Программа сетевой акселерации (ускорения) научно-технических, инновационных и предпринимательских инициатив молодежи, направленных на социально- экономическое развитие города Москвы

СОЗДАНИЕ ANDROID-ПРИЛОЖЕНИЯ «TEACHER’S ASSISTANT»

Направление: Интеллектуальные

информационные системы и

математическое моделирование

Исполнитель: И.С. Марчук 11 класс ГБОУ СОШ № 1852 г. Москвы

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Научный руководитель:

А.П. Набережная Учитель

ГБОУ СОШ № 1852 г. Москвы

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_

2017 г.

# СОДЕРЖАНИЕ:

Аннотация……………………………………………………………………………...3

Введение………………………………………………………………………………..4

Основная часть…………………………………………………………………………5

1 Анализ и уточнение требований, построение функциональных

спецификаций………………………………………………………………………….5

1.1 Анализ задания, выбор технологии, языка и среды разработки….….5

1.2 Разработка диаграмм вариантов использования приложения……….5

2 Определение структуры программного продукта………………………………..19

2.1 Анализ процесса обработки информации и выбор структур данных для

её хранения……………………………………………………………………………19

2.2 Построение структурной схемы программного продукта………………28

3 Разработка интерфейса………………………………………………………….…30

3.1 Построение диаграммы состояний интерфейса…………………………30

3.2 Проектирование форм ввода вывода информации………………………32

3.3Локализация…………………………………………………………………65

4 Выбор стратегии тестирования и разработки тестов………………………….….66

Заключение…………………………………………………………………………..…66

Список использованных источников………………………………………………... 67

# АННОТАЦИЯ

Представленная нами работа заключается в создании Аndroid приложения «Teacher’s assistant». Идея создания данного приложения была подсказана нам потенциальными пользователями – школьными учителями. Им требовался удобный способ для того, чтобы отмечать активность учеников на уроке.

Данное приложение предназначено для фиксации количества ответов учащегося. Подобная функция не имеет себе аналогов на медиа рынке.

Помимо этой функции приложение содержит ряд других, позволяющих с полным на то правом назвать его «Помощник учителя». Это функции создания расписания, виртуальных классов и многие другие.

Данное приложение было разработано для школьных учителей по их просьбе, протестировано ими и признано функционально удобным.

# ВВЕДЕНИЕ

Так как основная текущая деятельность – школьное обучение, идея приложения естественным образом родилась из школьной жизни.

Ежегодно учителя проводят открытые уроки в школе. Мой учитель математики также давал подобное мероприятие. Урок проходил в динамичной форме, активно. Важным условием урока, как стало понятно от реакции гостей, было использование новых технологий. Надо сказать, что мои одноклассники сняли видеоролики по заданной теме и выложили их в облако для оперативного использования на уроке. Все было весьма динамично и эффектно. Однако, при правильном ответе наш учитель раздавал элементы поздравительной открытки. Тем самым в конце урока каждому из учеников надо было сообщить учителю: сколько кусочков открытки у него на руках. В итоге, когда учитель понял, что зафиксировать информацию от всех за короткий промежуток времени ему не удасться, то предложил выйти и написать на доске каждому учащемуся его фамилию и количество частей открытки. Здесь возникла толкотня у доски, количество мела не соответствовало количеству желающих отписаться и пойти на перемену.

И еще один момент на уроке заставил меня задуматься о том, как помочь учителю. В момент правильных ответов, педагог шел к ученику чтобы отдать кусок открытки, символизирующий балл за правильный ответ. Если учащиеся, ответившие верно находились в разных концах кабинета, учитель стремительно менял направление и спешил к тому, кто ответил верно. Так как, как стало понятно, для педагога было важно провести урок динамично, то у меня закралось естественное подозрение о том, сколько километров набегал учитель за урок?

Современные технологии, активный темп урока и кусочки открыток, символизирующие верный ответ, не вязались в единое целое.

Поэтому естественным образом возникло желание написать приложение, которое могло бы одним касанием фиксировать верный ответ ученика, удачное решение или корректировку на уроке и т.п.

Довести контроль за верными ответами на уроке до автоматизма и преследуя цель, чтобы каждый ученик был оценен за идею, за правильное высказывание и не был забыт «загруженным» педагогом – вот что побудило меня заняться мобильным приложением для учителя.

# ОСНОВНАЯ ЧАСТЬ

**1 Анализ и уточнение требований, построение функциональных спецификаций**

**1.1 Анализ задания, выбор технологии, языка и среды разработки**

Для разработки приложения была выбрана библиотеки Android SDK, среду Android Studio и язык Java 7. Java – объектно-ориентированный язык программирования, который составляет основную часть разработки под OS Android. Так, как программа должна иметь привычный интерфейс для диалога с пользователем, выбор остановился на среде разработки Android Studio. Компания Google, являющаяся владельцем платформы Android, непрерывно поддерживает эту среду, выпуская новые библиотеки и плагины. Библиотеки Android SDK обладают огромным потенциалом и предоставляют разработчику широкий спектр инструментов для разработки. Основным достоинством Android Studio является то, что она используется в качестве посредника между пользователем, консолью ОС и файлами проекта, переводя это в обработку событий, что сильно облегчает работу разработчика, освобождая его от прямого контакта с сообщениями системы. Ещё одним достоинством Android Studio является возможность визуализированной работы с пользовательским интерфейсом, наличие удобных средств уведомления и логирования что позволяет увидеть результаты ещё на стадии разработки.

При проектировании использовался объектно-ориентированный подход, это позволяет разделить части приложения на отдельные составляющие со своими характеристиками и методами, особенно это касается разработки под Android, там каждая активность представляет собой отдельную часть приложения и для их создания и связи между собой необходимо использование ООП.

**1.2 Разработка диаграмм вариантов использования приложения**

При создании приложения упор ставился в первую очередь на удобство взаимодействия с интерфейсом.

При запуске приложения перед пользователем отображается активность главного меню. Из неё, он может перейти в активность с текущим уроком, активность с расписанием, активность со списком кабинетов, активность со списком классов и учеников, и активность настроек.

Таблица 1 – Описание для варианта использования приложения в режиме “Главное меню

|  |  |
| --- | --- |
| Название варианта | Приложение в режиме “Главное меню” |
| Цель | Навигация пользователя по приложению |
| Действующие лица | Пользователь |
| Краткое описание | Решение задачи предполагает предоставление пользователю быстрого доступа к элементам приложения |
| Тип варианта | Основной |

Таблица 2 – Типичный ход событий в режиме “Главное меню – текущий урок”

|  |  |
| --- | --- |
| Действия пользователя | Отклик приложения |
| 1 Пользователь выбирает пункт «текущий урок» | 2 Приложение ищет в базе данных урок совпадающий по времени с текущим временем и находя осуществляет закрытие текущей активности и переход на активность урока передавая ей ссылку на этот урок в базе данных |

Альтернатива

2откуда тут двойка? Если в базе данных нет урока совпадающего по времени с текущим временем, то приложение показывает пользователю всплывающее уведомление с текстом: «на текущий момент нет уроков в расписании».

Таблица 3 – Типичный ход событий в режиме “Главное меню – моё расписание”

|  |  |
| --- | --- |
| Действия пользователя | Отклик приложения |
| 1 Пользователь выбирает пункт «моё расписание» | 2 Приложение осуществляет закрытие текущей активности и переход на активность расписания на месяц |

Таблица 4 – Типичный ход событий в режиме “Главное меню – мои кабинеты”

|  |  |
| --- | --- |
| Действия пользователя | Отклик приложения |
| 1 Пользователь выбирает пункт «мои кабинеты» | 2 Приложение осуществляет закрытие текущей активности и переход на активность со списком кабинетов |

Таблица 5 – Типичный ход событий в режиме “Главное меню – мои ученики”

|  |  |
| --- | --- |
| Действия пользователя | Отклик приложения |
| 1 Пользователь выбирает пункт «мои ученики» | 2 Приложение осуществляет закрытие текущей активности и переход на активность со списком классов |

Таблица 6 – Типичный ход событий в режиме “Главное меню – настройки”

|  |  |
| --- | --- |
| Действия пользователя | Отклик приложения |
| 1 Пользователь выбирает пункт «настройки» | 2 Приложение осуществляет закрытие текущей активности и переход на активность настроек приложения |

После перехода в активность с классами, из базы данных в список выводятся классы уже созданные пользователем.

Рассмотрим создание классов в базе данных.

Таблица 7 – Описание для варианта использования приложения в режиме создания класса

Отрегулируй большие и маленькие буквы

|  |  |
| --- | --- |
| Название варианта | Приложение в режиме “ Создание класса ” |
| Цель | Создание класса |
| Действующие лица | Пользователь |
| Краткое описание | Решение задачи предполагает создание класса и сохранение его в базе данных |
| Тип варианта | Основной |

Таблица 8 – Типичный ход событий в режиме создания класса

|  |  |
| --- | --- |
| Действия пользователя | Отклик приложения |
| 1 Пользователь выбирает пункт «мои ученики» | 2 Приложение осуществляет закрытие текущей активности и переход на активность со списком классов |
| 3 Пользователь выбирает пункт «создание» | 4 Приложение показывает диалог с текстовым полем для имени класса и двумя кнопками «сохранить» и «отмена» |
| 5 Пользователь вводит название класса (например: «7 'Б'») и нажимает кнопку сохранить | 6 Приложение создаёт класс в базе данных выводит его в список классов и завершает диалог |

Альтернатива

5 Если пользователь нажал «отмена» то 4 пункт не срабатывает

Дополнительно

1.Необходимо обеспечить возможность выхода из варианта на любом этапе

Рассмотрим редактирование классов в базе данных.

Таблица 9 – Описание для варианта использования приложения в режиме редактирования класса

|  |  |
| --- | --- |
| Название варианта | Приложение в режиме “ редактирование классов ” |
| Цель | Редактирование класса |
| Действующие лица | пользователь |
| Краткое описание | Решение задачи предполагает редактирование класса и сохранение изменений в базе данных |
| Тип варианта | Основной |

Таблица 10 – Типичный ход событий в режиме редактирования класса

|  |  |
| --- | --- |
| Действия пользователя | Отклик приложения |
| 1 Пользователь выбирает пункт «мои ученики» | 2 Приложение осуществляет закрытие текущей активности и переход на активность со списком классов |
| 3 Пользователь удерживает нужный класс | 4 Приложение выводит флажки выбора возле каждого класса в списке (сразу пометив выбранный пользователем класс) и дополнительное меню с кнопками удалить и изменить |
| 5 Пользователь (если нужно) ставит флажки у других классов, а затем нажимает кнопку редактировать | 6 Приложение выводит диалоговое окно с текстовым полем для имени класса (если флажками выбран только один класс, то выводит его имя в текстовое поле) и две кнопки «сохранить» и «отмена» |
| 7 Пользователь редактирует имя класса и нажимает кнопку «сохранить» | 8 Приложение сохраняет изменения в базе данных, обновляет имя класса в списке, убирает из списка флажки выбора и завершает диалог |

Альтернатива

5 пользователь может нажать кнопку назад, это уберёт флажки выбора и дополнительное меню.

7 Если пользователь нажал «отмена» то идет откат ко 2 пункту.

Дополнительно

1.Необходимо обеспечить возможность выхода из варианта на любом этапе.

Рассмотрим удаление классов в базе данных.

Таблица 11 – Описание для варианта использования приложения в режиме удаления класса

|  |  |
| --- | --- |
| Название варианта | Приложение в режиме “ удаление классов ” |
| Цель | удаление класса |
| Действующие лица | пользователь |
| Краткое описание | Решение задачи предполагает удаление класса из базы данных |
| Тип варианта | Основной |

Таблица 12 – Типичный ход событий в режиме удаления класса

|  |  |
| --- | --- |
| Действия пользователя | Отклик приложения |
| 1 Пользователь выбирает пункт «мои ученики» | 2 Приложение осуществляет закрытие текущей активности и переход на активность со списком классов |
| 3 Пользователь удерживает нужный класс | 4 Приложение выводит флажки выбора возле каждого класса в списке (сразу пометив выбранный пользователем класс) и дополнительное меню с кнопками удалить и изменить |
| 5 Пользователь (если нужно) ставит флажки у других классов, а затем нажимает кнопку удалить | 6 приложение удаляет класс из базы данных, убирает флажки выбора и дополнительное меню и убирает удалённый класс из списка. |

Альтернатива

5 пользователь может нажать кнопку назад, это уберёт флажки выбора и дополнительное меню.

Дополнительно

1.Необходимо обеспечить возможность выхода из варианта на любом этапе.

Кому необходимо чего необходимо непонятно

При нажатии пользователем на класс приложение закрывает текущую активность и запускает активность с учениками, передав ей идентификатор выбранного пользователем класса. Активность выводит список уже созданных пользователем учеников, принадлежащих этому классу, из базы данных в список.

Ход событий создания, редактирования и удаления учеников аналогичен ходу событий у классов, за исключением диалога, где высвечивается поле не только имени, но и фамилии.

После хода событий описанного в таблице 4, активность выводит из базы данных в список уже созданные пользователем кабинеты.

Ход событий создания, редактирования и удаления кабинетов аналогичен, ходу событий у классов.

Пользователь может редактировать расстановку парт в созданном кабинете. Рассмотрим создание двухместных парт в кабинетах.

Таблица 11 – Описание для варианта использования приложения в режиме создания парты

|  |  |
| --- | --- |
| Название варианта | Приложение в режиме “создание парты” |
| Цель | создание парты |
| Действующие лица | пользователь |
| Краткое описание | Решение задачи предполагает создание в базе данных парты принадлежащей выбранному классу |
| Тип варианта | Основной |

Таблица 12 – Типичный ход событий в режиме создания парты

|  |  |
| --- | --- |
| Действия пользователя | Отклик приложения |
| 1 Пользователь выбирает пункт «мои кабинеты» | 2 Приложение осуществляет закрытие текущей активности и переход на активность со списком кабинетов |
| 3 Пользователь выбирает из списка уже созданных необходимый кабинет | 4Приложение осуществляет закрытие текущей активности, переход на активность редактора кабинетов с идентификатором выбранного кабинета и  выводит в активность уже созданные парты. |
| 5 Пользователь нажимает кнопку «добавить» | 6 Приложение создаёт двухместную парту в базе данных и выводит ее в центре экрана |
| 7 Пользователь касается парты на экране | 8 Приложение заменяет кнопку «добавить» на кнопку «удалить» |
| 9 Пользователь передвигает пальцем по экрану парту, размещая её так как ему нужно затем убирает палец с экрана | 10 Приложение сохраняет координаты парты в базе данных и заменяет кнопку «удалить» на кнопку «добавить» |
| 11 Пользователь нажимает кнопку назад | 12 Приложение осуществляет закрытие текущей активности и переход на активность кабинетов. |
| 13 Пользователь нажимает кнопку назад | 14 Приложение осуществляет закрытие текущей активности и переход на активность главного меню. |

Альтернатива

5 пользователь может нажать кнопку назад, произойдёт переход к пункту 9

Дополнительно

1.Необходимо обеспечить возможность выхода из варианта на любом этапе.

Рассмотрим удаление парт в кабинетах.

Таблица 13 – Описание для варианта использования приложения в режиме удаления парты

|  |  |
| --- | --- |
| Название варианта | Приложение в режиме “удаление парты” |
| Цель | удаление парты |
| Действующие лица | пользователь |
| Краткое описание | Решение задачи предполагает удаление из базы данных выбранной пользователем парты принадлежащей выбранному классу |
| Тип варианта | Основной |

Таблица 14 – Типичный ход событий в режиме удаления парты

|  |  |
| --- | --- |
| Действия пользователя | Отклик приложения |
| 1 Пользователь выбирает пункт «мои кабинеты» | 2 Приложение осуществляет закрытие текущей активности и переход на активность со списком кабинетов |
| 3 Пользователь выбирает из списка уже созданных необходимый кабинет | 4 Приложение осуществляет закрытие текущей активности, переход на активность редактора кабинетов с идентификатором выбранного кабинета и  выводит в активность уже созданные парты. |
| 5 Пользователь касается парты на экране | 6 Приложение заменяет кнопку «добавить» на кнопку «удалить» |
| 7 Пользователь передвигает пальцем парту к кнопке удалить | 8 Приложение удаляет парту из базы данных. |
| 9 Пользователь убирает палец с экрана | 10 Приложение заменяет кнопку «удалить» на кнопку «добавить» |
| 11 Пользователь нажимает кнопку назад | 12 Приложение осуществляет закрытие текущей активности и переход на активность кабинетов. |
| 13 Пользователь нажимает кнопку назад | 14 Приложение осуществляет закрытие текущей активности и переход на активность главного меню. |

Альтернатива

5 пользователь может нажать кнопку назад, произойдёт переход к пункту 11

Дополнительно

1.Необходимо обеспечить возможность выхода из варианта на любом этапе.

После хода событий описанного в таблице 3 активность выводит календарь на текущий месяц.

Рассмотрим создание урока

Таблица 15 – Описание для варианта использования приложения в режиме создания урока

|  |  |
| --- | --- |
| Название варианта | Приложение в режиме “ создание урока ” |
| Цель | создание урока |
| Действующие лица | пользователь |
| Краткое описание | Решение задачи предполагает создание урока в расписании |
| Тип варианта | Основной |

Таблица 16 – Типичный ход событий в режиме создания урока

|  |  |
| --- | --- |
| Действия пользователя | Отклик приложения |
| 1 Пользователь выбирает пункт «моё расписание» | 2 Приложение осуществляет закрытие текущей активности и переход на активность расписания.  Выводит календарь на текущий месяц выделяя в нём текущий день и те дни когда есть уроки и выводит кнопки «вперёд» и «назад» |
| 3 пользователь выбирает нужный ему день | 4 Приложение закрывает активность месяца и открывает активность дня передавая в неё выбранную дату. В активности дня выводятся все уже созданные в этом расписании уроки |
| 5 Пользователь выбирает пустое поле в расписании дня | 6 Приложение закрывает активность дня и открывает активность редактирования урока передавая в неё выбранное время |
| 7 Пользователь выбирает класс и кабинет в выпадающих списках. | 8 Если ученики в выбранном классе рассажены в выбранном кабинете приложение выводит в строку состояния: «ученики рассажены» иначе «ученики не рассажены!».  В выпадающий список предмета урока добавляются все уроки привязанные к выбранному классу и кнопка добавить урок |
| 9 Пользователь выбирает урок из списка и время урока и нажимает кнопку сохранить | 10 Приложение сохраняет урок в базе данных, закрывает текущую активность и переходит в активность расписания дня |
| 11 Пользователь нажимает кнопку назад | 12 Приложение осуществляет закрытие текущей активности и переход на активность расписания месяца. |
| 13 Пользователь нажимает кнопку назад | 14 Приложение осуществляет закрытие текущей активности и переход на активность главного меню. |

Альтернатива

3 пользователь может нажать кнопку назад, произойдёт переход к пункту 13

5 пользователь может нажать кнопку назад, произойдёт переход к пункту 11

7 пользователь может нажать кнопку отмена, произойдёт переход к пункту 11

Дополнительно

1.Необходимо обеспечить возможность выхода из варианта на любом этапе.

