприятия света и его влияния на организм человека, могут быть сведены к следующим:

* спектральный состав света, создаваемый искусственными источниками, должен приближаться к естественному;
* уровень освещенности должен быть достаточным и соответствовать гигиеническим нормам, учитывающим особенности зрительной работы;
* должна обеспечиваться равномерность и естественность уровня освещенности в помещении во избежание частой адаптации и утомления зрения;
* освещение не должно создавать блесткости как самих источников света, так и других предметов в пределах рабочей зоны.

В настоящее время в Республике Беларусь действуют санитарные нормы и правила из «Требования при работе с видеодисплейными терминалами и электронно-вычислительными машинами», утвержденными постановлением Министерства здравоохранения Республики Беларусь от 28 июня 2013 г. № 59. Приведу наиболее значимые из них:

* освещенность на поверхности стола в зоне размещения рабочего документа должна быть 300-500 люкс. Освещение не должно создавать бликов на поверхности экрана. Освещенность поверхности экрана не должна быть более 300 люкс;
* помещения для эксплуатации ЭВМ должны иметь естественное и искусственное освещение;
* запрещается выполнение основной работы с использованием ЭВМ на постоянных рабочих местах без естественного освещения, если это не обусловлено технологическим процессом;
* естественное освещение на рабочих местах с ВДТ, ЭВМ и ПЭВМ должно осуществляться через световые проемы, ориентированные преимущественно на север, северо-восток, восток, запад или северо-запад;
* рабочие столы следует размещать таким образом, чтобы экраны ЭВМ были ориентированы боковой стороной к световым проемам (исключение составляет периметральная расстановка рабочих мест), чтобы естественный свет падал преимущественно слева;
* искусственное освещение в помещениях для эксплуатации ЭВМ должно осуществляться системой общего равномерного освещения. В производственных, административных и общественных помещениях в случаях преимущественной работы с документами следует применять системы комбинированного освещения;
* в качестве источников света при искусственном освещении следует применять преимущественно люминесцентные лампы типа ЛБ и компактные люминесцентные лампы. При устройстве отраженного освещения в производственных, административных и общественных помещениях допускается применение металлогалогенных ламп. В светильниках местного освещения допускается применение ламп накаливания, в том числе галогенных;
* общее освещение при использовании люминесцентных светильников следует выполнять в виде сплошных или прерывистых линий светильников, расположенных сбоку от рабочих мест, параллельно линии зрения пользователя при рядном расположении видеодисплейных терминалов [2].

С целью обеспечения нормальных условий труда и защиты зрения человека в производственных помещениях должно устанавливаться освещение, отвечающее требованиям соответствующих норм и правил. Необходимо также при организации освещения обеспечить достаточно равномерное распределение светового потока и яркости на рабочей поверхности и в пределах окружающего пространства, отсутствие резких теней в поле зрения, оптимальную направленность светового потока. При несоблюдении этих требований возможна неразличимость или искаженность форм и размеров объектов, рельефности элементов, повышенная утомляемость за счет перенапряжения зрения и его ухудшения, обусловленной частой переадаптацией при крайней неравномерности освещения.

Важным санитарно-гигиеническим требованием является устранение пульсации (моргания) освещенности, что обеспечивается стабилизацией питающего напряжения, жестким креплением светильников, специальным включением люминесцентных ламп в светильнике.

В соответствие с требованиями к помещениям для эксплуатации ЭВМ в помещениях должно быть естественное и искусственное освещение. При этом дополнительное искусственное освещение применяется не только в темное, но и в светлое время суток.

Основное отличие ночных условий труда от дневных состоит в том, что при ночных условиях отсутствует достаточная освещенность поля зрения работающего равномерно распределенным световым потоком. Поэтому необходимо создать такое искусственное освещение, при котором суммарный световой поток от всех установленных в рабочей зоне светильников распределялся равномерно.

Таким образом при проектировании помещения для использования ЭВМ необходимо применять систему общего равномерного искусственного освещения.

Это проектирование представляет собой последовательность решения следующих задач:

* выбор типа источников света (ламп);
* выбор типа светильников;
* размещение светильников в плане помещения и определение их количества;
* расчет светового потока ламп светильников;
* выбор стандартной лампы.

Исходными данными для расчета являются:

* гигиеническая норма освещенности (300 лк - 500 лк);
* габаритные размеры помещения;
* коэффициенты отражения рабочих поверхностей, поверхностей стен и потолка [17].

Более подробно эти шаги будут описаны в следующих разделах.

