Муниципальное автономное общеобразовательное учреждение  
 «Инженерный лицей №83 имени Героя Советского Союза Пинского Матвея Савельевича Уфимского государственного нефтяного технического университета»

городского округа город Уфа Республики Башкортостан

Творческий проект по теме:

Разработка мобильного приложения для помощи в учёбе

Обучающийся:

Гагарина Алёна Александровна

Руководители проекта:

Гильдин Александр Григорьевич

Дружинская Елена Владимировна

г. Уфа

2023

ОГЛАВЛЕНИЕ

[ОГЛАВЛЕНИЕ 2](#_Toc132795729)

[ВВЕДЕНИЕ 3](#_Toc132795730)

[ОСНОВНАЯ ЧАСТЬ 3](#_Toc132795731)

[ГЛАВА 1. ПЛАНИРОВАНИЕ СТРУКТУРЫ И ФУНКЦИОНАЛА МОБИЛЬНОГО ПРИЛОЖЕНИЯ. 3](#_Toc132795732)

[ГЛАВА 2. ВЫБОР СПОСОБА РЕАЛИЗАЦИИ. 4](#_Toc132795733)

[ГЛАВА 3. РАЗРАБОТКА МОБИЛЬНОГО ПРИЛОЖЕНИЯ 4](#_Toc132795734)

[ЗАКЛЮЧЕНИЕ 7](#_Toc132795735)

[СПИСОК ЛИТЕРАТУРЫ. 8](#_Toc132795736)

# ВВЕДЕНИЕ

* 1. Описание проекта и постановка проблемы.

Одной из основных задач для многих студентов и школьников является выполнение домашних заданий. Поэтому многим из них необходимо правильно распоряжаться своим временем и уметь структурировать свои дела. В современном мире большинство молодых людей предпочитаю мобильные устройства для записи и хранения информации. В связи с этим, мобильное приложение для отслеживания записи домашнего задания кажется одним из удобнейших вариантов.

* 1. Актуальность проекта.

Большинство приложений такого типа в открытом доступе предназначены для англоязычной системы обучения. Также они нередко платные или с рекламой, поэтому мой проект будет являться хорошим аналогом.

* 1. Цель и задачи проекта.

Цель: Изучить методы написания мобильного приложения для Android, разработать понятное приложение для записи домашнего задания и удобного просмотра необходимых данных для учёбы.

Основные задачи проекта:

1. Изучить особенности написания приложения и выбрать среду разработки.
2. Выяснить какие функции для помощи с организацией учёбы необходимы.
3. Систематизировать полученные данные.

Разработать и создать приложение.

* 1. Ожидаемые результаты.

Разработка и запуск приложения.

# ОСНОВНАЯ ЧАСТЬ

# ГЛАВА 1. ПЛАНИРОВАНИЕ СТРУКТУРЫ И ФУНКЦИОНАЛА МОБИЛЬНОГО ПРИЛОЖЕНИЯ.

Исходя из поставленной цели, мы разработали концепцию работы и содержание будущего приложение.

Так приложение будет включать в себе данный функционал:

1. Создание:
2. Создание предметов — пользователь описывает учебный предмет.
3. Создание расписания на неделю — пользователь заполняет предметами расписание для каждого дня.
4. Запись домашнего задания — пользователь выбирает день и предмет, а после записывает домашнее задание
5. Просмотр:
6. Предметы — пользователь может просматривать информацию по предметы.
7. Расписание — пользователь может просматривать созданное расписание и домашнее задание для каждого предмета.

3. Дополнительно:

1. Таймер по принципу Tomato — умение правильно контролировать своё время — это одно из главных правил тайм-менеджмента. Мы решили использовать технику «Tomato», разработанную Франческо Чирилло в конце 1980—х годов. Этот метод использует таймер для разбивки работ на набор интервалов, разделённых перерывами. Техника «Tomato» повышает продуктивность за счёт регулярных коротких запланированных перерывов.