Редактирование урока происходит аналогичным образом, но в 5 пункте пользователь выбирает урок который хочет редактировать, долго удерживая его.

Чтобы рассадить учеников пользователю необходимо в редакторе урока, выбрав класс и кабинет, нажать кнопку «рассадить».

Рассмотрим редактор рассадки

Таблица 18 – Описание для варианта использования приложения в режиме смены места ученика

|  |  |
| --- | --- |
| Название варианта | Приложение в режиме “смены места ученика” |
| Цель | Рассадить учеников |
| Действующие лица | Пользователь |
| Краткое описание | Решение задачи предполагает рассадить учеников выбранного класса в выбранном кабинете |
| Тип варианта | Основной |

Таблица 19 – Типичный ход событий в режиме смены места ученика

|  |  |
| --- | --- |
| Действия пользователя | Отклик приложения |
|  | 0 Таблица 16 пункт 8 |
| 1 Пользователь выбирает пункт «рассадить» | 2 Приложение осуществляет закрытие текущей активности и переход на активность редактора рассадки, с передачей в неё идентификаторов класса и кабинета.  Выводит парты в этом кабинете и рассаженных учеников. |
| 3 пользователь выбирает рассаженного ученика | 4 Приложение удаляет из базы данных зависимость ученик-место и удаляет ученика с парты на экране |
| 5 Пользователь выбирает пустое место на парте | 6 Приложение запускает диалог, в котором выводится список не рассаженных учеников. |
| 7 Пользователь выбирает ученика | 8 Приложение создаёт зависимость «ученик-место», отображает ученика на выбранном месте и закрывает диалог |
| 11 Пользователь нажимает кнопку назад | 12 Приложение осуществляет закрытие текущей активности и переход на активность расписания дня. |
| 13 Пользователь нажимает кнопку назад | 14 Приложение осуществляет закрытие текущей активности и переход на активность расписания месяца. |
| 15 Пользователь нажимает кнопку назад | 16 Приложение осуществляет закрытие текущей активности и переход на активность главного меню. |

Альтернатива

3 пользователь может нажать кнопку назад, произойдёт переход к пункту 11

5 пользователь может нажать кнопку назад, произойдёт переход к пункту 11

7 пользователь может нажать на область вне диалога, приложение завершит диалог, произойдёт переход к пункту 11

Дополнительно

1.Необходимо обеспечить возможность выхода из варианта на любом этапе.

После того как все параметры введены пользователь может начать урок, сделать это можно двумя способами:

1 способ описан в Таблице 2;

2 способ (с описанием самого урока):

Таблица 15 – Описание для варианта использования приложения в режиме проведения урока

|  |  |
| --- | --- |
| Название варианта | Приложение в режиме “ проведение урока ” |
| Цель | Провести урок |
| Действующие лица | Пользователь |
| Краткое описание | Решение задачи предполагает проведение урока |
| Тип варианта | Основной |

Таблица 16 – Типичный ход событий в режиме проведения урока

|  |  |
| --- | --- |
| Действия пользователя | Отклик приложения |
| 1 Пользователь выбирает пункт «моё расписание» | 2 Приложение осуществляет закрытие текущей активности и переход на активность расписания.  Выводит календарь на текущий месяц выделяя в нём текущий день и те дни когда есть уроки и выводит кнопки «вперёд» и «назад» |
| 3 пользователь выбирает нужный ему день, где есть урок | 4 Приложение закрывает активность месяца и открывает активность дня, передавая в неё выбранную дату. В активности дня выводятся все уже созданные в этом расписании уроки |
| 5 Пользователь выбирает урок | 6 Приложение закрывает активность дня и открывает активность урока передавая в неё идентификатор выбранного урока.  В активности приложение выводит парты выбранного кабинета (информация о кабинете содержится в уроке в базе данных) и на них иконки «без оценки», учеников выбранного класса(информация о классе содержится в уроке в базе данных). |
| 7 Пользователь удерживает ученика | 8 Приложение выводит контекстное меню со списком оценок от 1 до 5 |
| 9 Пользователь выбирает оценку, которую хочет поставить ученику | 10 Приложение выводит оценку на иконке ученика, заменяя иконку «без оценки» на иконку с оценкой |
| 11 Пользователь удерживает ту же иконку ученика | 12 Приложение выводит контекстное меню со списком оценок от «1» до «5»,кроме выбранной ранее, пункт «нет оценки» и пункт «новая оценка» |
| 13 Пользователь нажимает пункт новая оценка | 14 Приложение выводит предыдущую оценку в верхнем углу иконки ученика и ставит иконку «без оценки» |
| 15 Пользователь удерживает иконку ученика | 16 Приложение выводит контекстное меню со списком оценок от 1 до 5 |
| 17 Пользователь выбирает оценку которую хочет поставить ученику | 18 Приложение выводит оценку на иконке ученика, заменяя иконку «без оценки» на иконку с оценкой, не меняя оценки в углу |
| 19 Пользователь нажимает кнопку закончить урок | 20 Приложение закрывает активность урока и открывает активность итогов урока передавая в неё оценки и учеников.  Вводится список учеников и напротив оценок. |
| 21 Пользователь нажимает пункт меню «сохранить» | 22 Приложение сохраняет оценки за этот урок в базе данных, закрывает активность итогов урока и открывает активность дня. В активности дня выводятся все уже созданные в этом расписании уроки. |
| 23 Пользователь нажимает кнопку «назад» | 24 Приложение осуществляет закрытие текущей активности и переход на активность расписания на месяц.  Выводит календарь на текущий месяц, выделяя в нём текущий день и те дни когда есть уроки и выводит кнопки «вперёд» и «назад» |
| 25 Пользователь нажимает кнопку назад | 26 Приложение закрывает активность месяца и открывает активность главного меню. |

Альтернатива

3 пользователь нажимает кнопку «назад», произойдёт переход к пункту 25

5,7,11,15 пользователь нажимает кнопку «назад», произойдёт переход к пункту 23

9,13,17 пользователь нажимает область вне диалога или кнопку «назад», произойдёт переход к пункту 11

21 пользователь нажимает кнопку назад, произойдёт переход к пункту 23 без сохранения оценок

Дополнительно

1.Необходимо обеспечить возможность выхода из варианта на любом этапе.

2.На уроке можно поставить ученику максимум 3 оценки, которые можно отредактировать, нажимая на ученика или вызывая его контекстное меню

**2 Определение структуры программного продукта**

**2.1 Анализ процесса обработки информации и выбор структур данных для её хранения**

Для хранения данных было решено использовать реляционную базу данных(далее БД).

Одним из примеров реализации реляционных БД является встраиваемая библиотека SQLite.

OS Android полностью поддерживает SQLite, которая позволяет создавать базы данных и хранить в них информацию, а также выполнять сложные запросы к ним.

Для работы с базой данных необходимо создать класс, унаследованный от SQLiteOpenHelper, который нужен для первой инициализации БД на устройстве, обновления и создания структур её таблиц.

import android.content.ContentValues;

import android.content.Context;

import android.database.Cursor;

import android.database.sqlite.SQLiteDatabase;

import android.database.sqlite.SQLiteOpenHelper;

import android.util.Log;

import java.util.ArrayList;

import java.util.Arrays;

import java.util.Calendar;

import java.util.Date;

public class DataBaseOpenHelper extends SQLiteOpenHelper {

private static final int DB\_VERSION = 4;

public DataBaseOpenHelper(Context context) {

super(context, SchoolContract.DB\_NAME, null, DB\_VERSION);

}

@Override

public void onCreate(SQLiteDatabase db) {

updateDatabase(db, 0, 1);

}

@Override

public void onUpgrade(SQLiteDatabase db,int oldVersion,intnewVersion){

updateDatabase(db, oldVersion, newVersion);

}

@Override

public void onOpen(SQLiteDatabase db) {

//нужен для удаления каскадом

db.execSQL("PRAGMA foreign\_keys = ON");

super.onOpen(db);

}

private void updateDatabase(SQLiteDatabase db, int oldVersion, int

newVersion) {

Log.i("DataBaseOpenHelper", "updateDatabase old=" + oldVersion +

"new=" + newVersion);

if (oldVersion < 5) {

//если база версии 5 и выше, то она не запустит этот код

db.execSQL("PRAGMA foreign\_keys = OFF");

db.execSQL("DROP TABLE IF EXISTS " +

SchoolContract.TableCabinets.NAME\_TABLE\_CABINETS + ";");

db.execSQL("DROP TABLE IF EXISTS " +

SchoolContract.TableDesks.NAME\_TABLE\_DESKS + ";");

db.execSQL("DROP TABLE IF EXISTS " +

SchoolContract.TablePlaces.NAME\_TABLE\_PLACES + ";");

db.execSQL("DROP TABLE IF EXISTS " +

SchoolContract.TableClasses.NAME\_TABLE\_CLASSES + ";");

db.execSQL("DROP TABLE IF EXISTS " +

SchoolContract.TableLearners.NAME\_TABLE\_LEARNERS + ";");

db.execSQL("DROP TABLE IF EXISTS " +

SchoolContract.TableLearnersOnPlaces.

NAME\_TABLE\_LEARNERS\_ON\_PLACES + ";");

db.execSQL("DROP TABLE IF EXISTS " +

SchoolContract.TableLearnersGrades.

NAME\_TABLE\_LEARNERS\_GRADES + ";");

db.execSQL("DROP TABLE IF EXISTS " +

SchoolContract.TableLessons.NAME\_TABLE\_LESSONS + ";");

db.execSQL("DROP TABLE IF EXISTS " + SchoolContract.TableLessonAndTimeWithCabinet.

NAME\_TABLE\_LESSONS\_AND\_TIME + ";");

db.execSQL("PRAGMA foreign\_keys = ON");

//кабинет

String sql = "CREATE TABLE " +

SchoolContract.TableCabinets.NAME\_TABLE\_CABINETS + "( " + SchoolContract.TableCabinets.KEY\_CABINET\_ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " + SchoolContract.TableCabinets.COLUMN\_NAME + " VARCHAR );";

db.execSQL(sql);

//парта

sql = "CREATE TABLE " +

SchoolContract.TableDesks.NAME\_TABLE\_DESKS + "( " +

SchoolContract.TableDesks.KEY\_DESK\_ID + " INTEGER PRIMARY

KEY AUTOINCREMENT, " +SchoolContract.TableDesks.COLUMN\_X

+ " INTEGER, " +SchoolContract.TableDesks.COLUMN\_Y +

" INTEGER, " +SchoolContract.TableDesks.

COLUMN\_NUMBER\_OF\_PLACES + " INTEGER, " + SchoolContract.

TableDesks.KEY\_CABINET\_ID + " INTEGER, "+"FOREIGN KEY("+

SchoolContract.TableDesks.KEY\_CABINET\_ID + ") REFERENCES "+

SchoolContract.TableCabinets.NAME\_TABLE\_CABINETS + "(" +SchoolContract. TableCabinets.KEY\_CABINET\_ID +

")ON DELETE CASCADE);";

db.execSQL(sql);

//место

sql = "CREATE TABLE " +

SchoolContract.TablePlaces.NAME\_TABLE\_PLACES + " ( " + SchoolContract.TablePlaces.KEY\_PLACE\_ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " +

SchoolContract.TablePlaces.KEY\_DESK\_ID + " INTEGER, " +

SchoolContract.TablePlaces.COLUMN\_ORDINAL + " INTEGER, " +

"FOREIGN KEY(" +

SchoolContract.TablePlaces.KEY\_DESK\_ID + ") REFERENCES " + SchoolContract.TableDesks.NAME\_TABLE\_DESKS + " (" + SchoolContract.TableDesks.KEY\_DESK\_ID + ") ON DELETE CASCADE ); ";

db.execSQL(sql);

//класс

sql = "CREATE TABLE " +

SchoolContract.TableClasses.NAME\_TABLE\_CLASSES + " ( " + SchoolContract.TableClasses.KEY\_CLASS\_ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " +

SchoolContract.TableClasses.COLUMN\_CLASS\_NAME + "

VARCHAR ); ";

db.execSQL(sql);

//ученик

sql = "CREATE TABLE " +

SchoolContract.TableLearners.NAME\_TABLE\_LEARNERS + " ( " + SchoolContract.TableLearners.KEY\_LEARNER\_ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " +

SchoolContract.TableLearners.COLUMN\_FIRST\_NAME +

" VARCHAR, " +

SchoolContract.TableLearners.COLUMN\_SECOND\_NAME +

" VARCHAR, " +

SchoolContract.TableLearners.KEY\_CLASS\_ID +

" INTEGER, " +

"FOREIGN KEY(" + SchoolContract.TableLearners.KEY\_CLASS\_ID +

") REFERENCES " + SchoolContract.TableClasses.NAME\_TABLE\_CLASSES + " (" + SchoolContract.TableClasses.KEY\_CLASS\_ID + ") ON DELETE CASCADE ); ";

db.execSQL(sql);

//ученик-место

sql = "CREATE TABLE " +

SchoolContract.TableLearnersOnPlaces.NAME\_TABLE\_LEARNERS\_ON\_PLACES + " ( " + SchoolContract.TableLearnersOnPlaces.KEY\_ATTITUDES\_ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " +

SchoolContract.TableLearnersOnPlaces.KEY\_LEARNER\_ID + "

INTEGER, " +

SchoolContract.TableLearnersOnPlaces.KEY\_PLACE\_ID + "

INTEGER, " + "FOREIGN KEY(" + SchoolContract.TableLearnersOnPlaces.KEY\_LEARNER\_ID + ") REFERENCES " + SchoolContract.TableLearners.NAME\_TABLE\_LEARNERS + " (" + SchoolContract.TableLearners.KEY\_LEARNER\_ID + ") ON DELETE CASCADE, " +"FOREIGN KEY(" + SchoolContract.TableLearnersOnPlaces.KEY\_PLACE\_ID + ") REFERENCES " + SchoolContract.TablePlaces.NAME\_TABLE\_PLACES + " (" + SchoolContract.TablePlaces.KEY\_PLACE\_ID + ") ON DELETE CASCADE ); ";

db.execSQL(sql);

//оценки учеников

sql = "CREATE TABLE " +

SchoolContract.TableLearnersGrades.NAME\_TABLE\_LEARNERS\_GRADS

+ " ( " + SchoolContract.TableLearnersGrades.KEY\_GRADE\_ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " +

SchoolContract.TableLearnersGrades.KEY\_LEARNER\_ID

+ " INTEGER, " +

SchoolContract.TableLearnersGrades.COLUMN\_GRADE +

" INTEGER, " +

SchoolContract.TableLearnersGrades.KEY\_LESSON\_ID + "

INTEGER, " + "FOREIGN KEY(" + SchoolContract.TableLearnersGrades.KEY\_LEARNER\_ID + ") REFERENCES " + SchoolContract.TableLearners.NAME\_TABLE\_LEARNERS + " (" + SchoolContract.TableLearners.KEY\_LEARNER\_ID + ") ON DELETE CASCADE ); ";

db.execSQL(sql);

//уроки

sql = "CREATE TABLE " +

SchoolContract.TableLessons.NAME\_TABLE\_LESSONS + " ( " + SchoolContract.TableLessons.KEY\_LESSON\_ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " +

SchoolContract.TableLessons.COLUMN\_NAME + " VARCHAR, "+

SchoolContract.TableLessons.KEY\_CLASS\_ID +

" INTEGER, " +

"FOREIGN KEY(" + SchoolContract.TableLessons.KEY\_CLASS\_ID + ") REFERENCES " + SchoolContract.TableClasses.NAME\_TABLE\_CLASSES + " (" + SchoolContract.TableClasses.KEY\_CLASS\_ID + ") ON DELETE CASCADE ); ";

db.execSQL(sql);

//урок и его время проведения

sql = "CREATE TABLE " +

SchoolContract.TableLessonAndTimeWithCabinet.NAME\_TABLE\_LESSONS\_AND\_TIME + " ( " + SchoolContract.TableLessonAndTimeWithCabinet.KEY\_LESSON\_AND\_TIME\_ATTITUDE\_ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " + SchoolContract.TableLessonAndTimeWithCabinet.KEY\_LESSON\_ID + " INTEGER, " +

SchoolContract.TableLessonAndTimeWithCabinet.KEY\_CABINET\_ID + " INTEGER, " +

SchoolContract.TableLessonAndTimeWithCabinet.

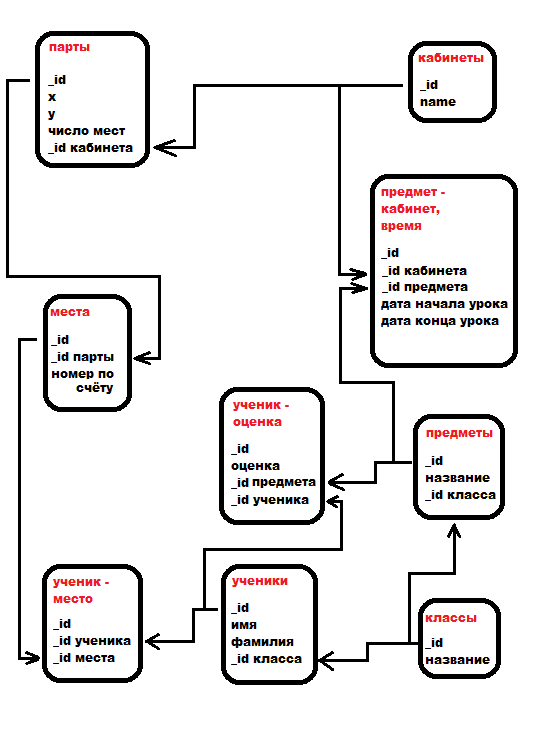
COLUMN\_DATE\_BEGIN + " INTEGER, " + SchoolContract.TableLessonAndTimeWithCabinet.COLUMN\_DATE\_END + " INTEGER, " + "FOREIGN KEY(" + SchoolContract.TableLessonAndTimeWithCabinet.KEY\_LESSON\_ID + ") REFERENCES " + SchoolContract.TableLessons.NAME\_TABLE\_LESSONS + " (" + SchoolContract.TableLessons.KEY\_LESSON\_ID + ") ON DELETE CASCADE ); ";

db.execSQL(sql);

}

}

...