Разрабатываемое мобильное приложение «Расписание учебного заведения» не предполагает длительных сессий. За счет короткого времени использования приложения, им можно пользоваться без вреда для здоровья в любых световых условиях, которые не причиняют вред сами по себе. Также следует не забывать про регулировку яркости экрана до комфортных значений.

7.1.2 Гигиеническая оценка искусственного освещения и его нормирование.

Чтобы оценивать свет, надо для начала понять, что представляет собой цвет, какими параметрами характеризуется и как влияет на человека.

Видимое излучение (свет) – участок общего электромагнитного спектра, состоящего из 7 основных цветов, именно он непосредственно вызывает зрительное ощущение.

Свет является естественным условием жизнедеятельности человека. Он оказывает положительное влияние на эмоциональное состояние человека, воздействует на обмен веществ, сердечно-сосудистую, нервно-психическую системы, является важным стимулятором не только зрительного анализатора, но и организма в целом. Более 80 % всей информации о внешней среде поступает в мозг человека через глаза [15].

Рациональное освещение производственных помещений оказывает положительное психофизиологическое воздействие на работающих, способствует повышению производительности труда, обеспечению его безопасности, сохранению высокой работоспособности человека в процессе труда.

При недостаточной освещенности и плохом качестве освещения состояние зрительных функций исходно неудовлетворительное, в процессе выполнения работы повышается утомление глаз, возрастает опасность травматизма. Установлено, что плохое освещение является причиной примерно 5 % несчастных случаев на предприятиях, а также глазных болезней, головных болей, быстрой утомляемости.

С другой стороны, существует опасность отрицательного влияния на органы зрения слишком большой яркости (блескости) источников света. Следствием этого может явиться временное нарушение зрительных функций глаза (явление слепимости).

С целью обеспечения нормальных условий труда и защиты зрения человека в производственных помещениях должно быть установлено освещение, отвечающее требованиям соответствующих норм и правил.

Нормы освещенности построены на основе классификации работ по определенным количественным признакам.

Производственное освещение нормируется в зависимости от:

* точности зрительной работы;
* яркости фона;
* контраста объекта и фона;
* системы освещения.

Точность зрительной работы характеризуется минимальным размером объекта различения. Объект различения – это элемент рассматриваемого объекта минимального размера, который нужно узнавать и различать. По степени точности все зрительные работы делятся на восемь разрядов. В свою очередь разряды делятся на четыре подразряда в зависимости от характеристики фона и контраста между объектом и фоном. Деление разрядов зрительных работ на подразряды дает возможность более дифференцированно выбрать освещенность для каждой зрительной работы [17].

Гигиеническая оценка производственного освещения заключается в измерении или расчете фактической освещенности на рабочей поверхности и сравнении ее с нормативным значением, которое указано в строительных или отраслевых нормах освещенности рабочих мест, в зависимости от вида работ. Рабочей считается поверхность, на которой непосредственно производится работа.

Нормирование искусственного освещения осуществляется непосредственно по минимальной освещенности рабочей поверхности (лк) в соответствии с СНБ 2.04.05–98.

Нормируемые значения освещенности в люксах, отличающиеся на одну ступень, образуют шкалу: 30; 50; 75; 100; 150; 200; 300; 400; 500; 600; 750; 1000; 1250; 1500; 2000; 2500; 3000; 3500; 4000; 4500; 5000 лк [18].

В ряде случаев нормы освещенности необходимо повышать или понижать на одну ступень по шкале освещенности. Например, если работа связана с повышенной опасностью травматизма, размещением деталей на движущихся поверхностях, если напряженная зрительная работа производится непрерывно в течение рабочего дня или различаемые объекты расположены от глаз далее, чем на 0,5 м, нормы освещенности повышаются на одну ступень согласно шкале освещенности. При кратковременном пребывании людей или при наличии оборудования, не требующего постоянного обслуживания, нормы освещенности следует снижать на одну ступень.

Нормирование естественного освещения. Непостоянство естественного света даже в течение короткого промежутка времени вызвало необходимость нормировать естественное освещение с помощью относительного показателя – коэффициента естественной освещенности.

КЕО – это отношение естественной освещенности, создаваемой в некоторой точке заданной плоскости внутри помещения, к одновременному значению наружной горизонтальной освещенности, создаваемой светом полностью открытого небосвода, выраженное в процентах:

|  |  |
| --- | --- |
|  | (7.1) |

где - освещенность какой-либо точки внутри помещения;

- освещенность точки вне помещения.