2. Удаление записей — пользователь может удалять предметы, расписание по дню и т.д.

4. Информация о приложении — пользователь сможет увидеть описание приложения, а так же ссылки на используемые при создании ресурсы.

Все указанные настройки будут хранить в базе данных.

# ГЛАВА 2. ВЫБОР СПОСОБА РЕАЛИЗАЦИИ.

После планирования функционала приложения, мы можем выделить инструменты для его реализации: запись, редактирование и просмотр базы данных, возможность пользоваться таймером, удобная выгрузка списков для просмотра информации.

Так как платформой этого приложения будет Android - нам подойдут следующие среды разработок: Eclipse или Android Studio. Учитывая что Android Studio является официальной средой разработки от Google и наиболее понятной для новичков, мы выбрали ей в качестве фреймворка для нашего приложения. В данной среде есть два языка программирования: Java и Kotling. Мы выбрали Java - наиболее популярный и привычный для понимания. Базу данных мы реализуем через встроенный класс SQLiteOpenHelper. Для таймера воспользуемся классом MediaPlayer для воспроизведения звукового сигнала и классом CountDownTimer для обратного отсчета.

Все функции данного приложения будут реализованы через android класс Fragment для экономии память и увеличения скорости работы приложения. Переключение между функциями приложения будет выполнено с помощью специального компонента интерфейса - Navigation Drawer, представляющего собой меню, выезжающее на экран при нажатии кнопки или плавному проведению пальца.

Так же нам понадобится диалоговое окно, для подтверждения или просмотра информации. Диалоговое окно создаётся через тип данных Dialog и заполняется при помощи классов-наследников от DialogFragment через тип данных AlertDialog.Builder, устанавливающий заголовок, текст, иконку окна и нейтральную, позитивную, негативную кнопку со своим Listener - это уведомляемый о некотором событии объект.

# ГЛАВА 3. РАЗРАБОТКА МОБИЛЬНОГО ПРИЛОЖЕНИЯ

Из главного меню приложения пользователь сможет переключаться на другие экраны, в которых и будут реализованы опции приложения, а именно: создание и просмотр предметов и расписания, запись домашнего задания, таймер, удаление данных приложения и настройки, а так же просмотр информации о приложении.

Изображения всех визуальных компонентов мобильного приложения представлены в приложении данного проекта.

**Параграф 1. Таймер по принципу «Tomato».**

Для реализации таймера мы создали фрагмент, где находится запись времени на учёбу, а так же перерывы, они представляют собой EditText - текстовые поля с возможностью заполнения только цифрами, это ограничение реализуется через методы inputType со значением "number". А само время указывается в минутах. Изначально эти поля заполнены «25» и «5» соответственно - это наиболее известные промежутки для выполнения поставленных задач. Сверху располагается TextView - поле, предусмотренное для показа обратного отсчёта времени. По завершению таймера так же появляется надпись о завершение промежутка времени и воспроизводится звуковой сигнал при помощи класса MediaPlayer.

**Параграф 2. Создание и обновление базы данных.**

Для удобства мы будем создавать базу данных и обращаться к ней в отдельном классе DBHelper, который наследуется от SQLiteOpenHelper - это вспомогательный класс, встроенный в пакет android.database.sqlite.SQLiteDatabase.

В переопределенном методе onCreate(), который вызывается лишь один раз в процессе жизненного цикла приложения, создаём базу данных и таблицы:

1. Таблицу предметов, состоящую из целочисленного первичного ключа, текстовых полей: название предмета, кабинет, имя учителя, описание;
2. Таблицы для каждого учебного дня недели, состоящие из целочисленных первичных ключей, целочисленных номерах уроков, текстовых названий предметов и домашнего задания.

Далее мы будем открывать эту базу данных, используя тип данных SQLiteDatabase с методом getWritableDatabase() для чтения и изменения таблиц.