 Рисунок 1 - Диаграмма таблиц базы данных, использующейся приложении

Для работы с БД приложения необходимо создать экземпляр класса унаследованного от SQLiteOpenHelper, в нашем случае это DataBaseOpenHelper.

Пример чтения из базы данных (получение не рассаженных учеников):

...

public ArrayList<Long> getNotPutLearnersIdByCabinetIdAndClassId(

long cabinetId,long classId){

ArrayList<Long> placesId = new ArrayList<>();

{//получаем id мест

ArrayList<Long> desksId = new ArrayList<>();

{

Cursor desksCursor = this.getDesksByCabinetId(cabinetId);

while (desksCursor.moveToNext()) {

desksId.add(desksCursor.getLong(

desksCursor.getColumnIndex(

SchoolContract.TableDesks.KEY\_DESK\_ID)));

}

desksCursor.close();

}

for (int i = 0; i < desksId.size(); i++) {

Cursor placeCursor = this.getPlacesByDeskId(desksId.get(i));

while (placeCursor.moveToNext()) {

placesId.add(placeCursor.getLong(

placeCursor.getColumnIndex(

SchoolContract.TablePlaces.KEY\_PLACE\_ID)));

}

placeCursor.close();

}

}

ArrayList<Long> learnersId = new ArrayList<>();

{//получаем id

Cursor learnersCursor = this.getLearnersByClassId(classId);

while (learnersCursor.moveToNext()) {

learnersId.add(learnersCursor.getLong(

learnersCursor.getColumnIndex(

SchoolContract.TableLearners.KEY\_LEARNER\_ID)));

}

learnersCursor.close();

}

ArrayList<Long> answer = new ArrayList<>();

//пробегаемся по ученикам

for (int i = 0; i < learnersId.size(); i++) {

boolean flag = false;

for (int j = 0; j < placesId.size(); j++) {

//пробегаемся с учеником по местам

Cursor cursor = this.getAttitudeByLearnerIdAndPlaceId(

learnersId.get(i), placesId.get(j));

//если есть совпадение ученик - место

if (cursor.getCount() != 0) {

flag = true;

}

cursor.close();

}

if (!flag) {//если совпадений нет, то,

answer.add(learnersId.get(i));

//запоминаем не рассаженного ученика

}

}

Log.i("DBOpenHelper",

"getNotPutLearnersIdByCabinetIdAndClassId cabinetId=" +

cabinetId + " classId=" + classId +

" return=" + answer);

return answer;

}

...

Для хранения настроек приложения был выбран класс SharedPreferences. SharedPreferences – это постоянное хранилище на платформе Android, способное хранить в себе примитивные типы данных, строки, массивы строк в виде XML-файла. Значения хранятся в виде: ключ – значение.

**2.2 Построение структурной схемы программного продукта**

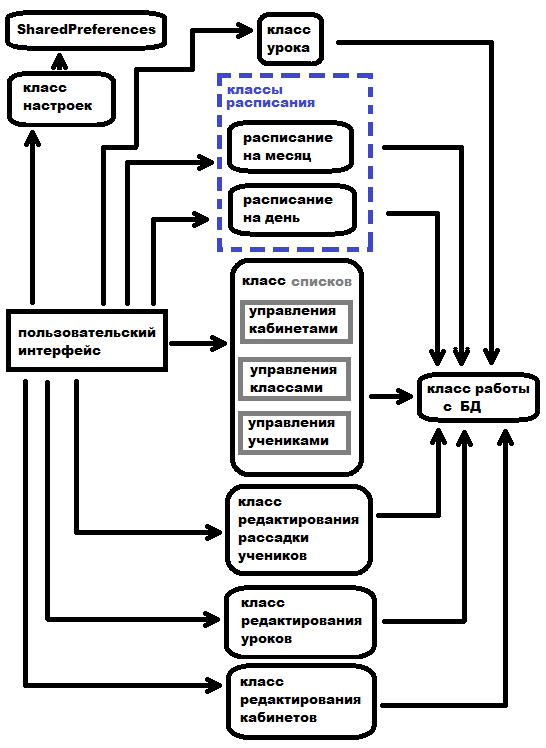


Рисунок 2 - Структурная схема пользовательской программы.

Java – объектно-ориентированный язык программирования, а потому в нем нет понятия модулей, но есть понятие классов, в которых реализуются необходимые функции:

Класс списков – выводит информацию из базы данных в список, реализует возможности добавления, редактирования и удаления. Состоит из трёх частей: кабинеты, классы и ученики.

Класс урока – выводит из базы данных парты и учеников в графический интерфейс, позволяет пользователю добавлять оценки в базу данных через графический интерфейс

Классы расписания – два класса выводящих в графический интерфейс расписание пользователя с созданными уроками.

Класс редактирования уроков – задание параметров для определённого урока

Класс редактирования кабинетов с графическим интерфейсом – изменение координат, создание и удаление парт в кабинетах.

Класс редактирования рассадки учеников – создание, удаление зависимостей «ученик – место» с графическим интерфейсом.

Класс работы с базой данных – класс отвечающий за все операции с БД.

Класс настроек – позволяет пользователю отредактировать интерфейс и очистить БД.

**3 Разработка интерфейса**

**3.1 Построение диаграммы состояний интерфейса**

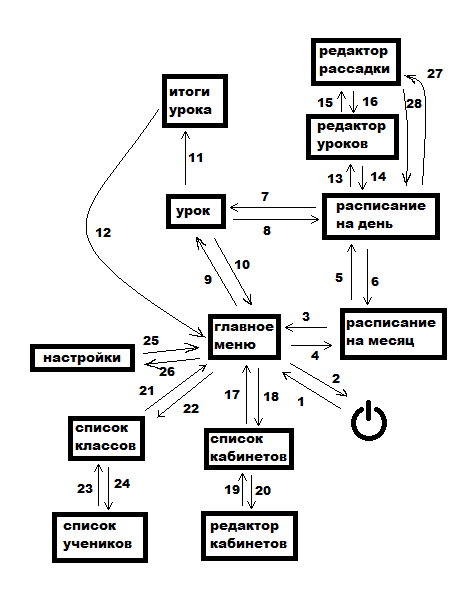
****

Рисунок 3 – Диаграмма состояний интерфейса приложения

1 – запуск приложения;

2 – нажатие кнопки «назад»;

3 – нажатие кнопки «назад»;

4 – выбор пункта «моё расписание»;

5 – выбор дня;

6 - нажатие кнопки «назад»;

7 – выбор урока;

8 - нажатие кнопки «назад»;

9 - выбор пункта «текущий урок» (при наличии урока на текущее время);

10 - нажатие кнопки «назад»;

11 - нажатие кнопки «закончить»;

12 - нажатие кнопки «сохранить» или «назад» (без сохранения результатов);

13 – нажатие на пункт списка без урока или долгое нажатие на урок;

14 - нажатие кнопки «назад»;

15 - нажатие кнопки «редактировать рассадку»;

16 - нажатие кнопки «назад»;

17 - нажатие кнопки «назад»;

18 - выбор пункта «мои кабинеты»;

19 - нажатие кнопки «назад»;

20 - выбор кабинета из списка;

21 - нажатие кнопки «назад»;

22 - выбор пункта «мои ученики»;

23 - нажатие кнопки «назад»;

24 - выбор класса из списка;

25 - нажатие кнопки «назад»;

26 - выбор пункта «настройки»;

27 - выбор урока (если ученики не рассажены в классе где проходит урок);

28 - нажатие кнопки «назад»;

Так как структура списков одинакова с целью уменьшения размера приложения было решено сделать универсальную активность для отображения списков (ученики, классы, кабинеты), которая работает в разных режимах.

**3.2 Проектирование форм ввода вывода информации**

Так как приложение разрабатывалось для учительской аудитории, основной задачей стало создать максимально удобный интерфейс.

При входе в приложение пользователю предлагается главное меню с навигацией по всему приложению:



Рисунок 4 – главное меню

Из главного меню пользователю доступны «текущий урок», «расписание», «список кабинетов», «список классов», «настройки». Рассмотрим «список кабинетов»:

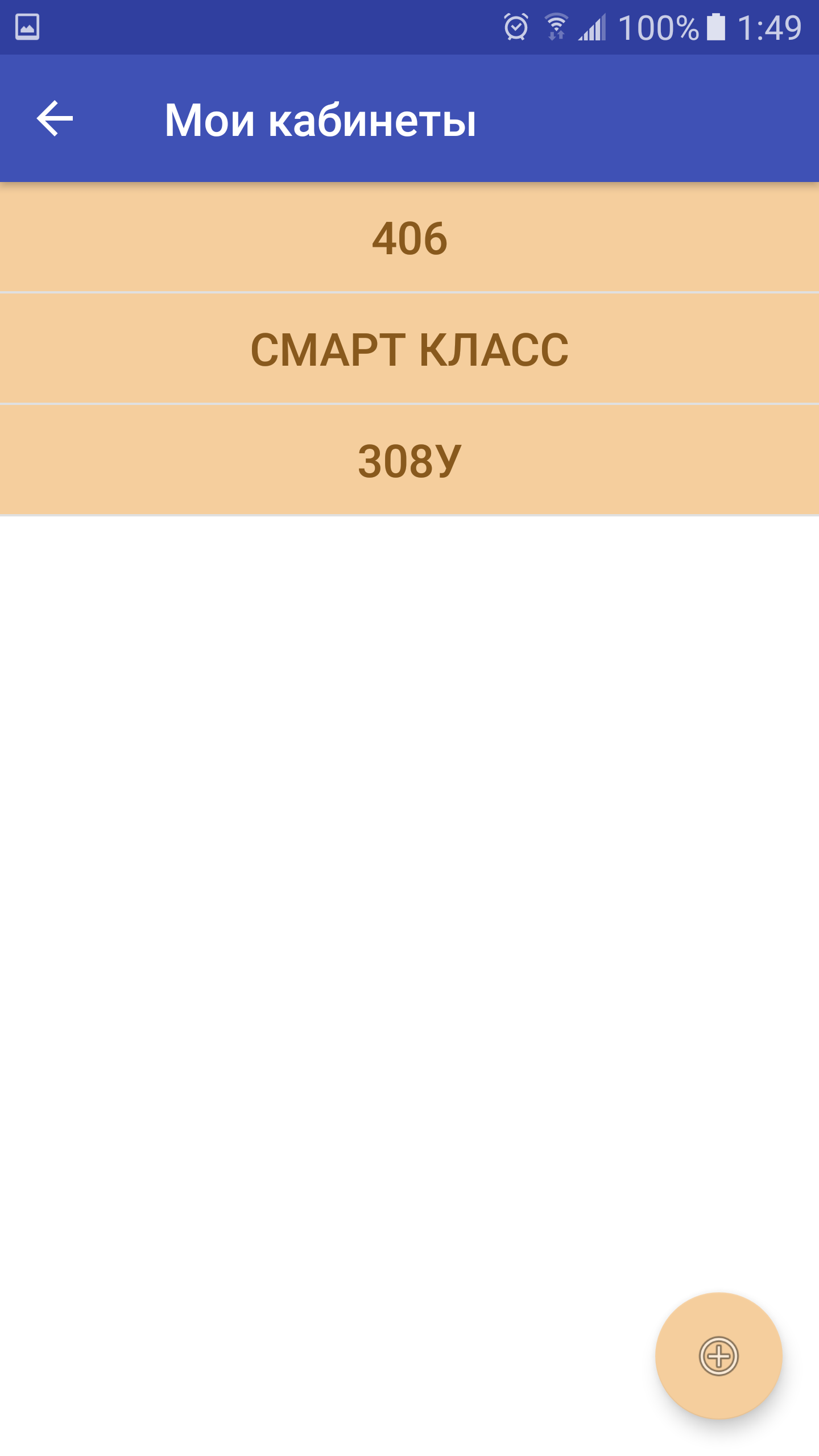


Рисунок 5 – список кабинетов

Список построен на ListView. В правом нижнем углу находится кнопка для добавления кабинета, при нажатии на неё открывается диалог «создание кабинета» в котором пользователю нужно ввести название нового кабинета.

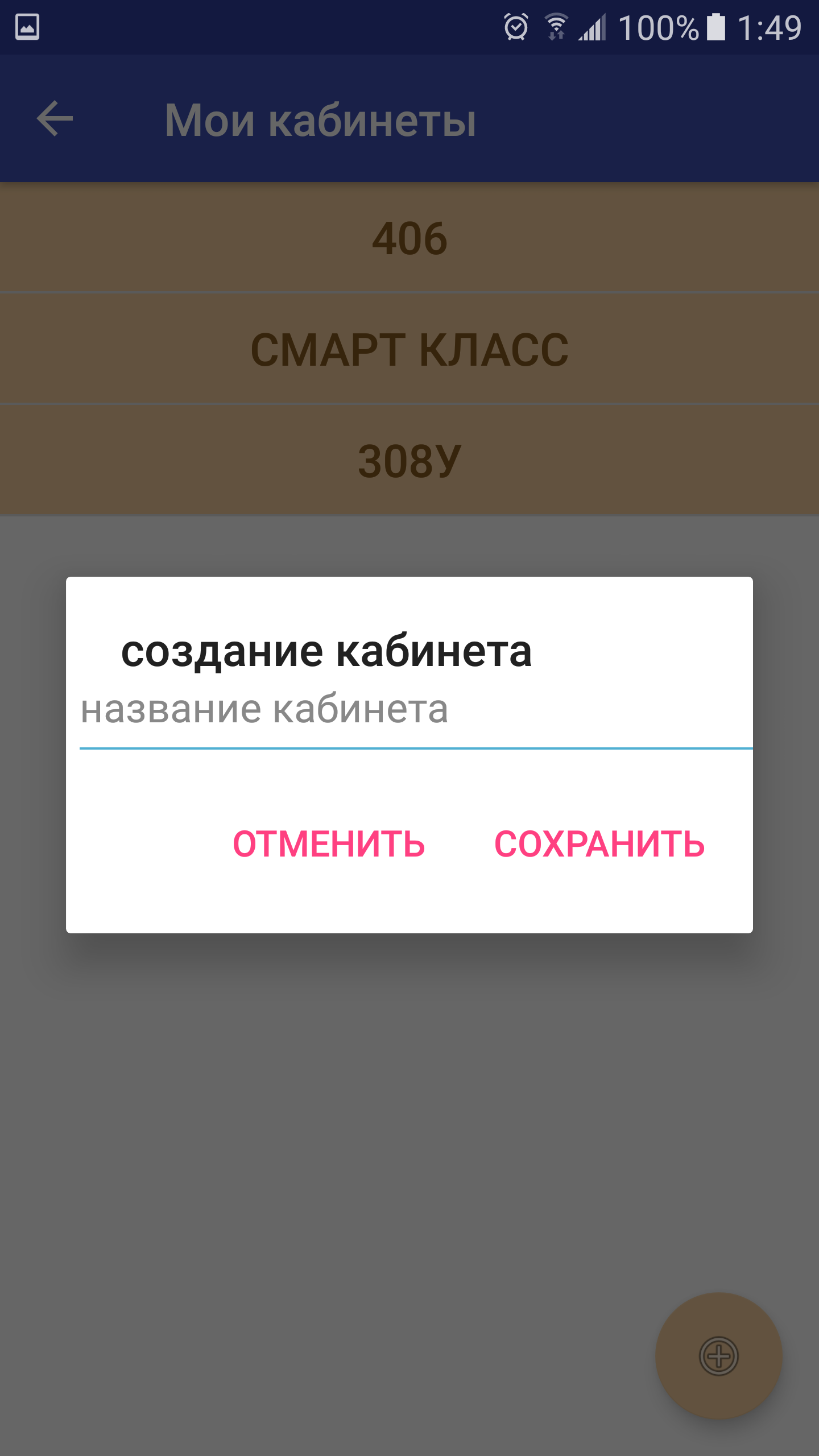


Рисунок 6 – список кабинетов – диалог создания

При длинном нажатии на пункт списка открывается дополнительное меню. Кнопка «редактировать» открывает диалог редактирования имени кабинета.