В небольших помещениях при одностороннем боковом естественном освещении нормируется минимальное значение КЕО в точке, расположенной на расстоянии 1 м от стены, наиболее удаленной от световых проемов, на пересечении вертикальной плоскости характерного разреза помещения и условной рабочей поверхности. При двустороннем боковом освещении – в точке посередине помещения на пересечении вертикальной плоскости характерного разреза помещения и условной рабочей поверхности. При верхнем или комбинированном естественном освещении нормируется среднее значение КЕО в точках, расположенных на пересечении вертикальной плоскости характерного разреза помещения и условной рабочей поверхности. Первая и последняя точки принимаются на расстоянии 1м от поверхности стен (перегородок) или осей колонн.

Характерный разрез помещения – поперечный разрез посередине помещения, плоскость которого перпендикулярна к плоскости остекления световых проемов (при боковом освещении) или к продольной оси пролетов помещения. В характерный разрез помещения должны попадать участки с наибольшим количеством рабочих мест, а также точки рабочей зоны, наиболее удаленные от световых проемов.

Условная рабочая поверхность – условно принятая горизонтальная поверхность, расположенная на высоте 0,8 м от пола.

Совмещенное освещение нормируют так же, как и естественное, с помощью КЕО в зависимости от выполняемого разряда зрительной работы и конструктивного исполнения искусственного освещения.

7.1.3 Моделирование искусственного освещения для рабочих мест при работе с проектируемым объектом.

С целью оптимизации освещения рабочих мест, создания благоприятных условий труда, повышения работоспособности проводят инженерно-технические мероприятия по расчету и проектированию производственного освещения.

7.1.3.1 Расчет естественного освещения

Основной задачей светотехнических расчетов при естественном освещении является определение требуемой площади световых проемов. Предварительный расчет площади световых проемов производится [18]:

При боковом освещении помещений по формуле:

|  |  |
| --- | --- |
|  | (7.2) |

где - площадь световых проемов (в свету) при боковом освещении, ;

- площадь пола помещения, ;

- нормированное значение КЕО, %;

- коэффициент запаса;

- световая характеристика окон;

- коэффициент, учитывающий затенение окон противостоящими зданиями;

- общий коэффициент светопропускания;

- коэффициент, учитывающий повышение КЕО при боковом освещении благодаря свету, отраженному от поверхности помещения и подстилающего слоя, прилегающего к зданию.

При верхнем освещении по формуле:

|  |  |
| --- | --- |
|  | (7.3) |

где - площадь световых проемов (в свету) при верхнем освещении, ;

- площадь пола помещения, ;

- нормированное значение КЕО, %;

- коэффициент запаса;

- световая характеристика фонаря или светового проема в плоскости покрытия;

- общий коэффициент светопропускания;

- коэффициент, учитывающий повышение КЕО при верхнем освещении благодаря свету, отраженному от поверхности помещения;

- коэффициент, учитывающий тип фонаря.

При недостаточной естественной освещенности выполняется проектирование и расчет искусственного освещения.

7.1.3.2 Расчет искусственного освещения

Основной задачей светотехнических расчетов при искусственном освещении является определение потребной мощности электрической осветительной установки. Ниже будут приведены наиболее известные методы, применяемыми при расчете искусственного освещения [18].

7.1.3.2.1 Метод коэффициента использования светового потока

Этот метод предназначен для расчета общего равномерного освещения горизонтальных поверхностей при отсутствии затемняющего оборудования.

Порядок выполнения расчета при проектировании системы общего равномерного искусственного освещения для производственного помещения представляет собой последовательное решение следующих задач:

* выбор типа источника света (лампы – газоразрядные или накаливания);
* выбор типа светильника исходя из характеристик помещения;
* определение расчетной высоты подвеса светильника по формуле:

|  |  |
| --- | --- |
|  | (7.4) |

где H - высота помещения, м;

- высота свеса светильника от потолка, м ( 0 - 1,5 м);

- высота освещаемой рабочей поверхности, м ( 0,8 м).

* определение расстояния между светильниками L (рядами светильников) по формуле:

|  |  |
| --- | --- |
|  | (7.5) |

где, - оптимальное отношение расстояния между светильниками L к высоте их подвеса .

Размещение светильников на плане помещения. Предварительно необходимо определить расстояние от крайнего светильника (ряда светильников) до стены помещения по следующим формулам:

Если рабочие места расположены у стен:

|  |  |
| --- | --- |
|  | (7.6) |

Если у стен расположены проходы:

|  |  |
| --- | --- |
|  | (7.7) |

Затем оценим, сколько рядов можно разместить в помещении:

|  |  |
| --- | --- |
|  | (7.8) |

откуда

|  |  |
| --- | --- |
|  | (7.9) |

где - количество светильников в ряду;

a - длина помещения, м;

- длина светильника, м.