Для обновления и добавления записей в таблицу нам так же понадобится объект типа ContentValues - данный объект является словарём, состоящем из набора пар: имя столбца, новое значение. Добавление таких пар осуществляется через метод put(). Добавление новых записей в базу данных происходит через метод insert, принимающий имя таблицы, куда необходимо добавить запись, значение столбца, которое может равняться null, если ContentValues окажется пустым, элемент типа ContentValues.

Обновление происходит через метод update с параметрами: имя таблицы, элемент типа ContentValues, условие обновление, значение аргументов для условия.

Удаление записей в таблице производится через метод delete с параметрами: название таблицы, условие удаления, значение аргументов для условия.

**Параграф 3. Запись предметов.**

Пользователь записывает в текстовые поля EditText название предмета, имя учителя, кабинет и описание. Каждое поле имеет подсказу для правильного ввода, реализуемое через метод hint. Поле для ввода описания позволяет добавить многострочный текст через метод singleLine со значением "false", а так же прокручивать его, используя метод scrollbars. При нажатии кнопки «Добавить» мы приводим название предмета в вид “Первая буква заглавная, остальные строчные” при помощи методов .toUpperCase() и toLowerCase() соответственно, далее через класс

DBHelper проверяем, есть ли такой предмет в базе данных. Для этого при помощи оператора запроса “Select” узнаем количество записей где имя предмета равно введённому и если таких записей нет - создаём и через всплывающее уведомление Toast выводим сообщение о успешном добавление предмета, а так же обнуляем все поля в самом фрагменте. Если такой предмет уже добавлен - сообщаем об этом пользователю.

**Параграф 4. Запись расписания.**

В данном фрагменте пользователь выбирает день и урок через элемент выпадающего списка Spinner - данный элемент получает String массив со значениями: дни недели и предметы, создаётся адаптер AdapterView - он упрощает данных с элементами управления, подключается метод onItemSelected, отслеживающий какой элемент выбран. Пользователь вводит номер урока - он не может повторяться в EditText с методом inputType и параметром "date", позволяющем вводить только целочисленные данные. Через класс DBHelper проверяем, есть ли такой номер урока в этом расписании, если да - выводим Toast сообщение с просьбой изменить номер урока, если нет - Toast сообщение с подтверждением о добавлении урока.

**Параграф 5. Запись домашнего задания.**

Пользователь выбирает день через элемент Spinner; далее второй Spinner заполняется массивом из названий уроков в этот день, названия не повторяются и домашнее задание для одинаковых предметов в один и тот же день одинаковое. Пользователь заполняет поле EditText для ввода описания, оно позволяет добавить многострочный текст через метод singleLine со значением "false", а так же прокручивать его, используя метод scrollbars.

В класс DBHelper передаётся день, название предмета и домашнее задание.Там определяется нужная таблица и через метод update обновляются значения полей, отведённых для записи домашнего задания, где поле с названием предмета соответствует переданному названию.

Так же эту функцию можно вызвать с главного экрана приложения.

**Параграф 6. Просмотр предметов.**

Этот фрагмент наследуется от класса ListFragment и позволяет создавать список данных. Для этого создаётся AdapterView с параметрами: Context - интерфейс, предоставляющий глобальную информацию о среде приложения, html разметка для списка, список предметов в формате ArrayList - он возвращается из класса DBHelper. Даллее метод setListAdapter() с параметром AdapterView подключает полученный аdapter для отображения на эране пользователя. Пользователь видит список уже добавленных предметов. При нажатии, отслеживаемый встроенным методом onListItemClick, появляется диалоговое окно с подробным описанием предмета.

**Параграф 7. Просмотр расписания.**

Пользователь выбирает день недели из элементов Spinner, при помощи метода класса DBHelper определяются предметы в этот день, отсортированные по возрастанию их номеров уроков, через параметр ORDER BY в запросе к базе данных. Как и в случае просмотра предметов, создаётся AdapterView, он подключается методом setListAdapter(). При нажатии появляется диалоговое окно, отображающее название предмета, кабинет, учителя, описание и домашнее задание. Эти данные получены из метода класса DBHelper: он по названию урока возвращает его данные и по имени дня и названию предмета определяет нужную таблицу, а далее находит, через запрос SELECT, в ней поля где имя предмета соответствует переданному.