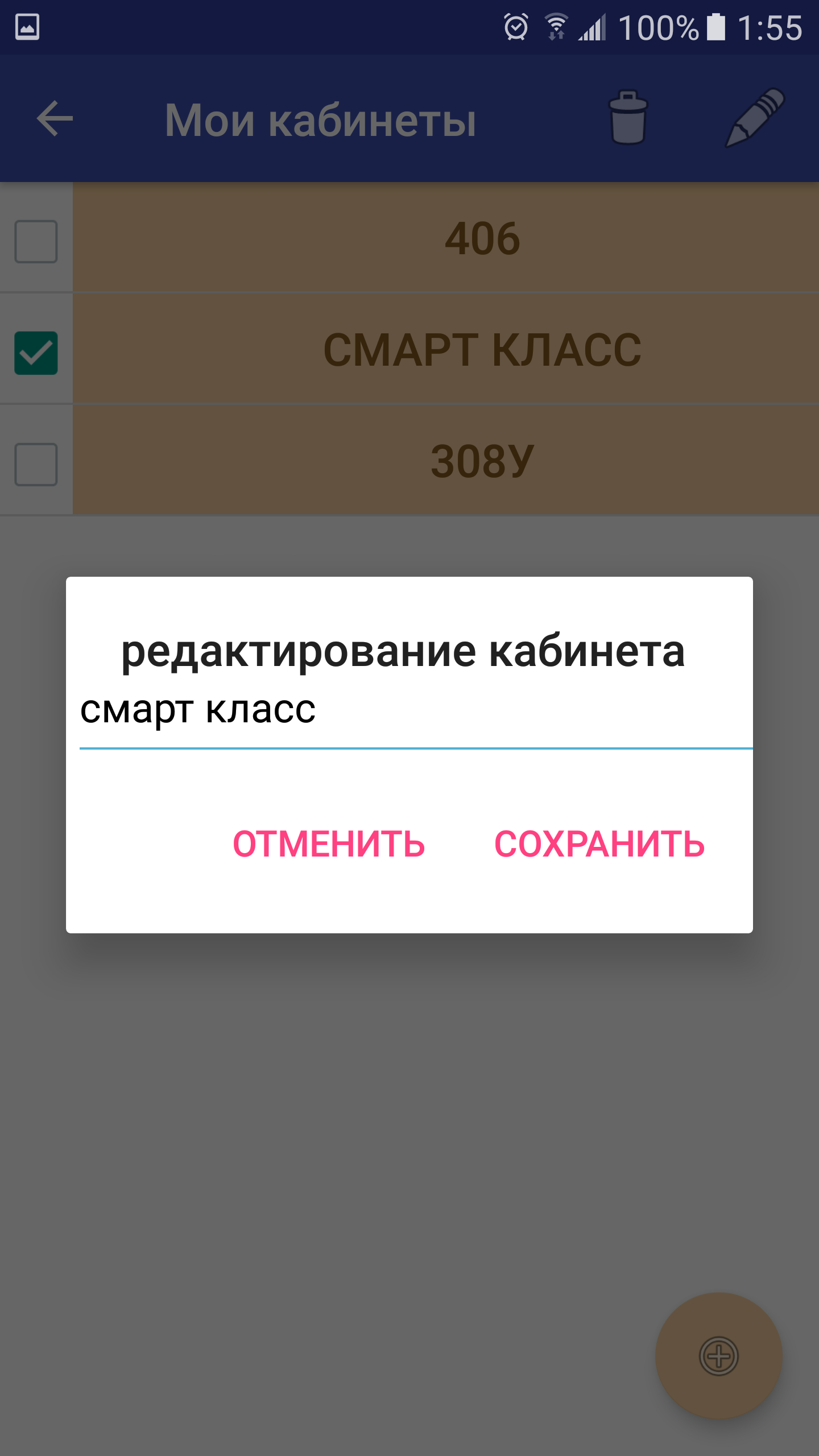


Рисунок 7 – список кабинетов – диалог редактирования имени

Нажатие на кнопку «удалить», удаляет выбранные кабинеты.

Нажатие на кабинет из списка, открывает активность редактирования кабинета.

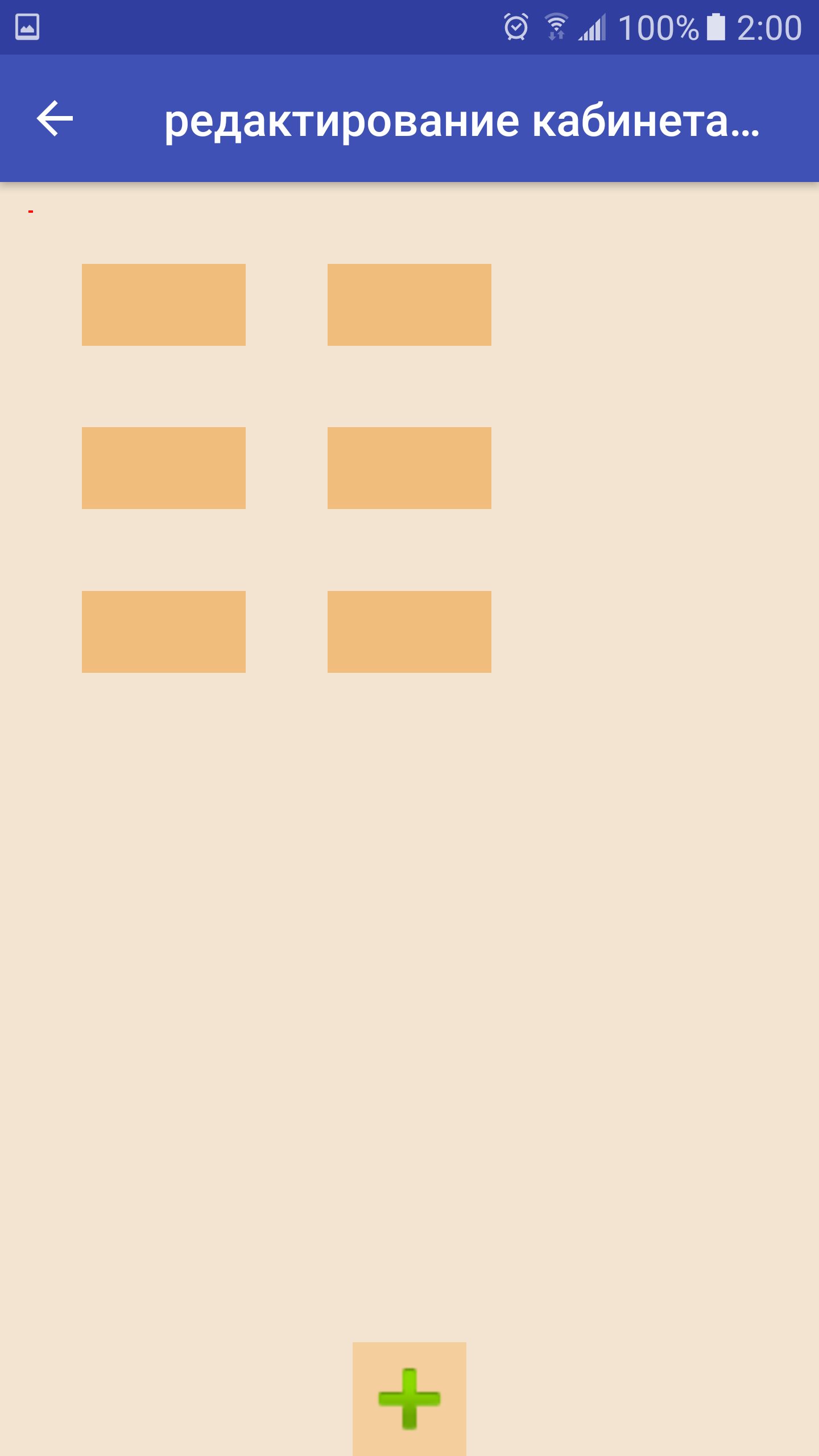


Рисунок 8 – редактирование кабинета

При нажатии на кнопку «добавить» в центре экрана появляется новая парта.

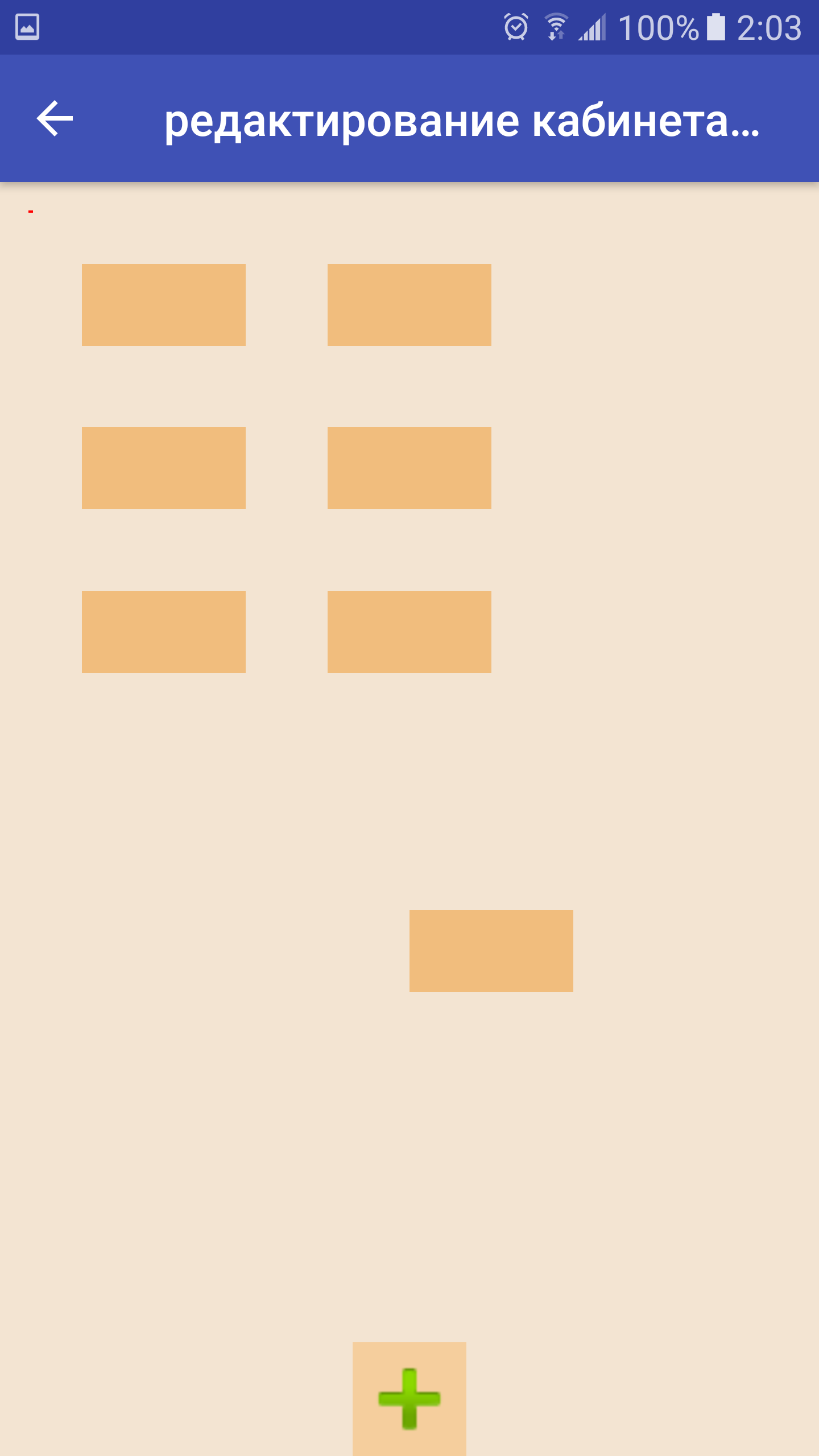


Рисунок 9 – редактирование кабинета – новая парта

При нажатии удерживании парты, её можно перемещать по экрану, при этом кнопка «добавить» будет меняться на кнопку «удалить».

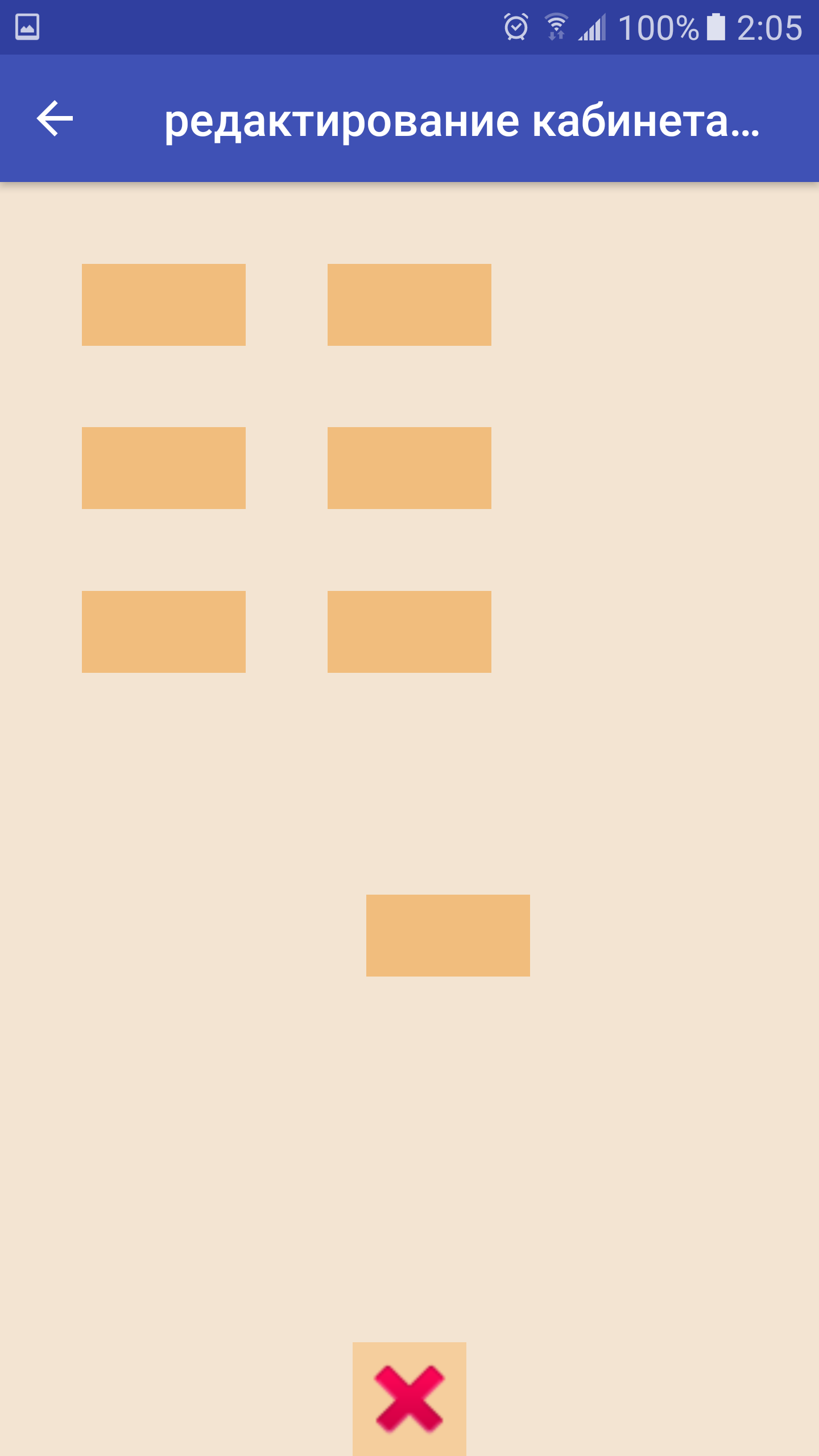


Рисунок 10 – редактирование кабинета – перемещение парты

Если переместить парту к кнопке «удалить», парта удаляется с экрана и из БД, а кнопка «удалить» заменится на кнопку «добавить».

Рассмотрим список классов.