По полученным данным на плане помещения, вычерченном в масштабе, производится окончательное уточнение расположения светильников и их количества.

Определение коэффициента использования светового потока в зависимости от индекса помещения, типа светильника и коэффициентов отражения потолка , стен и рабочей поверхности .

Индекс помещения определяется по формуле:

|  |  |
| --- | --- |
|  | (7.10) |

где i - индекс помещения;

a и b - длина и ширина помещения, м.

Расчет светового потока лампы, необходимого для создания на рабочих поверхностях освещенности на все время эксплуатации осветительной установки.

Световой поток лампы определяется по формуле:

|  |  |
| --- | --- |
|  | (7.11) |

где F - световой поток одной лампы, лм;

- нормативная минимальная освещенность, лк;

S - освещаемая площадь помещения, ;

N - число светильников на плане помещения;

n - число ламп в одном светильнике;

- коэффициент использования светового потока (в долях);

Z - коэффициент неравномерности минимальной освещенности: рекомендуется принимать значение Z, равное 1,15 для ламп накаливания и равное 1,1 для люминесцентных ламп.

Выбор ближайшей стандартной лампы по полученному в результате расчета требуемому световому потоку. Допускается отклонение светового потока лампы не более чем на –10 ... +20 %.

При невозможности выбора лампы с таким приближением корректируют количество светильников.

7.1.3.2.2 Точечный метод

Этот метод позволяет рассчитать освещенность конкретной точки на горизонтальной и наклонной поверхностях при общем локализованном и комбинированном освещении.

Расчет освещенности выполняют по формуле:

|  |  |
| --- | --- |
|  | (7.12) |

где E - освещенность, лк;

I - сила света в направлении от источника на данную точку рабочей поверхности, кд;

- угол между нормалью к рабочей поверхности и направлением светового потока на источник, градусы;

- коэффициент запаса;

- высота подвеса светильника над рабочей поверхностью, м.

7.1.3.2.3 Метод удельной мощности (метод Ватт)

Это наиболее простой метод для приближенного расчета искусственного освещения. Он основан на определении по светотехническим справочникам удельной мощности осветительной установки в зависимости от заданных параметров установки и числа светильников. Требуемая мощность лампы подсчитывается по выражению:

|  |  |
| --- | --- |
|  | (7.13) |

где - мощность одной лампы, Вт;

- удельная мощность, Вт/м 2;

S - площадь помещения, м2;

N - число светильников.

Таким образом были рассмотрены все методы расчета искусственного освещения.

# Заключение

В ходе выполнения дипломной работы были решены основные задачи и получены следующие результаты:

* выполнен анализ предметной области;
* рассмотрены существующие программные решения;
* сформулированы требования к разрабатываемой системе;
* выполнено проектирование информационной системы;
* написан программный код на языках программирования Kotlin и JS.

В результате выполнения вышеуказанных задач, была успешно достигнута основная цель работы – реализована автоматизированная информационная система учёта расписания учебного заведения. Система соответствует сформулированным к разработке требованиям, обладает интерактивными элементами управления и интуитивно–понятным интерфейсом. Админ панель и мобильное приложение под платформу «Android» имеет простую навигационную структуру, дружественный и не перегруженный интерфейс, что облегчает доступ к необходимой информации.

В ходе тестирования в мобильном приложении и админ панели были найдены некоторые незначительные дефекты, которые были устранены в последующих этапах разработки.

Так же стоит отметить, что в дальнейшем админ панель и мобильное приложение можно модернизировать. Например, изменить интерфейс мобильного приложения, или добавить новый функционал в админ-панель.

Данную систему приложений можно применять в любом учебном заведении. Эту систему достаточно легко внедрить, поскольку она может быть изолирована и использована как полностью самостоятельный продукт, не зависящий от других сервисов. Из-за простоты админ панели, с ней могут быстро разобраться все желающие преподаватели и сотрудники, составляющие учебное расписание. А из-за простоты мобильного приложения у учащихся не будут возникать вопросы о том, как пользоваться этим приложение.

Автоматизация процессов распространения расписания поможет учащимся получать информацию о текущих и предстоящих парах в более короткие сроки, а также позволит учебным заведениям более быстро и эффективно составлять расписания.