**Параграф 8. Удаление записей.**

В данном фрагменте при нажатии на различные кнопки появляются диалоговые окна с просьбой подтвердить действия, если пользователь соглашается, нажимая на кнопку, ответственную за позитивную реакцию, - выполняются ниже описанные события, иначе - ничего не происходит. Удалять записи в базе данных можно по различным критериям:

* Всё домашнее задание на день - Выбор дня реализуется через элемент выпадающего списка Spinner - вызыватся метод класса DBHelper, который по переданному названию дня выбирает соответствующую таблицу данных и перезаписывает каждое поле, отведённое для записи домашнего задания, в пустые значения. Эти действия реализуется через метод “update”, чью работу мы описали во втором параграфе.
* Расписание для одного дня - Выбор дня реализуется через элемент Spinner, вызыватся метод класса DBHelper и через метод delete удаляется нужная таблица.
* Предмет - Пользователь выбирает название нужного предмета через элемент Spinner. В методе класса DBHelper через метод delete удаляются все записи в таблице предметов, где поле с именем предмета соответствует преданному.
* Все предметы - В методе класса DBHelper через метод delete удаляются все значения полей в таблице предметов.
* Всё расписание - В методе класса DBHelper через метод delete удаляются все значения полей во всех таблицах для хранений расписания и домашнего задания.
* Всё - В методе класса DBHelper через метод delete удаляется вся информация во всех таблицах.

**Параграф 9. О приложении.**

В данном фрагменте в элементе TextView пользователь может прочитать основную информацию о приложении, а так же найти ссылки на сайты, откуда были взять ресурсы для создания приложения.

ЗАКЛЮЧЕНИЕ.

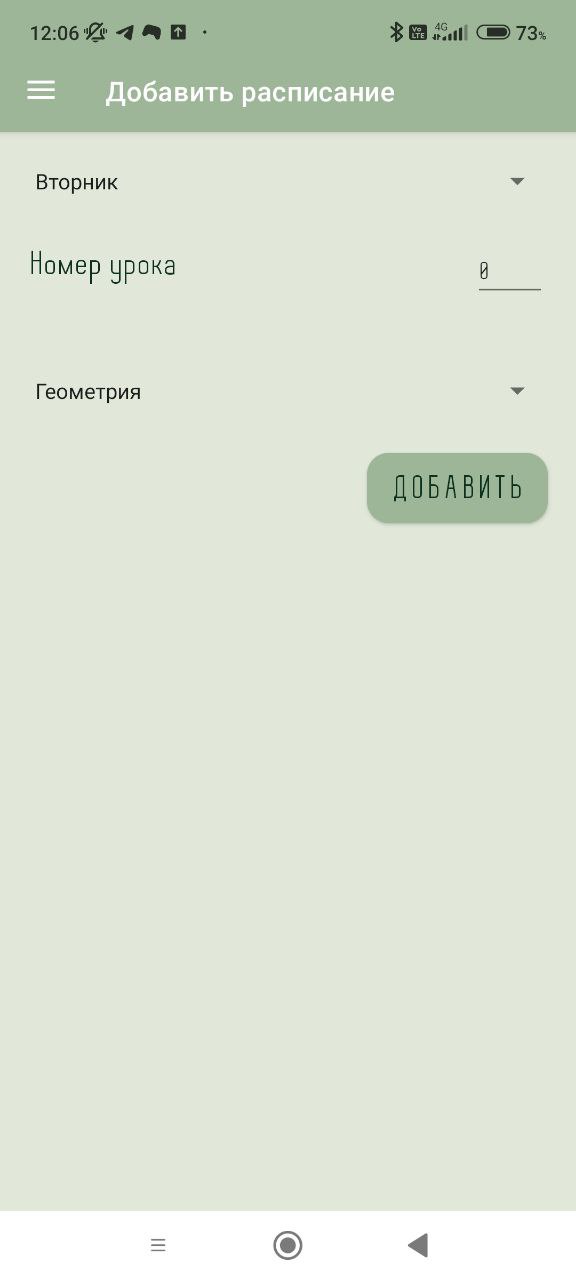
Во время работы над данным проектом мы изучили технологию написания мобильного приложения в Android Studio, а так же отдельно изучили навыки работы с SQL базами данных. Выполняя поставленные задачи, мы добились изначальной цели и в результате создали мобильное приложение, позволяющее контролировать учебное расписание и запись домашнего задания. Далее мы рассматриваем возможность продвижения и улучшения нашего продукта, а также публикации на различных интернет ресурсах, что поможет раскрыть потенциал нашей работы и упростит учёбу для многих пользователей.