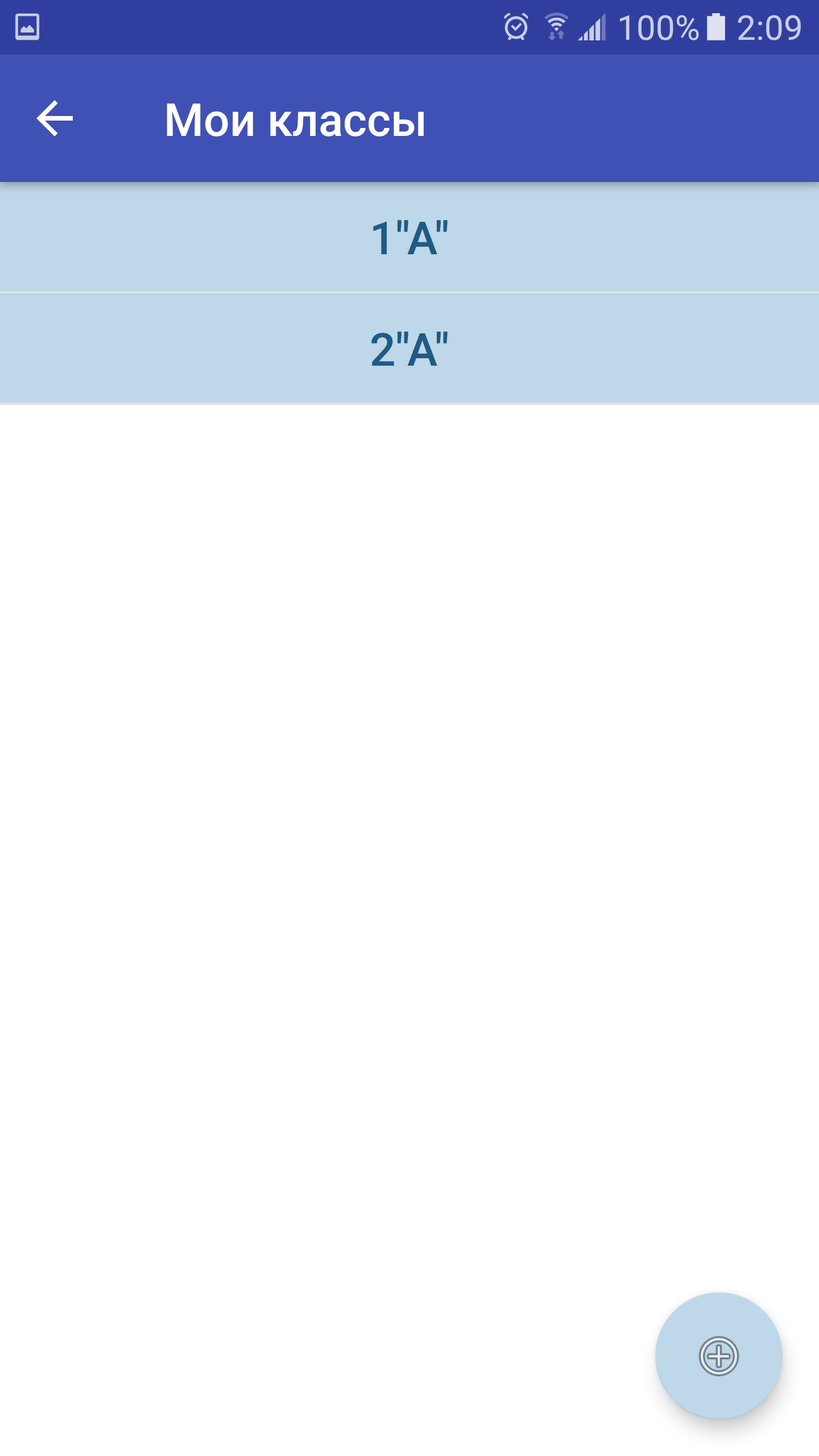


Рисунок 11 – список классов

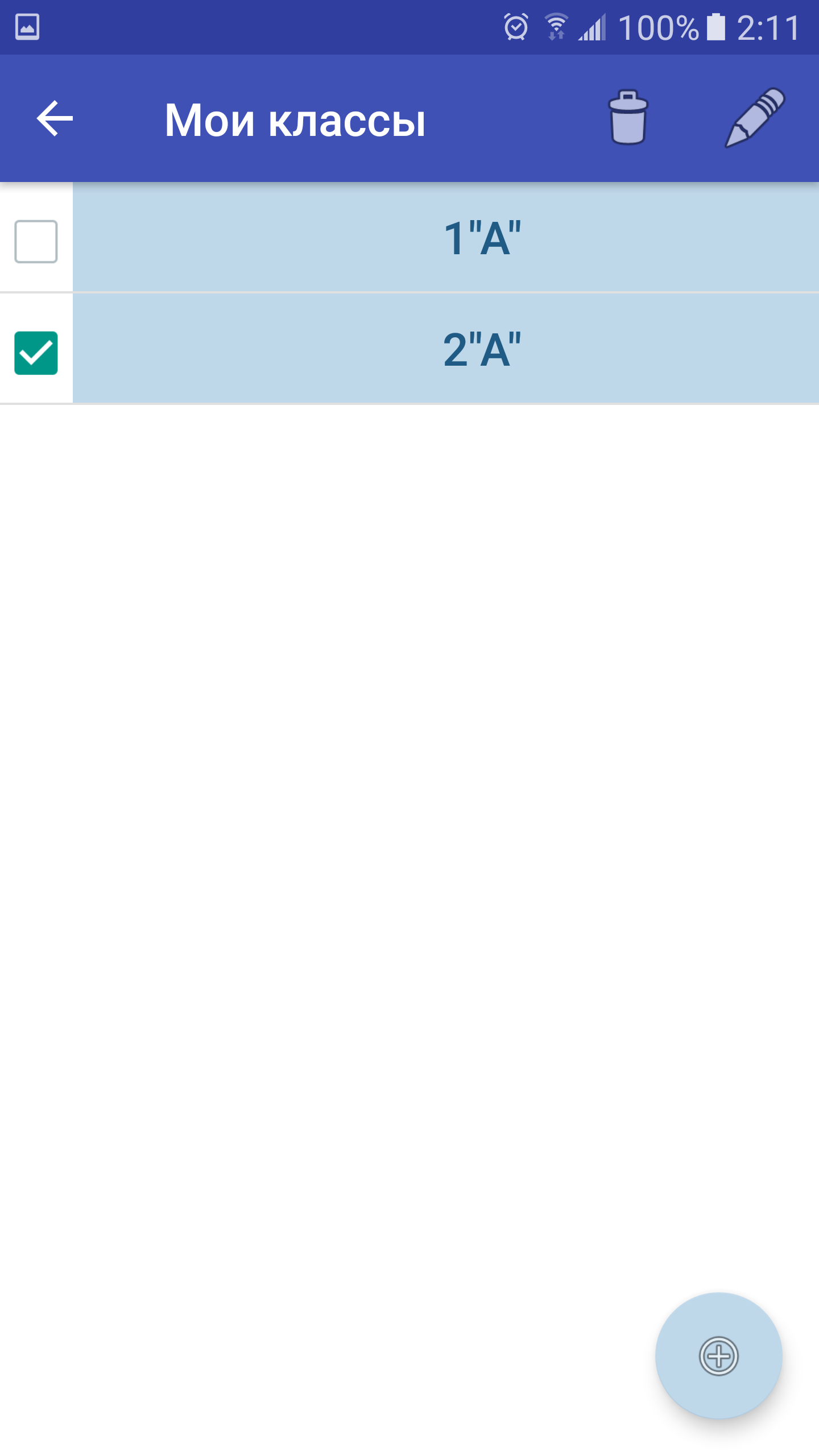


Рисунок 12 – список классов – элементы меню

Интерфейс аналогичен интерфейсу списка кабинетов, но при нажатии на пункт списка мы попадаем в список учеников в выбранном классе.



Рисунок 13 – список учеников

В этом списке интерфейс схож с двумя предыдущими, но в диалоге теперь два поля: для имени и для фамилии.

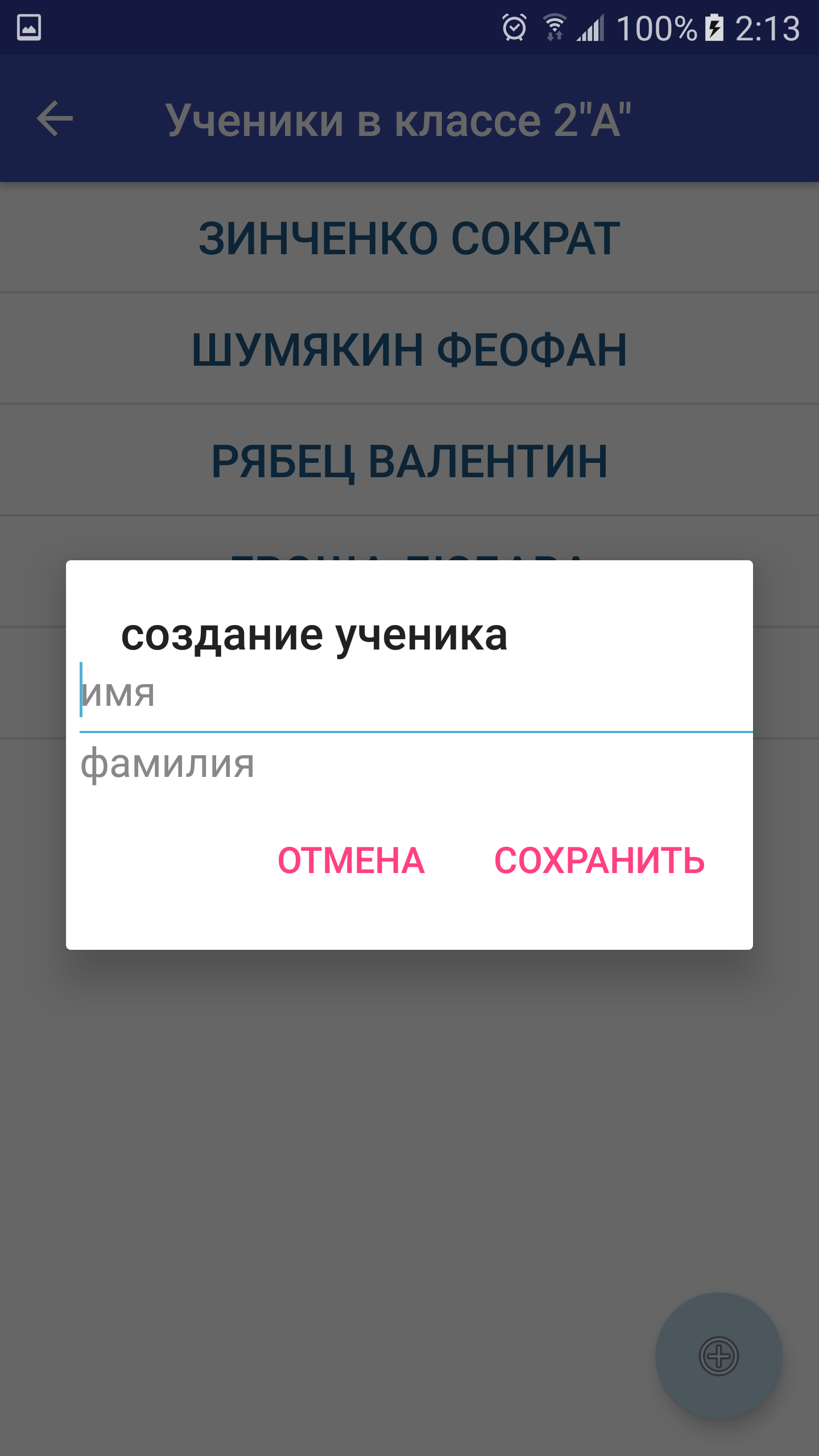


Рисунок 14 – список учеников – диалог создания

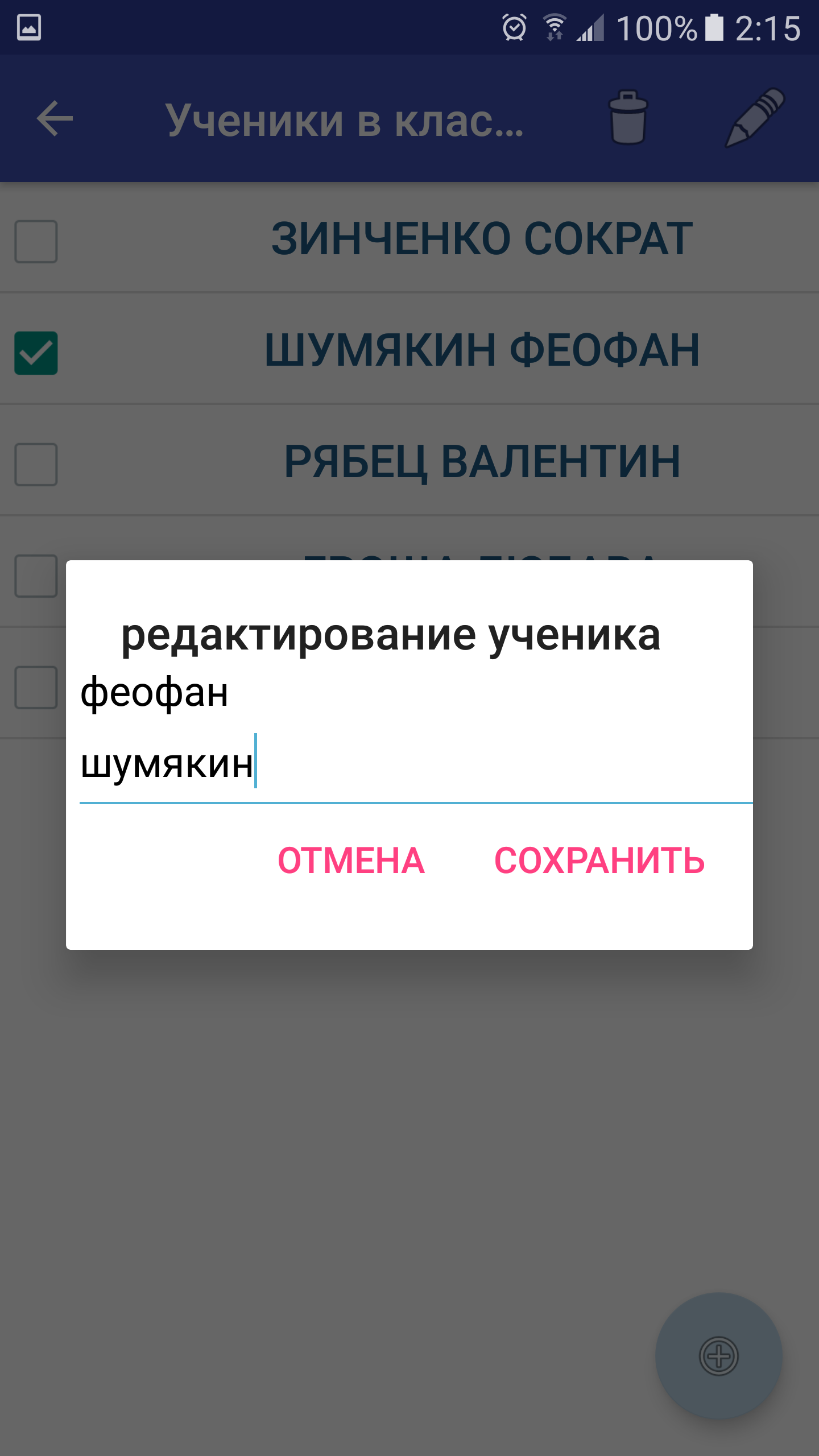


Рисунок 15 – список учеников – диалог редактирования

Рассмотрим пункт «моё расписание». Выводится календарь на текущий месяц, красным цветом обозначается текущий день, а тёмно-зелёным, день когда есть уроки.



Рисунок 16 – расписание на месяц

При помощи кнопок «назад» и «вперёд» или при помощи перелистывания, можно перемещаться между месяцами. При нажатии на день открывается расписание на этот день.

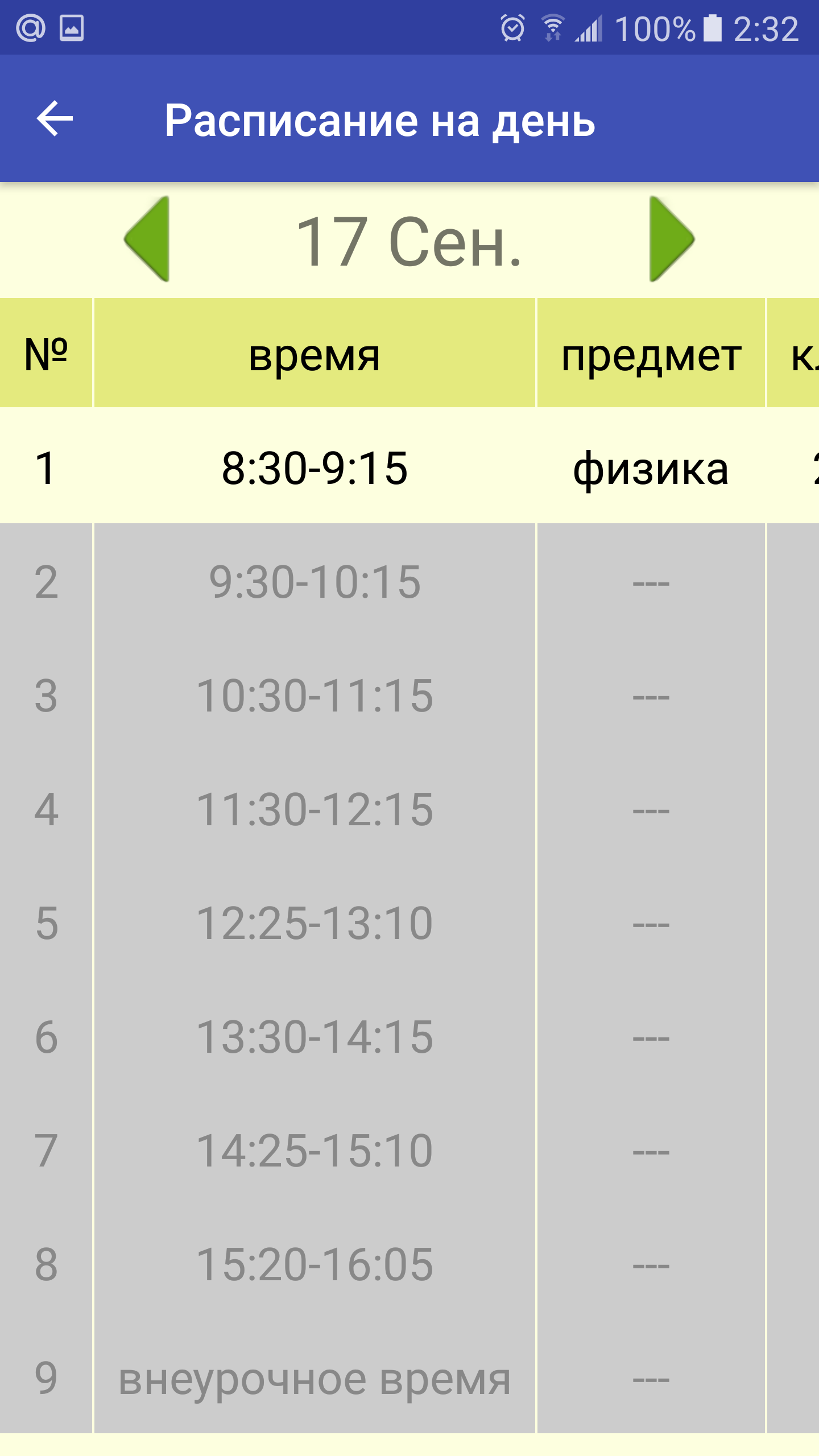


Рисунок 17 – расписание на день



Рисунок 18 – расписание на день



Рисунок 19 – расписание на день

При нажатии на пункт без урока или удерживании пункта с уроком открывается редактор урока.