# Список используемых источников

Литература

1. Медведев, В.И. M 42 Особенности объектно-ориентированного программирования на C++/CLI, C# и Java/ В.И. Медведев, 2-е изд., испр. и доп. - Казань: РИЦ «Школа», 2010

Нормативные документы

1. СанПиН от 28.06.2013 № 59. Санитарные нормы и правила «Требования при работе с видеодисплейными терминалами и электронно-вычислительными машинами»

Электронный ресурс

1. Android documentation – [Электронный ресурс]. – Режим доступа: https://developer.android.com/ Дата доступа: 19.04.2020.
2. Javalin tutorials – [Электронный ресурс]. – Режим доступа: https://javalin.io/tutorials/ Дата доступа: 19.04.2020.
3. Расписание БГУИР – [Электронный ресурс]. – Режим доступа: https://play.google.com/store/apps/details?id=com.bakan.universchedule Дата доступа: 19.04.2020.
4. BSUIR Schedule – [Электронный ресурс]. – Режим доступа: https://play.google.com/store/apps/details?id=com.bsuir.pechuro.bsuirschedule Дата доступа: 19.04.2020.
5. БГУИР Расписание– [Электронный ресурс]. – Режим доступа: https://play.google.com/store/apps/details?id=by.bsuir.schedule Дата доступа: 19.04.2020.
6. Расписание – [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/Расписание Дата доступа: 19.04.2020.
7. Урок – [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/Урок Дата доступа: 19.04.2020.
8. Ученик – [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/Ученик Дата доступа: 19.04.2020.
9. Образовательное учреждение – [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/Образовательное учреждение Дата доступа: 19.04.2020.
10. Пара – [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/ Пара Дата доступа: 19.04.2020.
11. MVP – [Электронный ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/Model-View-Presenter> Дата доступа: 19.04.2020.
12. Клиент-серверная архитектура – [Электронный ресурс]. – Режим доступа: <https://habr.com/ru/post/495698/> Дата доступа: 19.04.2020. Министерство охраны труда – [Электронный ресурс] – Режим доступа. http://mintrud.gov.by/ru/oxrana-truda– Дата доступа 02.02.2020
13. Охрана труда – [Электронный ресурс] – Режим доступа. https://studfile.net/preview/2954986/page:3/. – Дата доступа 02.02.2020
14. Нормативно правовая база – [Электронный ресурс] – Режим доступа. <http://minzdrav.gov.by/ru/dlya-spetsialistov/normativno-pravovaya-baza/tekhnicheskie-normativnye-pravovye-akty/teksty-tekhnicheskikh-normativnykh-aktov/sanitarnye-normy-pravila-i-gigienicheskie-normativy.php> – Дата доступа 02.02.2020
15. Освещение помещений и рабочих мест – [Электронный ресурс] – Режим доступа. https://www.protrud.com/обучение/учебный-курс/освещение-помещений-и-рабочих-мест/. – Дата доступа 02.02.2020
16. Расчет освещения – [Электронный ресурс]. – Режим доступа: https://studbooks.net/1697995/finansy/proektirovanie\_raschet\_iskusstvennogo\_osvescheniya\_vypolneniya\_rabot\_razrabotke\_proektiruemogo\_obekta. – Дата доступа: 01.02.2020

# Приложение А

(обязательное)

**Код программы**

class LoginController {

companion object {

val accessManager = AccessManager { handler, ctx, permittedRoles ->

when {

permittedRoles.contains(Roles.ANYONE) -> handler.handle(ctx)

ctx.getRolesOfCurrentUser().any { permittedRoles.contains(it) } -> handler.handle(ctx)

else -> ctx.status(401).json("not suthorized LOL")

}

}

}

}

private fun Context.getRolesOfCurrentUser(): Set<Roles> {

if(!basicAuthCredentialsExist()) return emptySet()

return Main.userInteractor.getUserRoles(basicAuthCredentials())

}

fun main() {

val app = Javalin.create { config ->

config.enableDevLogging()

config.enableCorsForAllOrigins()

config.accessManager(LoginController.accessManager)

}.start(80)

app.routes {

path(Path.TEACHERS) {

get(TeacherController.fetchAllTeachers, roles(Roles.WORKER, Roles.ADMIN))

post(TeacherController.addTeacher, roles(Roles.WORKER, Roles.ADMIN))

post("/check", TeacherController.canDeleteTeacher, roles(Roles.WORKER, Roles.ADMIN))

path(":teacher-id") {

get(TeacherController.getTeacher, roles(Roles.WORKER, Roles.ADMIN))

put(TeacherController.editTeacher, roles(Roles.WORKER, Roles.ADMIN))

delete(TeacherController.deleteTeacher, roles(Roles.WORKER, Roles.ADMIN))