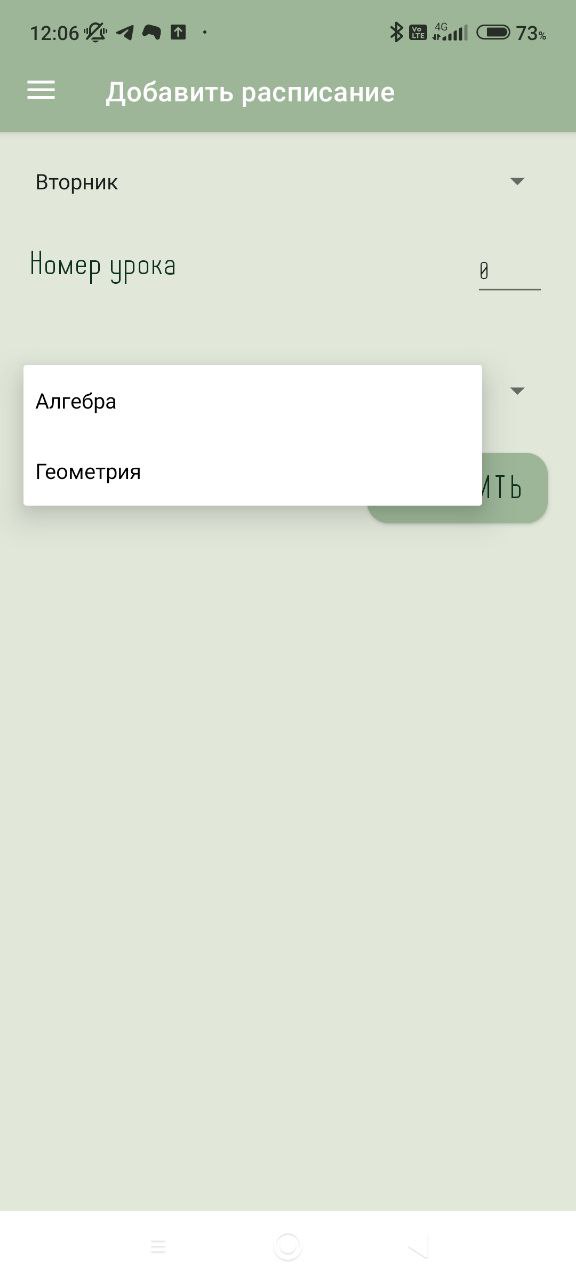
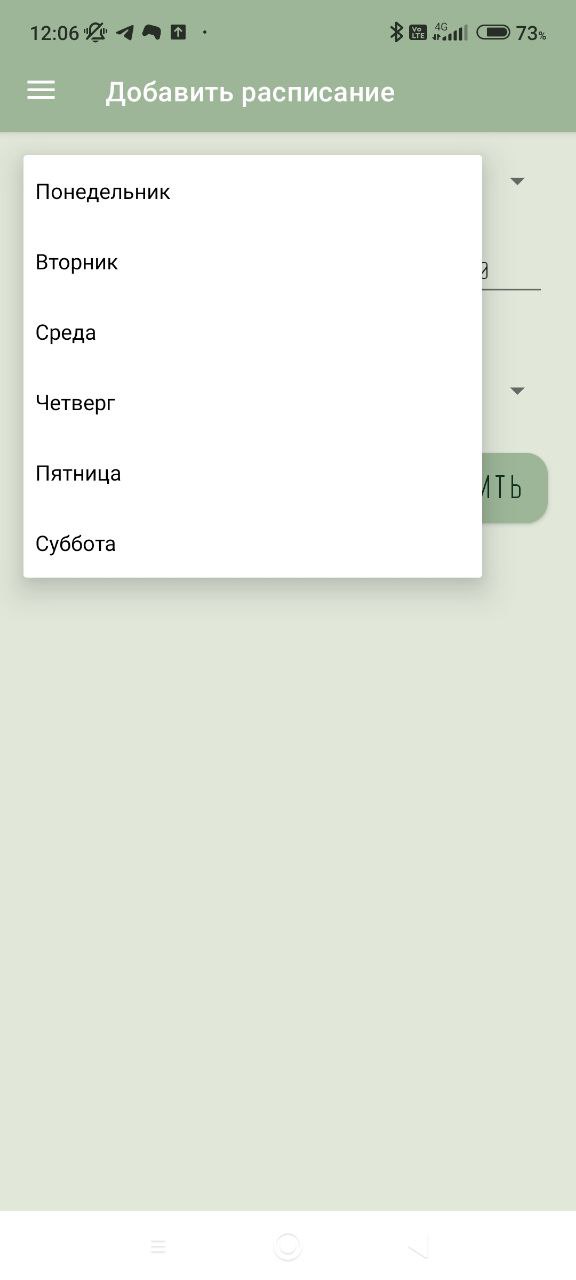
Посмотреть полученный проект можно по ссылке [https://disk.yandex.ru/d/nD0C\_X2AABOwQg](https://disk.yandex.ru/d/nD0C_X2AABOwQg.), код проекта загружен на GitHab