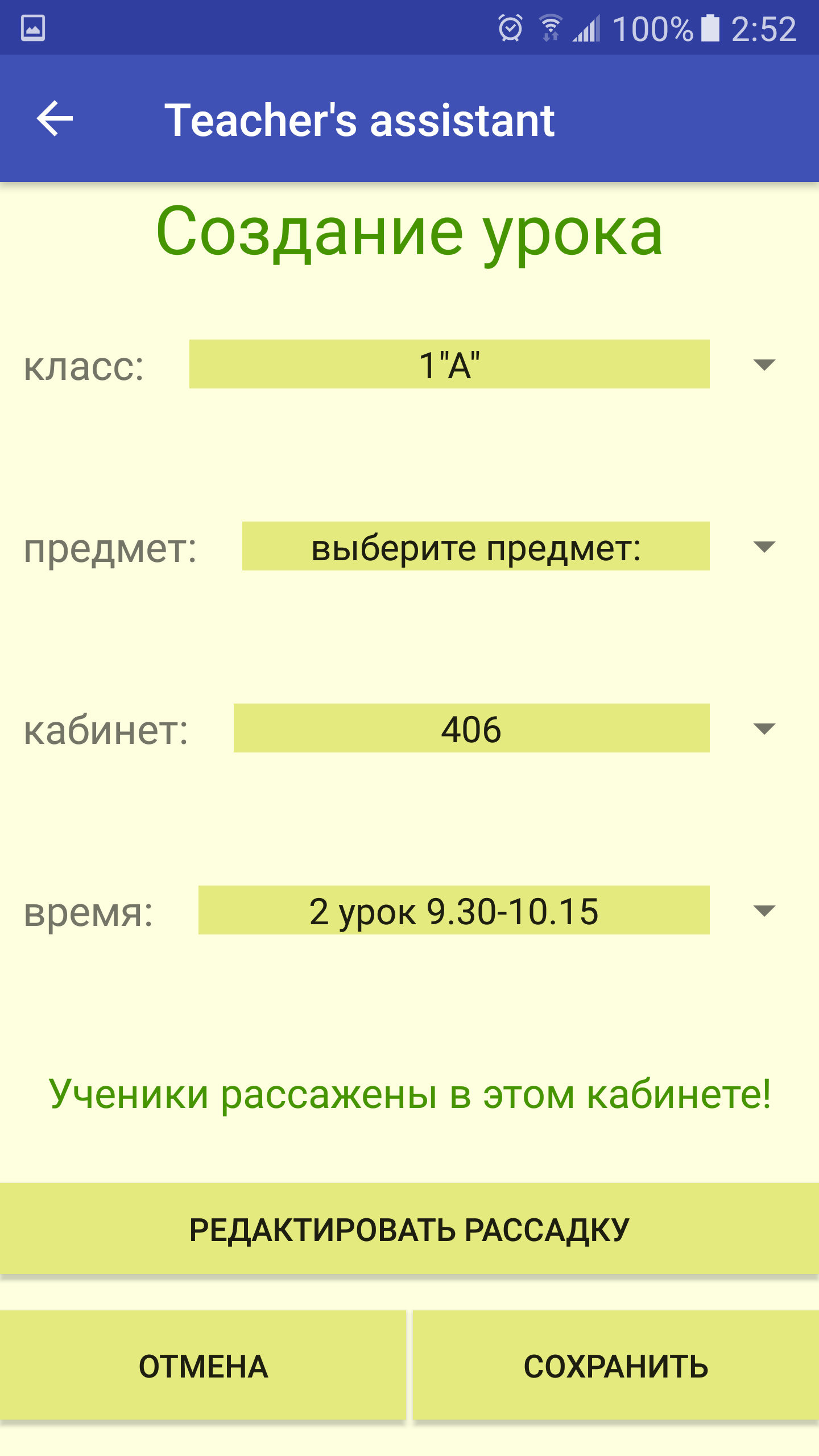


Рисунок 20 – редактор урока

В выпадающих списках можно выбрать параметры,

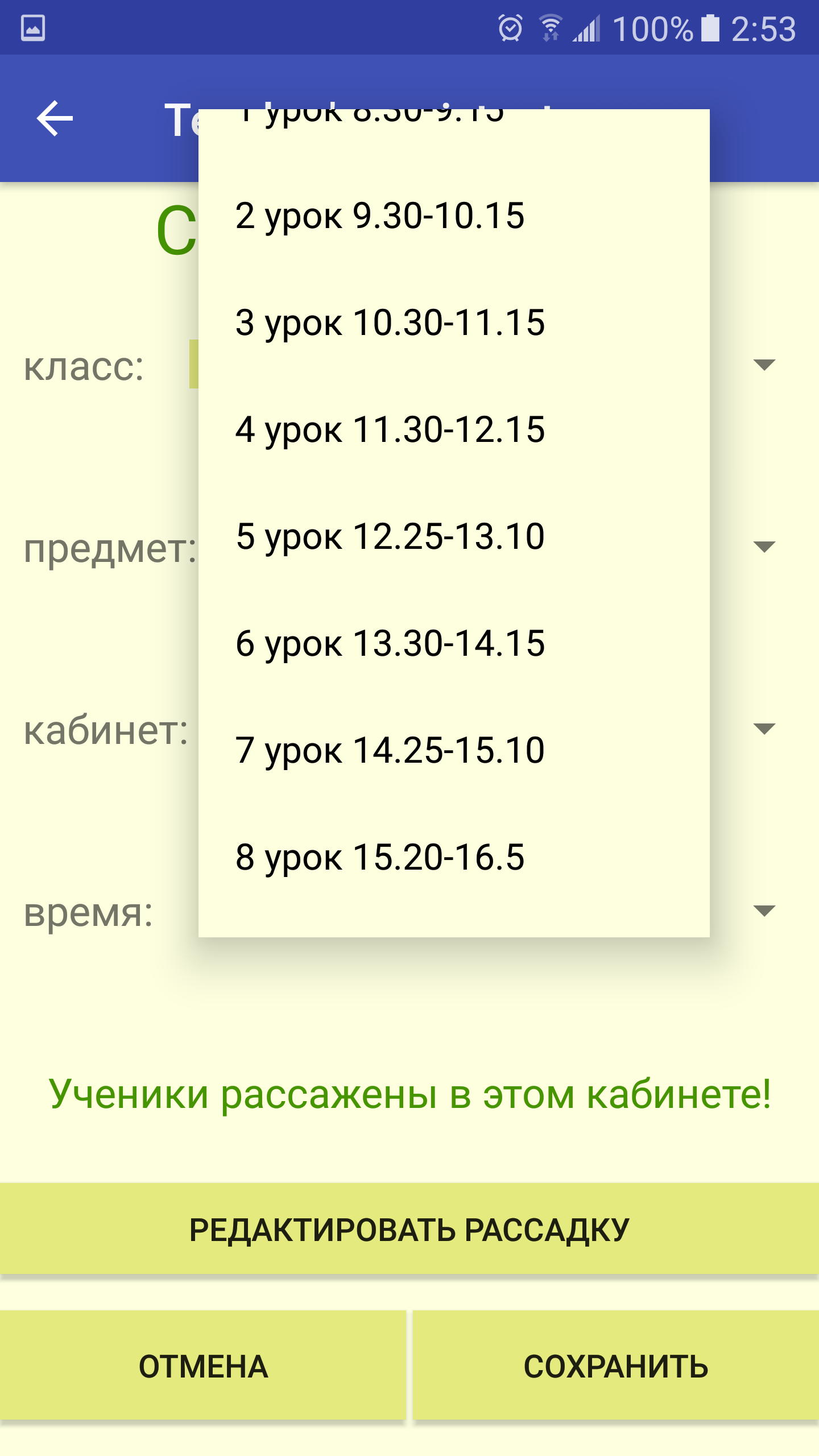


Рисунок 21 – редактор урока – список времени

В списке «предметы» пункт «создать предмет» открывает диалог, в котором надо ввести имя предмета.

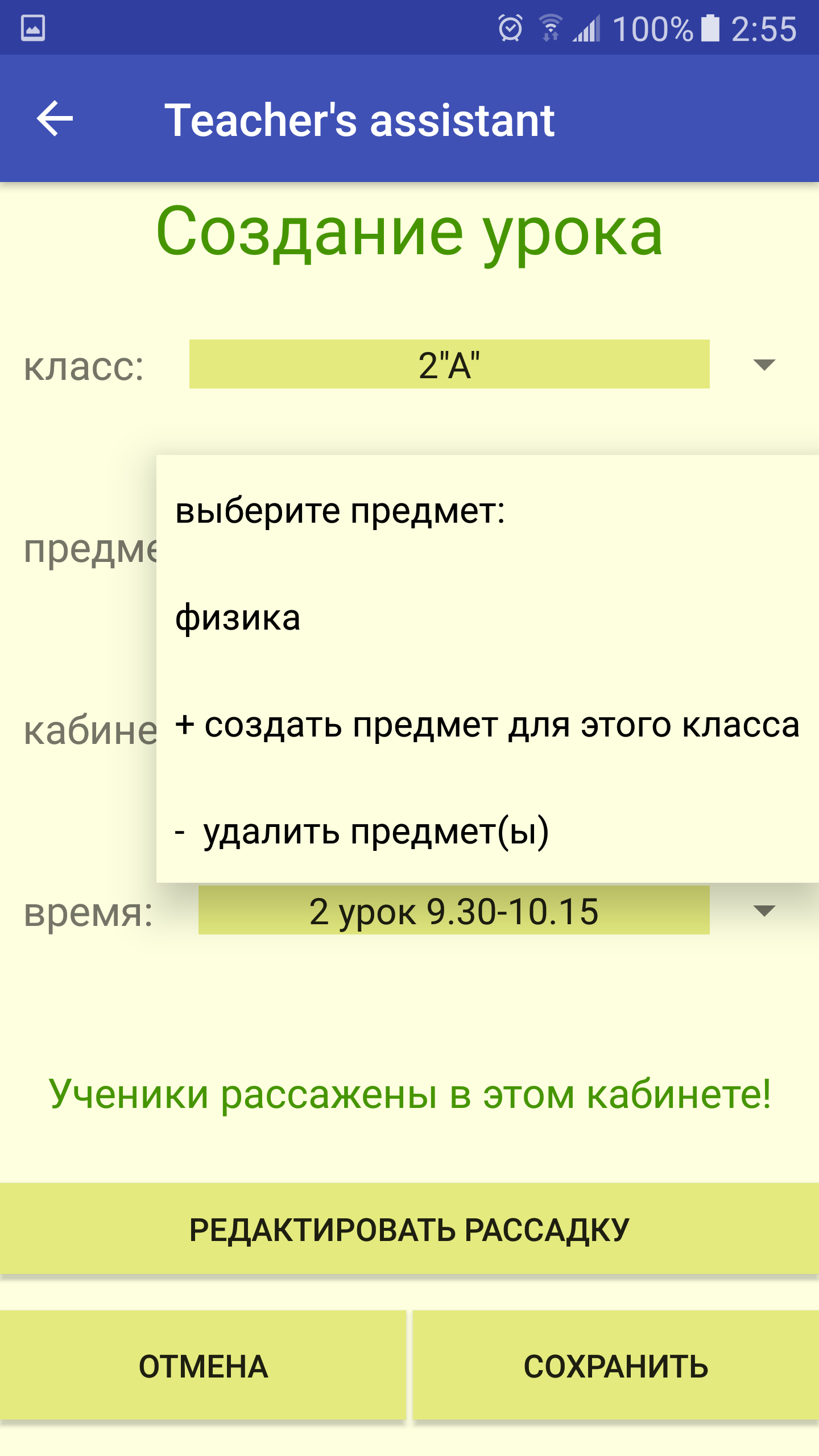


Рисунок 22 – редактор урока – список предметов

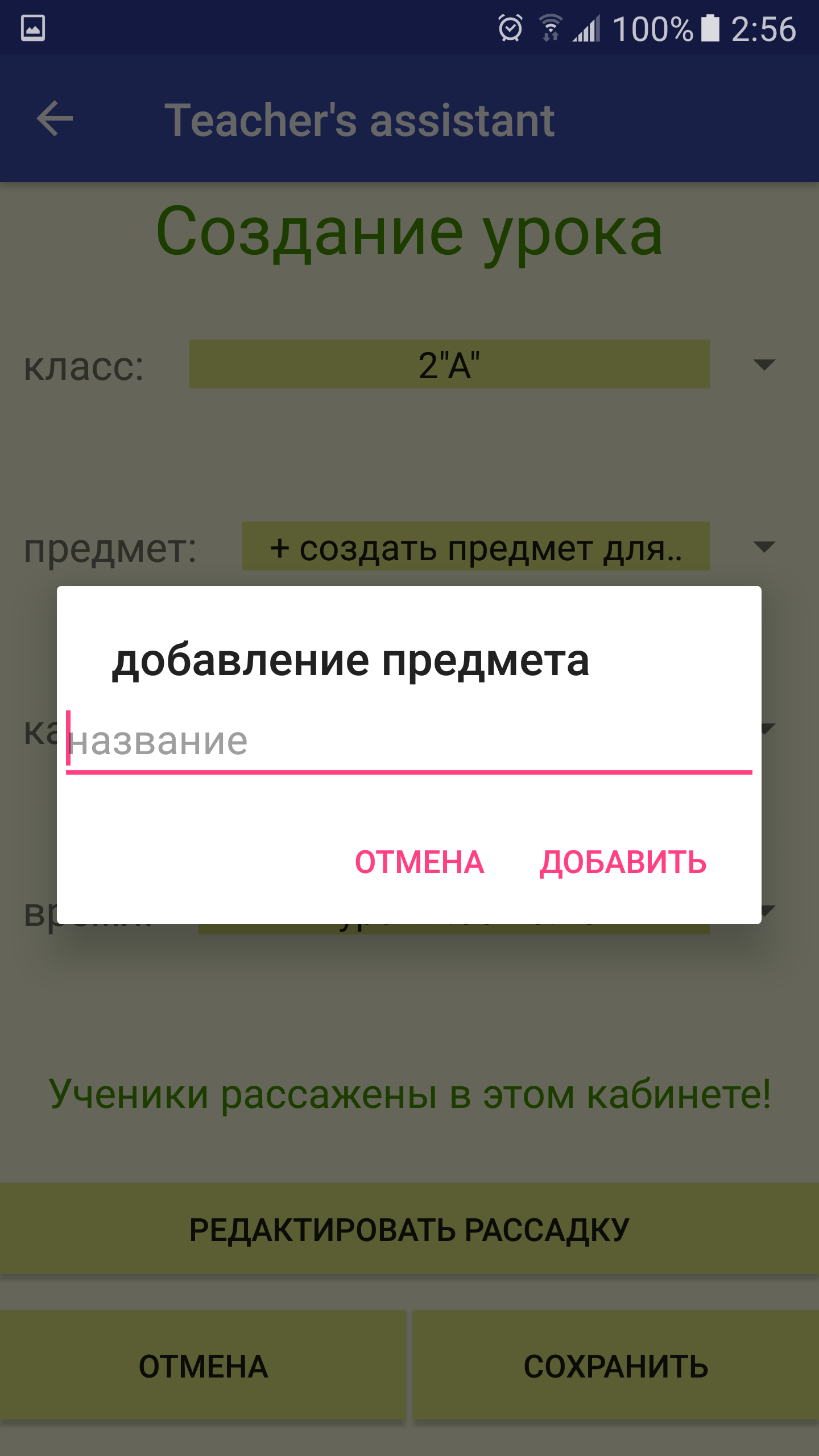


Рисунок 23 – редактор урока – список предметов – создание

А пункт «удалить предметы» (доступен если есть предметы) открывает диалог со списком доступных предметов и флажками выбора. При нажатии кнопки «удалить» удаляются все выбранные предметы.

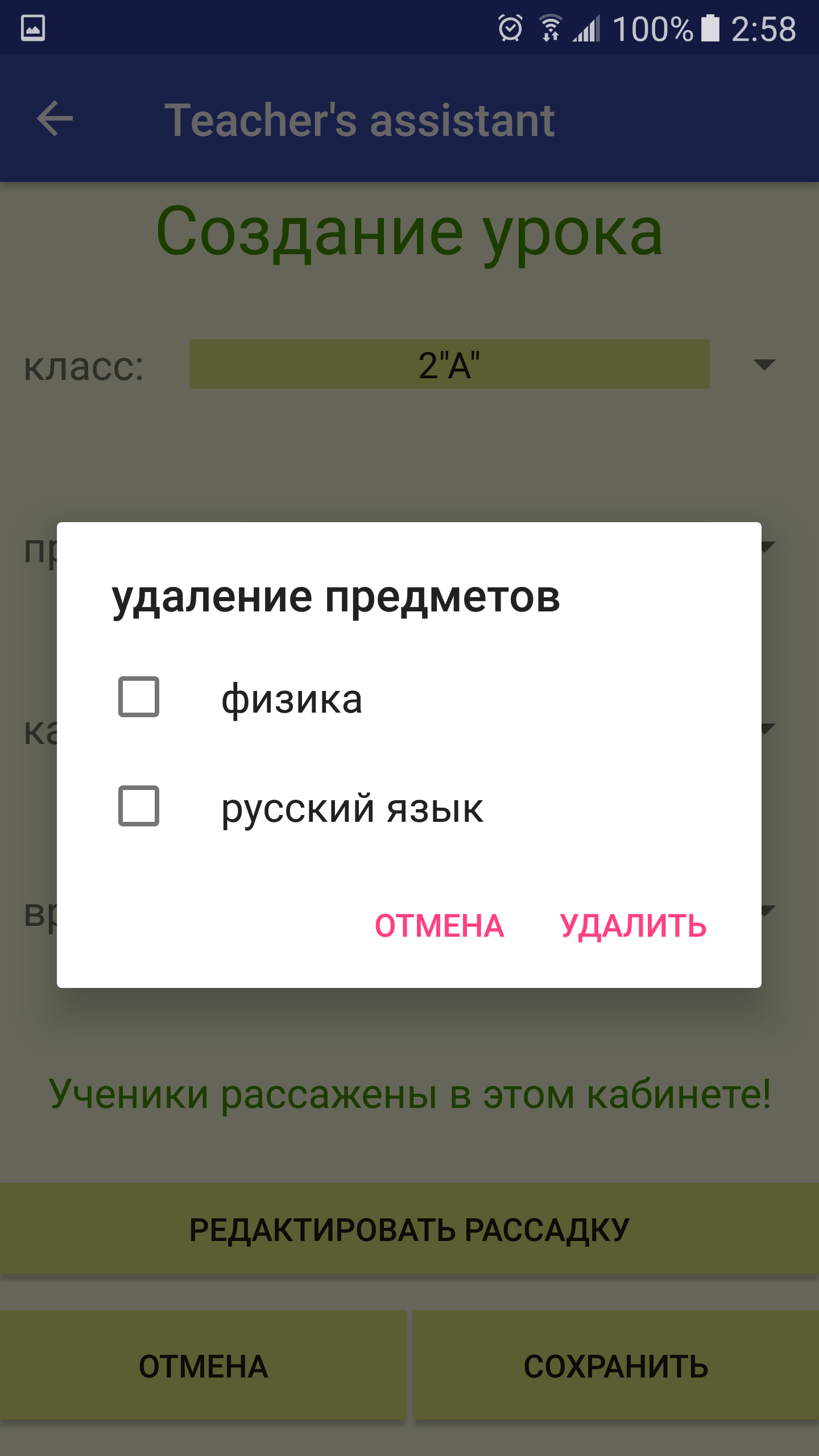


Рисунок 24 – редактор урока – список предметов - удаление

Ниже находится текстовое поле с информацией о том рассажены ли ученики выбранного класса в выбранном кабинете.