}

}

path(Path.DISCIPLINES) {

get(DisciplineController.fetchAllDisciplines, roles(Roles.WORKER, Roles.ADMIN))

post(DisciplineController.addDiscipline, roles(Roles.WORKER, Roles.ADMIN))

post("/check", DisciplineController.canDeleteDiscipline, roles(Roles.WORKER, Roles.ADMIN))

path(":discipline-id") {

get(DisciplineController.getDiscipline, roles(Roles.WORKER, Roles.ADMIN))

put(DisciplineController.editDiscipline, roles(Roles.WORKER, Roles.ADMIN))

delete(DisciplineController.deleteDiscipline, roles(Roles.WORKER, Roles.ADMIN))

}

}

path(Path.OFFICES) {

get(OfficeController.fetchAllOffices, roles(Roles.WORKER, Roles.ADMIN))

post(OfficeController.addOffice, roles(Roles.WORKER, Roles.ADMIN))

post("/check", OfficeController.canDeleteOffice, roles(Roles.WORKER, Roles.ADMIN))

path(":office-id") {

get(OfficeController.getOffice, roles(Roles.WORKER, Roles.ADMIN))

put(OfficeController.editOffice, roles(Roles.WORKER, Roles.ADMIN))

delete(OfficeController.deleteOffice, roles(Roles.WORKER, Roles.ADMIN))

}

}

path(Path.GROUPS) {

get(GroupController.fetchAllGroups, roles(Roles.WORKER, Roles.ADMIN))

post(GroupController.addGroup, roles(Roles.WORKER, Roles.ADMIN))

post("/check", GroupController.canDeleteGroup, roles(Roles.WORKER, Roles.ADMIN))

path(":group-id") {

get(GroupController.getGroup, roles(Roles.WORKER, Roles.ADMIN))

put(GroupController.editGroup, roles(Roles.WORKER, Roles.ADMIN))

delete(GroupController.deleteGroup, roles(Roles.WORKER, Roles.ADMIN))

}

}

path(Path.LESSONS) {

get(LessonController.fetchAllLessons, roles(Roles.WORKER, Roles.ADMIN))

post(LessonController.addLesson, roles(Roles.WORKER, Roles.ADMIN))

post("/check", LessonController.canDeleteLesson, roles(Roles.WORKER, Roles.ADMIN))

path(":lesson-id") {

get(LessonController.getLesson, roles(Roles.WORKER, Roles.ADMIN))

put(LessonController.editLesson, roles(Roles.WORKER, Roles.ADMIN))

delete(LessonController.deleteLesson, roles(Roles.WORKER, Roles.ADMIN))

}

}

path(Path.USERS) {

get(UserController.fetchAllUsers, roles(Roles.ADMIN))

post(UserController.addUser, roles(Roles.ADMIN))

path(":username") {

get(UserController.getUser, roles(Roles.ADMIN))

put(UserController.editUser, roles(Roles.ADMIN))

delete(UserController.deleteUser, roles(Roles.ADMIN))

}

path("isUsernameTaken") {

post(UserController.isUsernameTaken, roles(Roles.ADMIN))

}

}

path("changePassword") {

post(UserController.updatePassword, roles(Roles.WORKER, Roles.ADMIN))

}

path(Path.AUTHORIZE) {

post(UserController.authorizeCallback, roles(Roles.ANYONE))

}

path(Path.USER\_API) {

path(Path.TEACHERS) {

get(TeacherController.fetchAllTeachers, roles(Roles.ANYONE))

}

path(Path.LESSONS) {

get(LessonController.fetchAllLessonsForUser, roles(Roles.ANYONE))

}

path(Path.GROUPS) {

get(GroupController.fetchAllGroupsUser, roles(Roles.ANYONE))

}

}

}

//responses

app.get(Path.LOGIN\_REQUIRED, ResponseController.loginRequiredError)

app.get(Path.SUCCESS, ResponseController.success)

app.get(Path.ERROR, ResponseController.error)

app.after(AddHeader.addJSONHeader)

app.after(AddHeader.addCrossOriginHeader)

}

object Main {

val disciplineInteractor = DisciplineInteractorNew(DB.conn)

val teacherInteractor = TeacherInteractorNew(DB.conn)

val lessonInteractor = LessonInteractorNew(DB.conn)

val officeInteractor = OfficeInteractorNew(DB.conn)

val groupInteractor = GroupInteractorNew(DB.conn)

val userInteractor = UserInteractor(DB.conn)

}

enum class Roles : Role {

ANYONE, WORKER, ADMIN

}

class DB {

companion object {

val conn by lazy {

var conn = DriverManager.getConnection(DATABASE\_URL, DATABASE\_USER, DATABASE\_PASSWORD)

conn.createStatement().apply {

val ans = executeQuery("SELECT datname FROM pg\_catalog.pg\_database WHERE datname = '$DATABASE\_NAME';")

if(!ans.next()) {

conn.createStatement().apply {

executeUpdate("CREATE DATABASE $DATABASE\_NAME;")