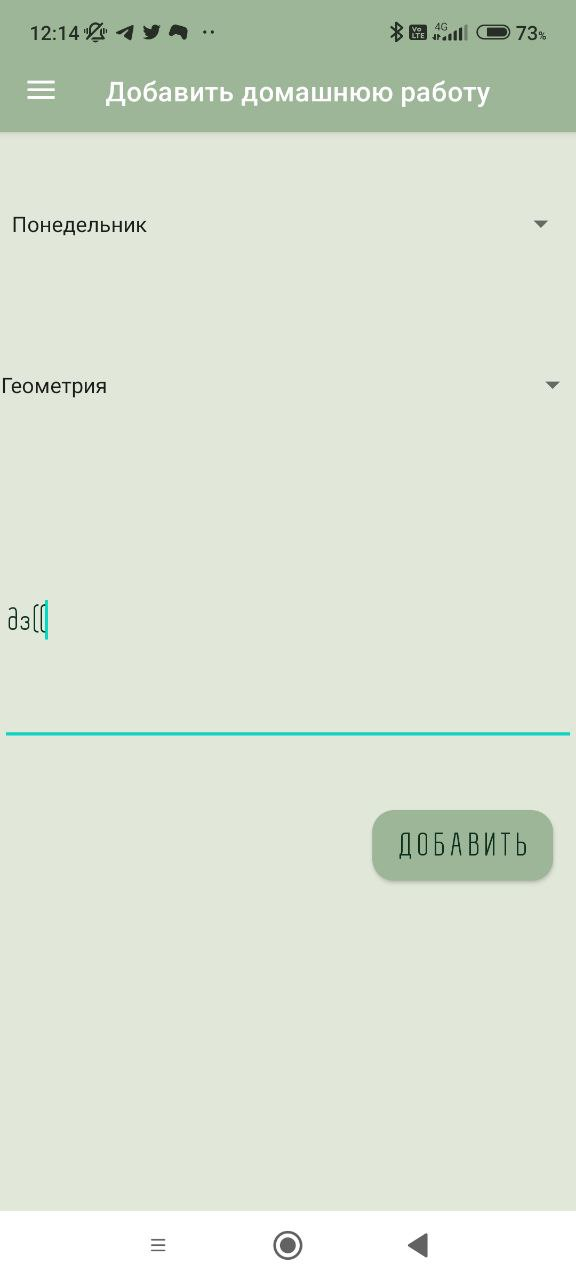