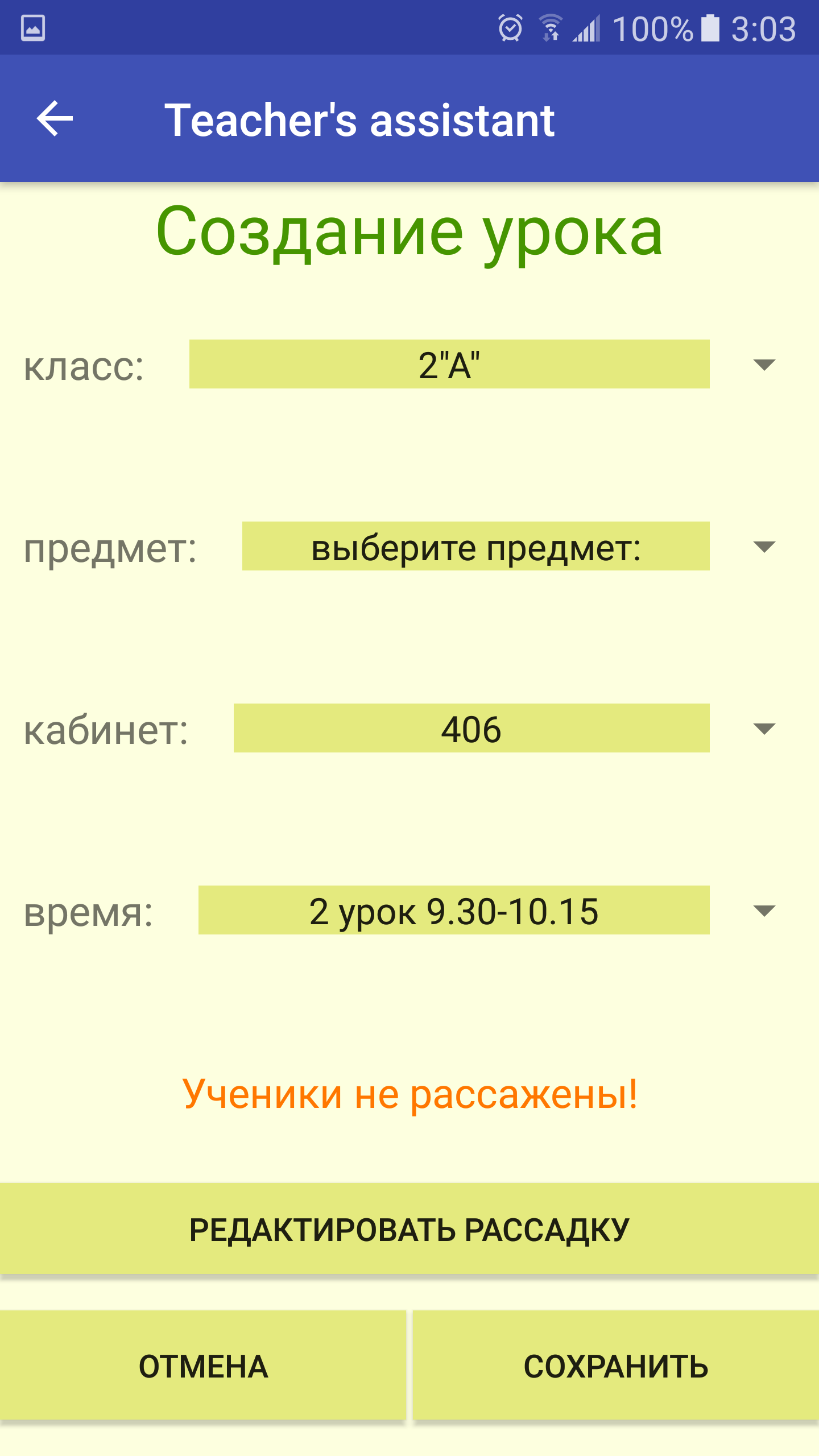


Рисунок 25 – редактор урока – ученики не рассажены

Под ним располагается кнопка перехода на редактор рассадки учеников выбранного класса в выбранном кабинете.

В самом низу есть кнопки «сохранить» и «отмена» (если в расписании на день выбран не пустой урок, то вместо «отмены» будет кнопка «удалить урок» ).

«Сохранить» - сохраняет все изменения, «удалить урок» - удаляет его из базы, «отмена» - тоже что и назад.

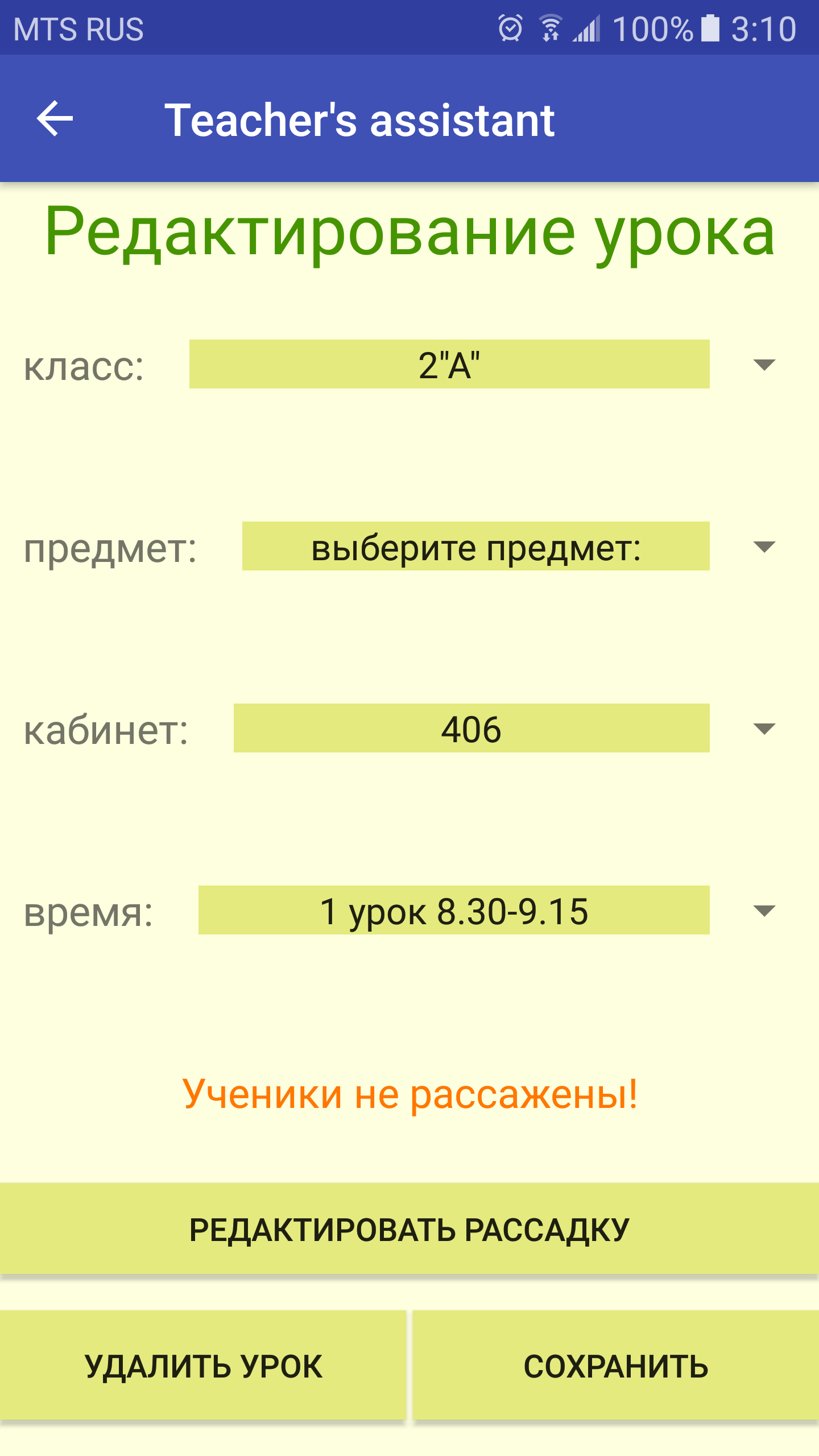


Рисунок 26 – редактор урока – редактирование уже созданного



Рисунок 27 – расписание на день – не рассаженные ученики

Рассмотрим редактор рассадки.

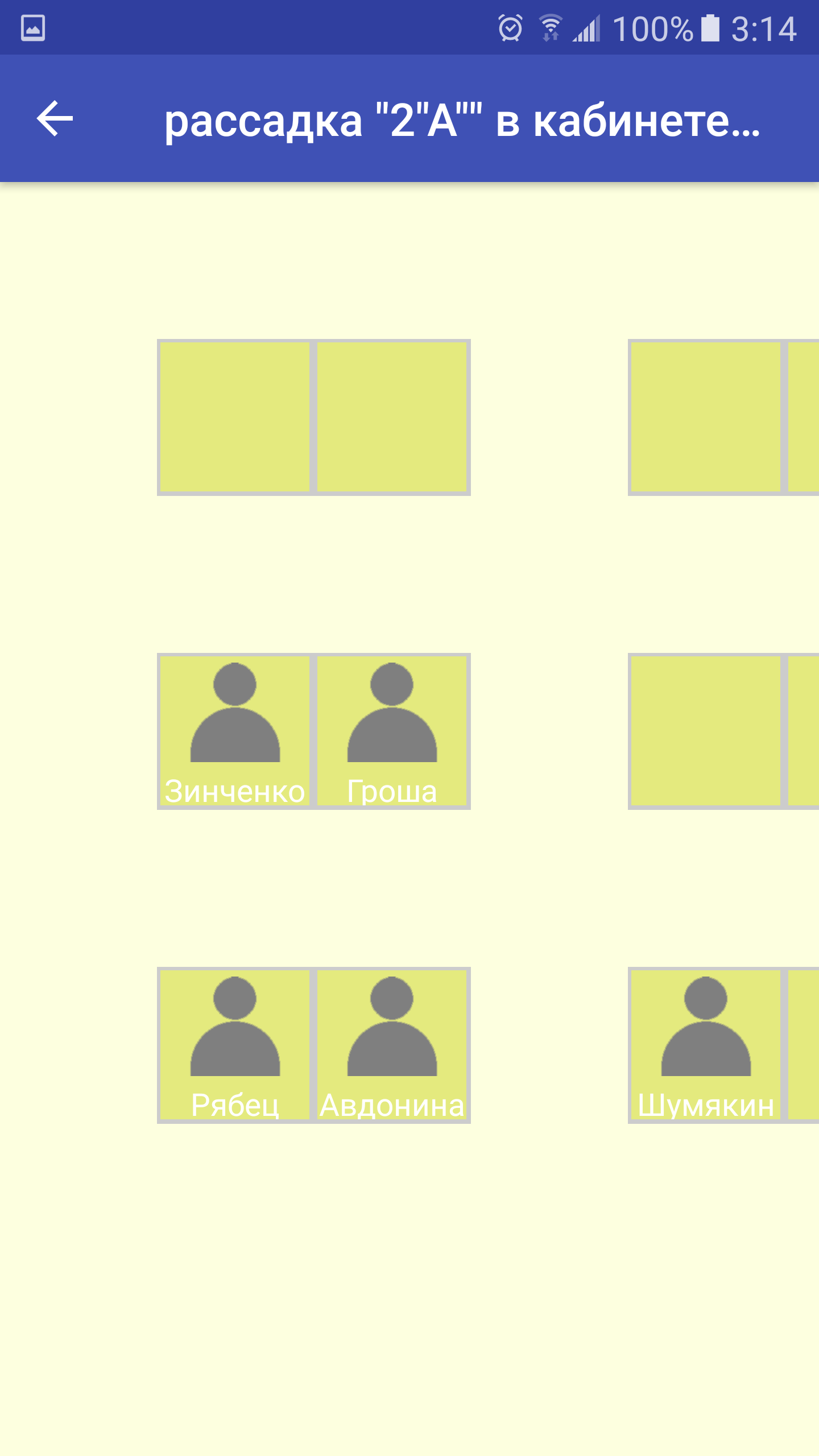


Рисунок 28 – редактор рассадки

Приложение выводит парты в выбранном кабинете и рассаженных учеников выбранного класса. Если не все ученики рассажены, то на всех свободных местах появляется кнопка «добавить».



Рисунок 29 – редактор рассадки – «Добавить»

При нажатии на кнопку «добавить» выводится диалог со списком не рассаженных учеников, а кнопка «добавить» на выбранном месте исчезает.

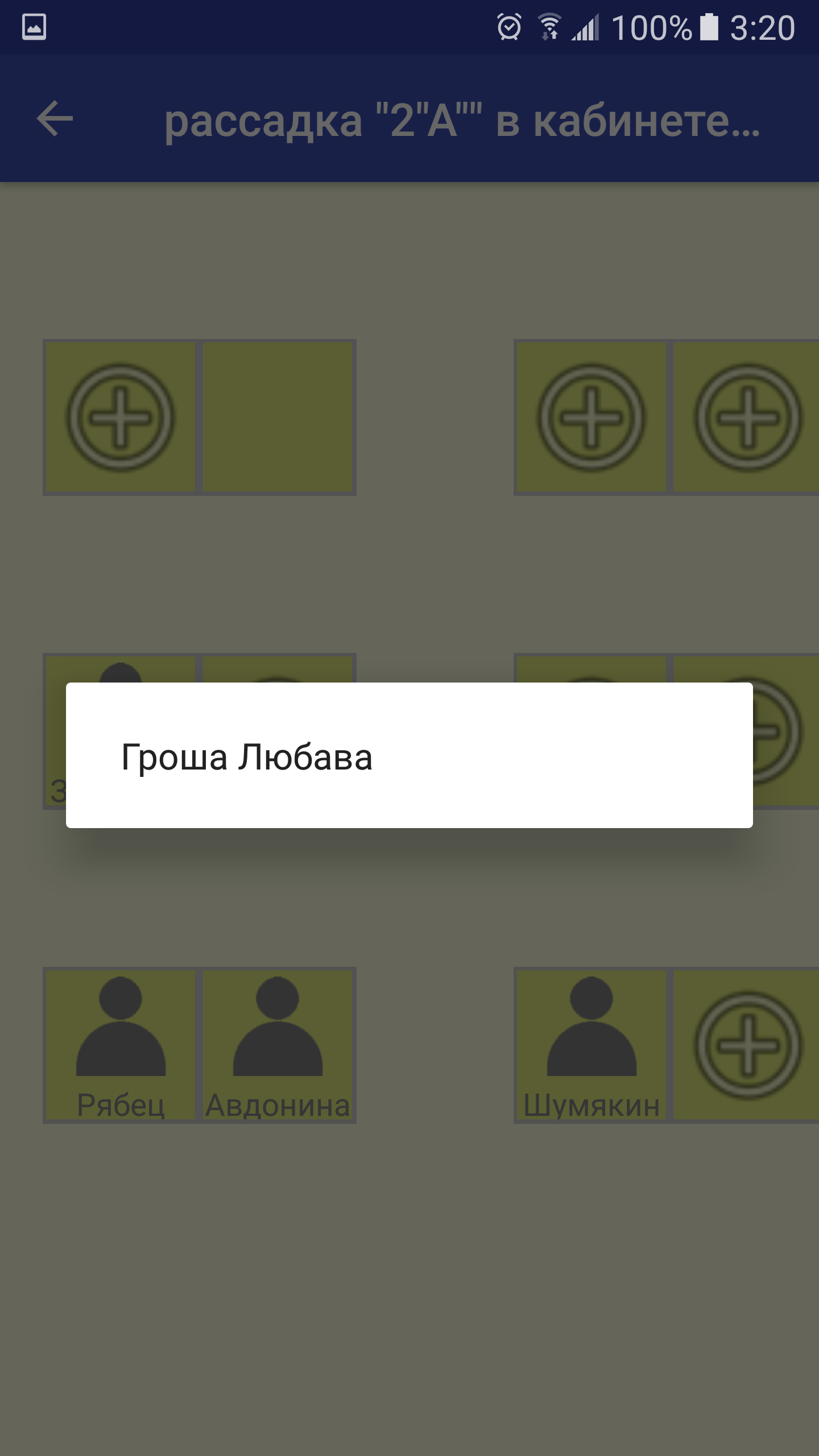


Рисунок 30 – редактор рассадки – список не рассаженных

После выбора ученика диалог закрывается, а ученик добавляется на выбранное место и сохраняется в базу данных.

Рассмотрим урок. При нажатии на урок в расписании дня открывается активность урока с переданными в неё учениками и классом.

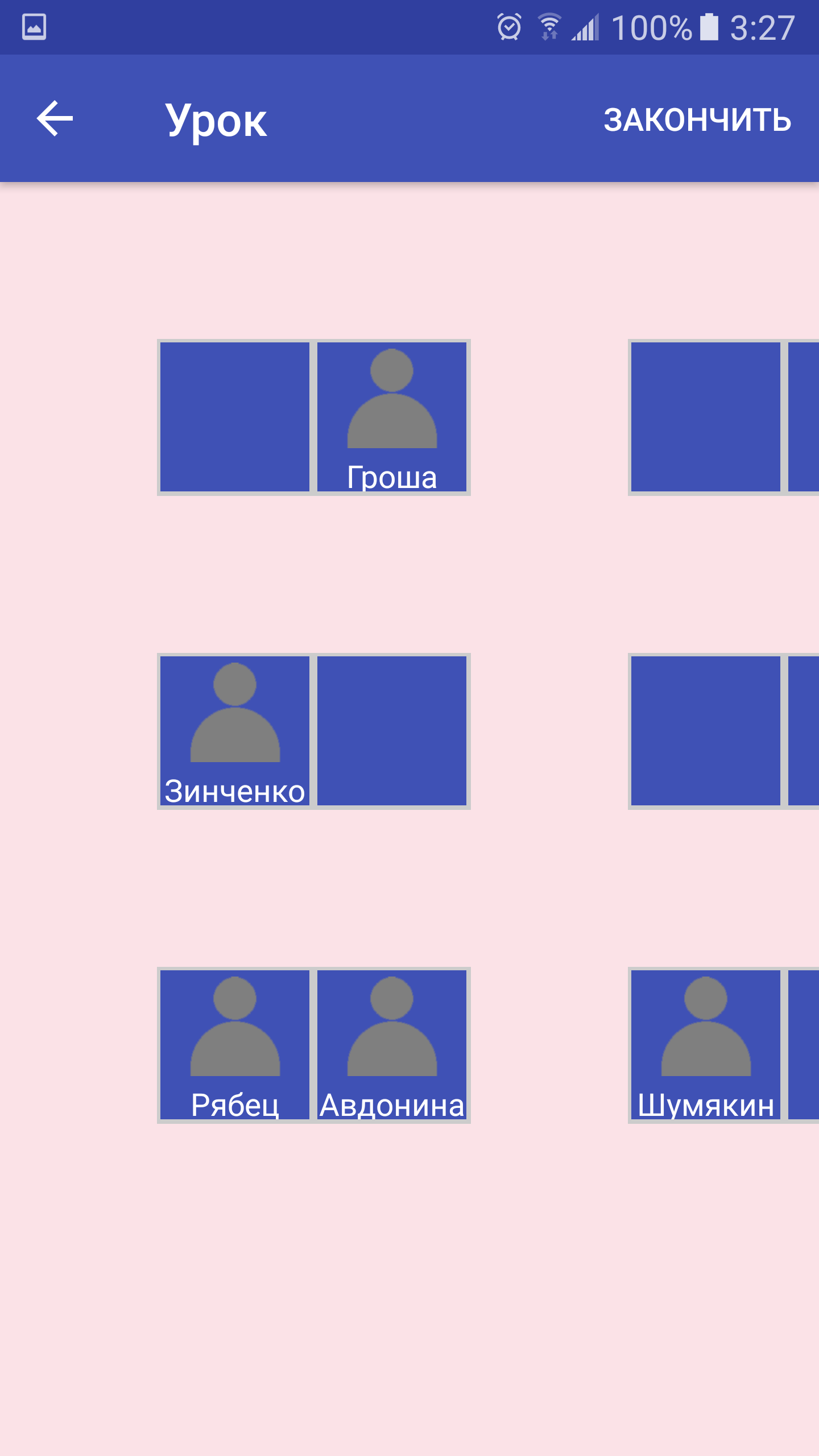


Рисунок 31 – урок

При нажатии на ученика к его оценке добавляется один балл.