}

}

}

conn.close()

conn = DriverManager.getConnection(DATABASE\_URL + DATABASE\_NAME, DATABASE\_USER, DATABASE\_PASSWORD)

var script = ""

FileReader(File("init.sql")).use {

script = it.readText()

}

conn.createStatement().apply {

executeUpdate(script)

}

conn

}

}

}

package common

import utils.GetAllParams

import java.sql.Connection

import java.sql.PreparedStatement

import java.sql.ResultSet

abstract class Interactor<T : Model<T>, FilterClass : Filter>(private val connection: Connection) {

abstract val tableParams: String

abstract val tableName: String

abstract val insertQuery: String

abstract val editQuery: String

abstract val referenceName: String

abstract fun addParamsToQueryForSearch(st: PreparedStatement, filter: FilterClass)

abstract fun getSearchPathQuery(filter: FilterClass): String

fun getAll(params: GetAllParams?, filter: FilterClass?): List<T> {

val searchQuery = if (filter == null) "" else getSearchPathQuery(filter)

val query = when (params) {

null -> "SELECT $tableParams FROM $tableName $searchQuery"

else -> "SELECT $tableParams FROM $tableName $searchQuery ORDER BY ${params.sortBy.toDBName()} ${params.order.name} LIMIT ? OFFSET ?"

}

val st = connection.prepareStatement(query)

if (filter != null) {

addParamsToQueryForSearch(st, filter)

}

val paramInSearchCount = if (filter != null) {

getSearchPathQuery(filter).count { it == '?' }

} else {

0

}

when {

params != null -> {

st.setInt(paramInSearchCount + 1, params.to - params.from)

st.setInt(paramInSearchCount + 2, params.from)

}

}

val res = st.executeQuery()

val ans = mutableListOf<T>()

while (res.next()) {

ans.add(res.getObject())

}

return ans

}

fun getAllCountNoPagination(filter: FilterClass?): Int {

if (filter == null) return getCount()

val query = "SELECT COUNT(\*) FROM $tableName ${getSearchPathQuery(filter)}"

val st = connection.prepareStatement(query)

addParamsToQueryForSearch(st, filter)

val res = st.executeQuery()

return if (res.next()) {

res.getInt(1)

} else {

0

}

}

open fun String.toDBName(): String {

return this

}

fun get(id: Int): T {

val st = connection.prepareStatement("SELECT $tableParams FROM $tableName WHERE id = ?")

st.setInt(1, id)

val res = st.executeQuery()

res.next()

return res.getObject()

}

abstract fun addParamsToQueryForInsert(st: PreparedStatement, obj: T)

fun add(obj: T): T {

val st = connection.prepareStatement("$insertQuery RETURNING $tableParams")

addParamsToQueryForInsert(st, obj.trimmed())

val res = st.executeQuery()

res.next()

return res.getObject()

}

abstract fun addParamsToQueryForEdit(st: PreparedStatement, obj: T)

fun edit(obj: T): T {

val st = connection.prepareStatement("$editQuery RETURNING $tableParams")

addParamsToQueryForEdit(st, obj.trimmed())

val res = st.executeQuery()

res.next()

return res.getObject()

}

fun delete(id: Int): T {

val st = connection.prepareStatement("DELETE FROM $tableName WHERE id = ? RETURNING $tableParams")

st.setInt(1, id)

val res = st.executeQuery()

res.next()

return res.getObject()

}

private fun getCount(): Int {

val st = connection.createStatement()

val res = st.executeQuery("SELECT COUNT(\*) FROM $tableName")

res.next()

return res.getInt(1)

}

fun canBeDeleted(ids: List<Int>): Boolean {

val st = connection.prepareStatement("SELECT COUNT(\*) FROM lesson WHERE $referenceName = ANY (?)")

val arr = connection.createArrayOf("integer", ids.toTypedArray())

st.setArray(1, arr)

val res = st.executeQuery()

res.next()

return res.getInt(1) == 0

}

abstract fun ResultSet.getObject(): T

}

class MainActivity : AppCompatActivity(), MainView {

private val fragmentTags = listOf(TeachersFragment.TAG, ScheduleFragment.TAG)

private val dialogs = MainDialogs(this)

private val presenter = MainPresenter(this)

private lateinit var bottomSheetBehavior: BottomSheetBehavior<LinearLayout>

@Volatile

private var canChangeTitle = true

private var lastSavedTitle = ""

override fun onCreate(savedInstanceState: Bundle?) {

Toothpick.inject(dialogs, Toothpick.openScope("APP"))