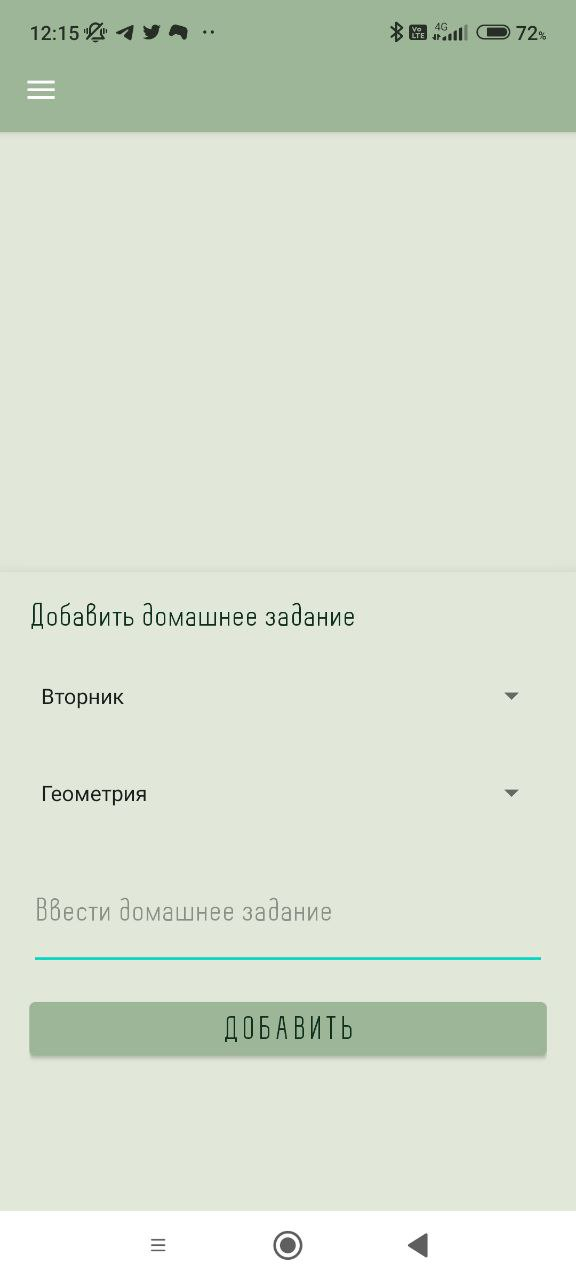
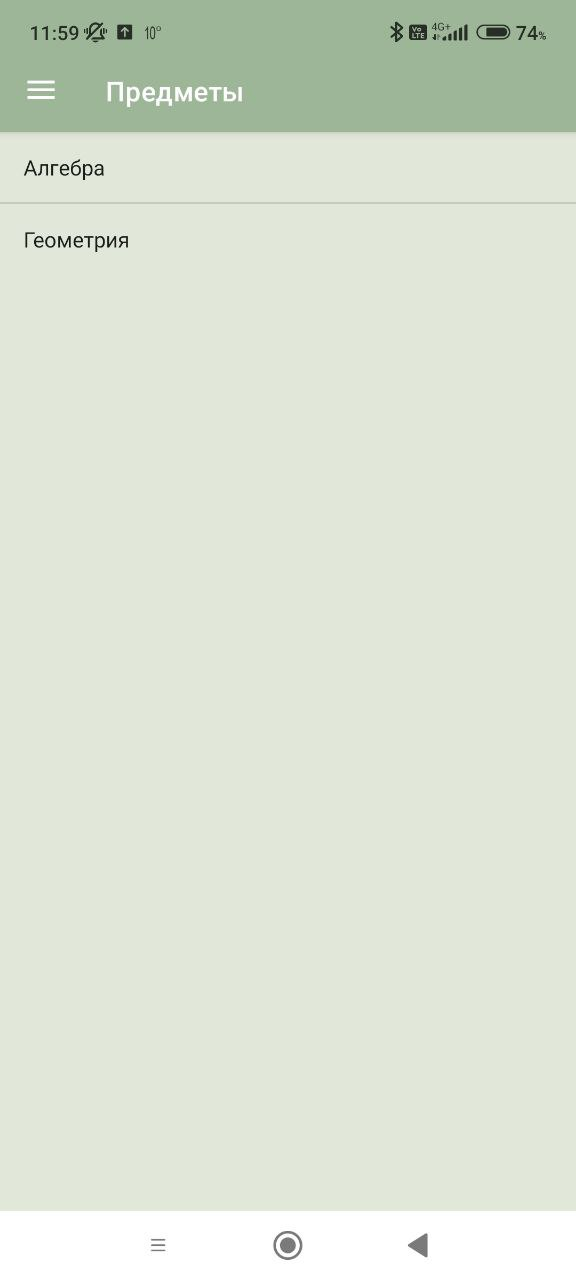