Рисунок 32 – урок – 1 балл

При долгом нажатии открывается контекстное меню с выбором оценки(нельзя поставить оценку которая уже стоит).

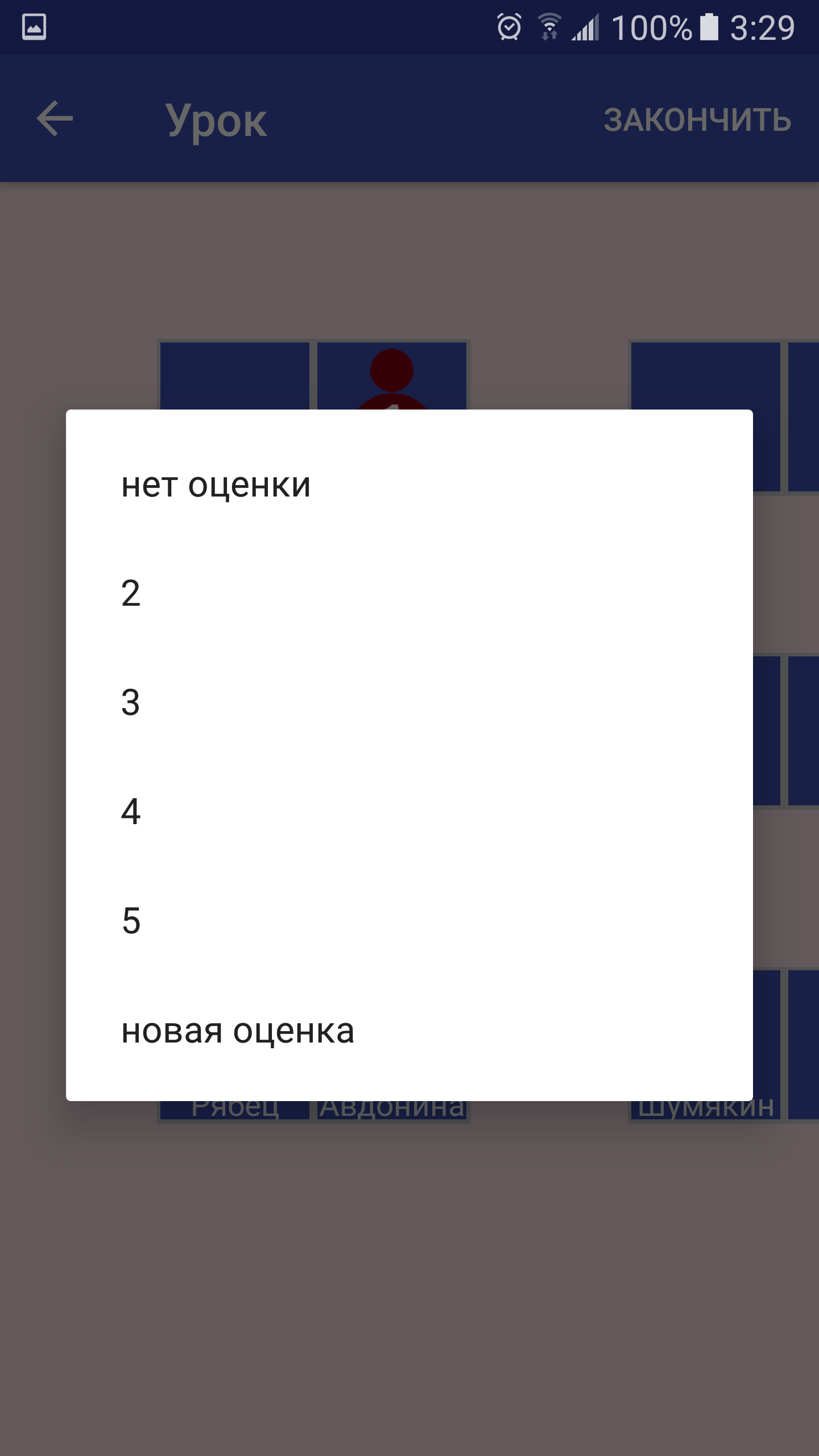


Рисунок 33 – урок – выбор оценки

Пункт «новая оценка» сохраняет оценку у ученика и даёт возможность поставить другую оценку.

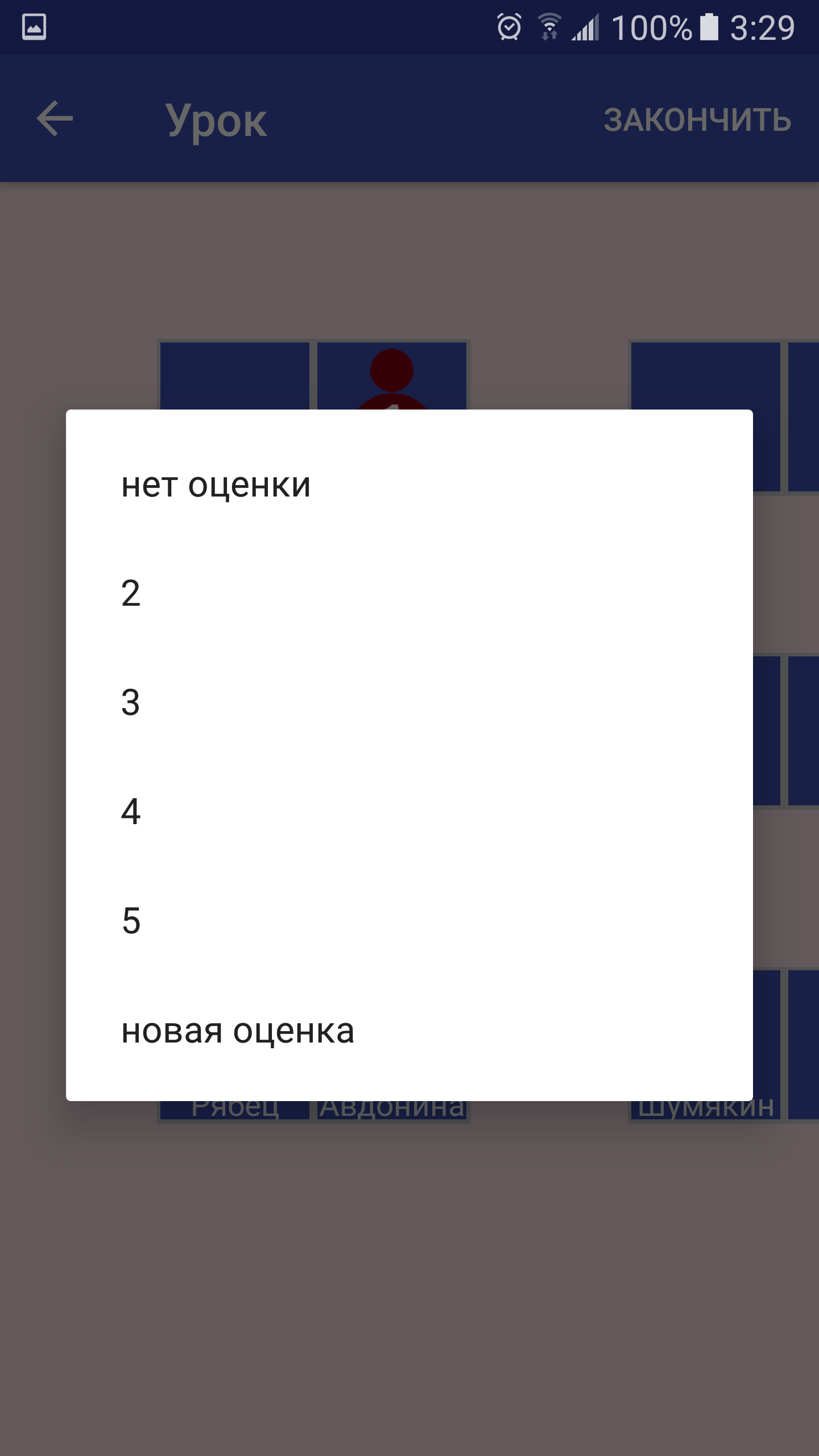


Рисунок 34 – урок – оценка сохранена

Таким образом у ученика на уроке может быть максимум 3 оценки.

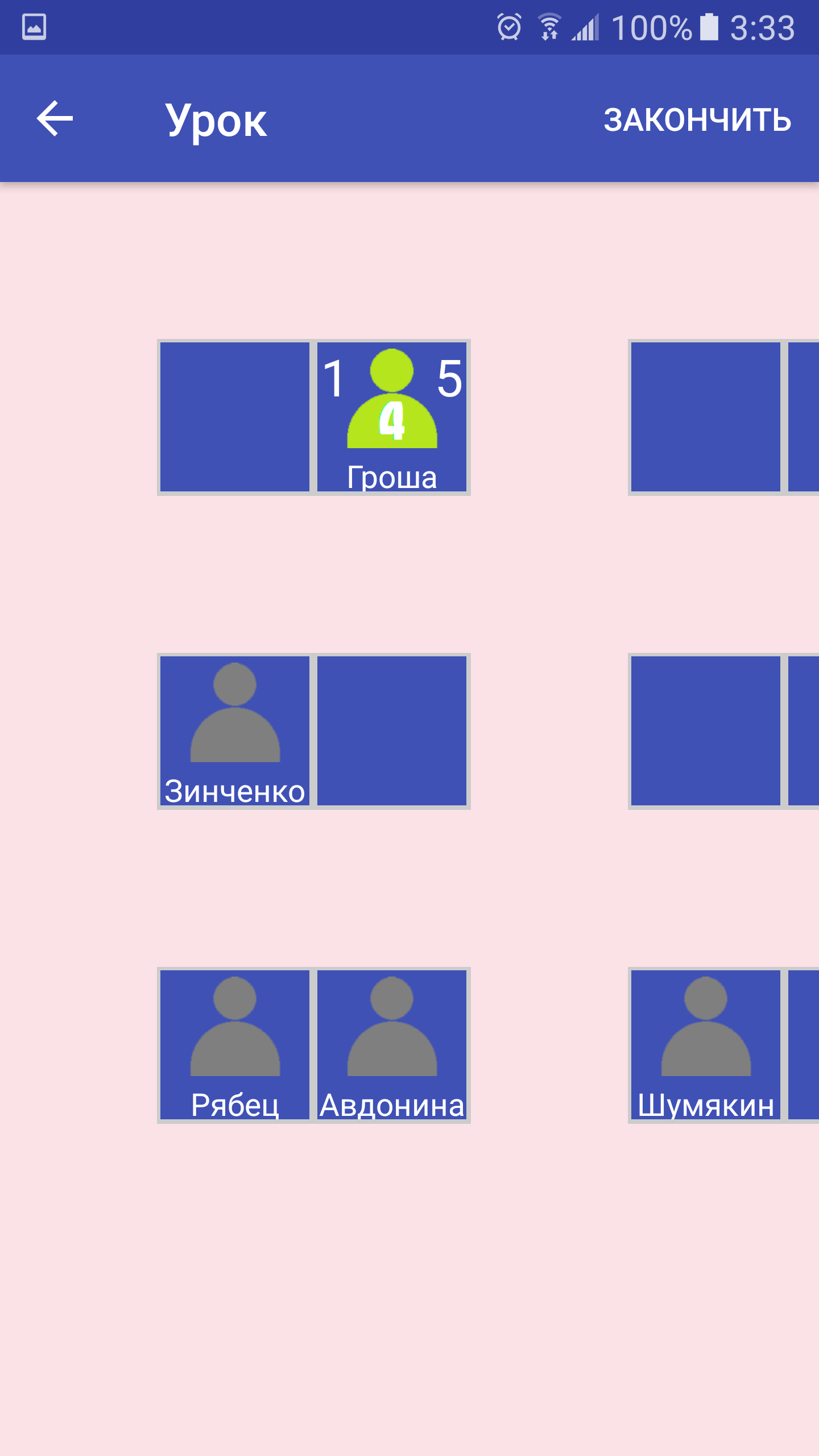


Рисунок 35 – урок – 3 оценки

После выставления оценок, пользователь нажимает кнопку «закончить урок». Происходит переход на активность итогов урока. Приложение выводит список учеников с оценками.

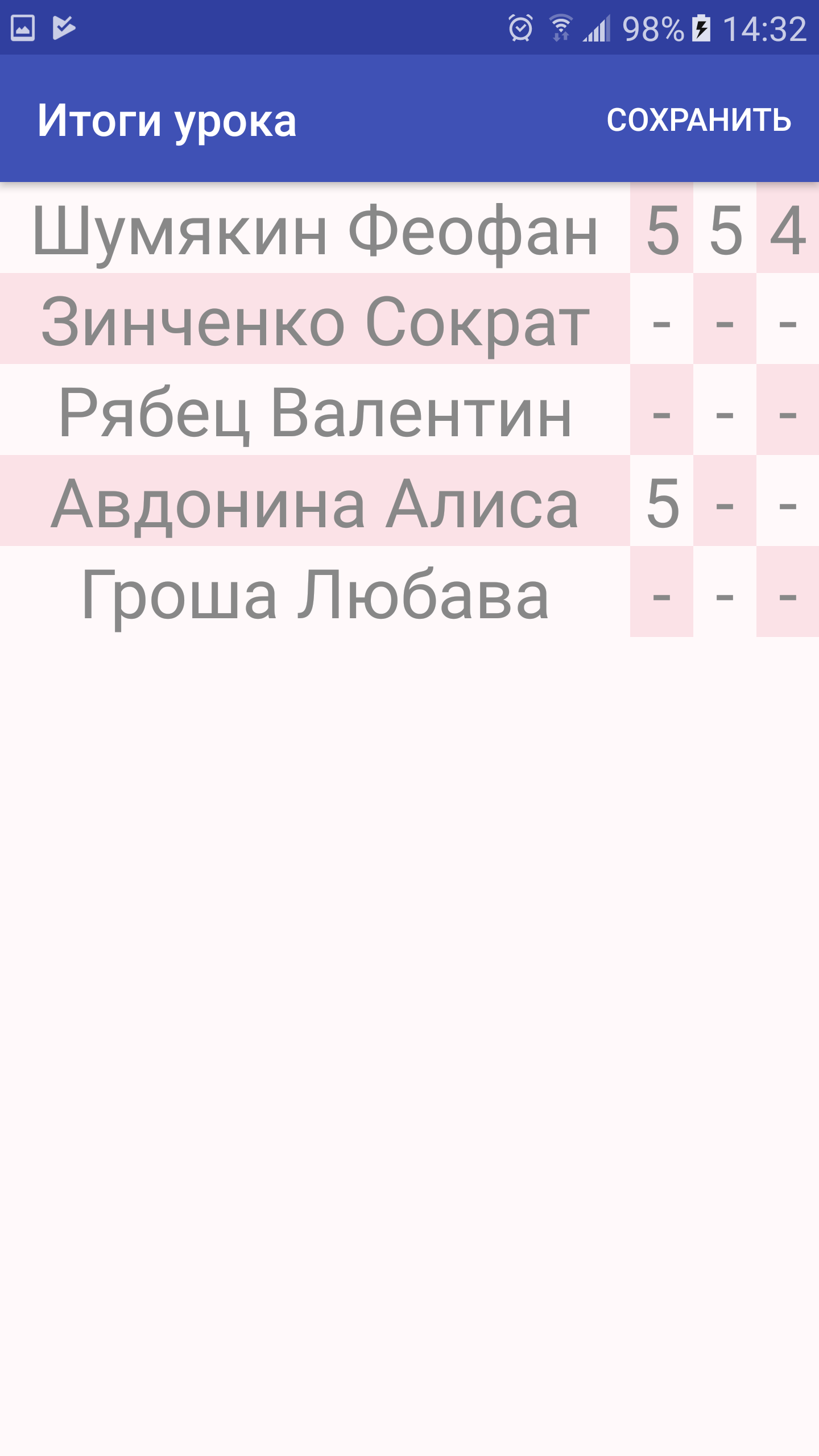


Рисунок 35 – Итоги урока

Пользователь нажимает кнопку «сохранить» оценки сохраняются в базу данных.

**3.3Локализация**

Если необходимо иметь несколько языков в приложении, то достаточно создать каталог values-xx в папке res/, где вместо xx подставить буквы страны, а в самом values-xx создать файл string.xml, в который добавлять строки для необходимых языков.

**4 Выбор стратегии тестирования и разработки тестов**

Существует 3 стратегии тестирования программного продукта: ручное тестирование. тестирование по принципу «белого ящика», тестирование по принципу «черного ящика».

Ручное тестирование эффективно. когда его проводят несколько человек, не задействованных в разработке программного продукта, поэтому, ввиду сложности организации такой группы, данный метод тестирования не был задействован.

Тестирование по методу «белого ящика» также не является необходимым для данной программы ввиду отсутствия сложных вычислительных алгоритмов, алгоритмов поиска и сортировки.

В процессе разработки был применен метод тестирования по принципу «черного ящика», а точнее его наиболее подходящий для данной программы вариант «предположение об ошибке».

# ЗАКЛЮЧЕНИЕ

В итоге разработки было реализовано приложение «Teachers assistant». Разработанная программа способна осуществлять все функции, указанные в техническом задании.

Кроме того, приложение имеет огромное количество потенциальных возможностей для усовершенствования. К примеру добавление просмотра оценок из БД, или вступительного описания.

В ходе разработки был получен опыт работы с OS Android, Android Studio и реляционными БД на примере SQLite

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ:

Сайты:

<https://developer.android.com/index.html>

<http://developer.alexanderklimov.ru/android/>

<https://habrahabr.ru>

<http://www.cyberforum.ru>

<https://stackoverflow.com>

<http://www.fandroid.info>