Toothpick.inject(presenter, Toothpick.openScope("APP"))

super.onCreate(savedInstanceState)

setContentView(R.layout.activity\_main)

bottomSheetBehavior = BottomSheetBehavior.from(bottomSheet)

bottomSheetBehavior.addBottomSheetCallback(object :

BottomSheetBehavior.BottomSheetCallback() {

override fun onSlide(bottomSheet: View, slideOffset: Float) {}

override fun onStateChanged(bottomSheet: View, newState: Int) {

}

})

bottom\_navigation.setOnNavigationItemSelectedListener { item ->

when (item.itemId) {

R.id.lessons -> {

supportFragmentManager.openFragment(ScheduleFragment.TAG)

item.isChecked = true

}

R.id.teachers -> {

supportFragmentManager.openFragment(TeachersFragment.TAG)

item.isChecked = true

}

R.id.settings -> {

bottomSheetBehavior.state = BottomSheetBehavior.STATE\_EXPANDED

}

}

false

}

bottom\_navigation.selectedItemId = R.id.lessons

presenter.loadCurrentGroupAndUpdateTitle()

}

private fun FragmentManager.openFragment(tag: String) {

when (tag) {

TeachersFragment.TAG -> title = "Преподаватели"

ScheduleFragment.TAG -> title = lastSavedTitle

}

canChangeTitle = ScheduleFragment.TAG == tag

val transaction = beginTransaction()

val foundFragment = findFragmentByTag(tag)

if (foundFragment == null) {

val fragment = when (tag) {

TeachersFragment.TAG -> TeachersFragment()

ScheduleFragment.TAG -> ScheduleFragment()

else -> throw IllegalArgumentException()

}

transaction.add(R.id.fragmentHolder, fragment, tag)

} else {

transaction.show(foundFragment)

}

for (fragmentTag in fragmentTags) {

if (fragmentTag != tag) {

findFragmentByTag(fragmentTag)?.let { transaction.hide(it) }

}

}

transaction.commitNow()

}

override fun getDialogs(): MainDialogs {

return dialogs

}

override fun setTitle(text: String) {

lastSavedTitle = text

if(canChangeTitle) {

title = text

}

}

override fun hideBottomSheet() {

bottomSheetBehavior.state = BottomSheetBehavior.STATE\_COLLAPSED

}

} fun getSchedule(groupName: String): Single<Pair<List<Schedule>, Boolean>> {

return Single.create<List<Schedule>> { emitter ->

val date = Calendar.getInstance()

if (date.get(Calendar.DAY\_OF\_WEEK) == Calendar.MONDAY) {

date.add(Calendar.DAY\_OF\_MONTH, -2)

} else {

date.add(Calendar.DAY\_OF\_MONTH, -1)

}

date.set(Calendar.HOUR\_OF\_DAY, 0)

date.set(Calendar.MINUTE, 0)

date.set(Calendar.SECOND, 0)

date.set(Calendar.MILLISECOND, 0)

val request = Request.Builder()

.url("${Settings.URL}/user\_api/lessons?from=${date.timeInMillis}")

.get()

.build()

okHttpClient.newCall(request).enqueue(object : Callback {

override fun onFailure(call: Call, e: IOException) {

emitter.onError(e)

}

override fun onResponse(call: Call, response: Response) {

val responseString = response.body?.string() ?: "[]"

val res = gson.fromJson(responseString, Array<Schedule>::class.java).toList()

emitter.onSuccess(res)

}

})

}

.map { list ->

list.filter { it.group == groupName }

}

.doOnSuccess { scheduleDao.replaceAllSchedule(it) }

.map { it to false }

.onErrorResumeNext {

loadFromCache()

.map { it to true }

}

.subscribeOn(Schedulers.io())

} fun getTeachers(): Single<Pair<List<Teacher>, Boolean>> {

return Single.create<List<Teacher>> { emitter ->

val request = Request.Builder()

.url("${Settings.URL}/user\_api/teachers")

.get()

.build()

okHttpClient.newCall(request).enqueue(object : Callback {

override fun onFailure(call: Call, e: IOException) {

emitter.onError(e)

}

override fun onResponse(call: Call, response: Response) {

val responseString = response.body?.string() ?: "[]"

val res =

Gson().fromJson(responseString, Array<Teacher>::class.java).toList()

emitter.onSuccess(res)

}

})

}

.doOnSuccess {

teacherDao.replaceAllTeachers(it)

}

.map { it to false }

.onErrorResumeNext {

getTeachersFromCache()

.map { it to true }

}

.subscribeOn(Schedulers.io())

}