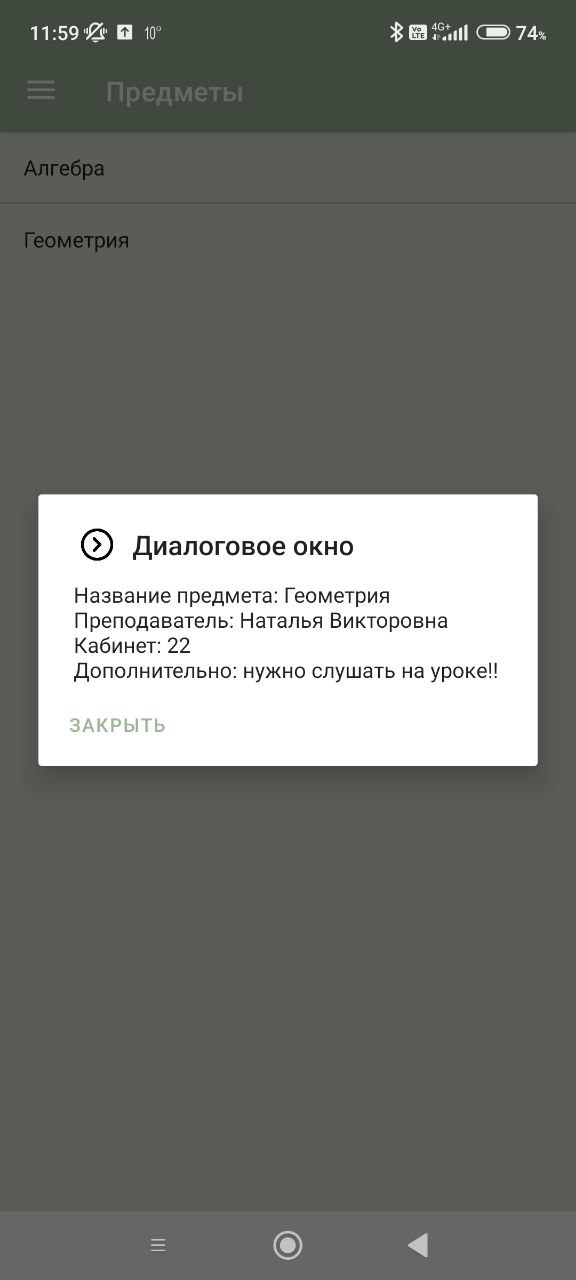
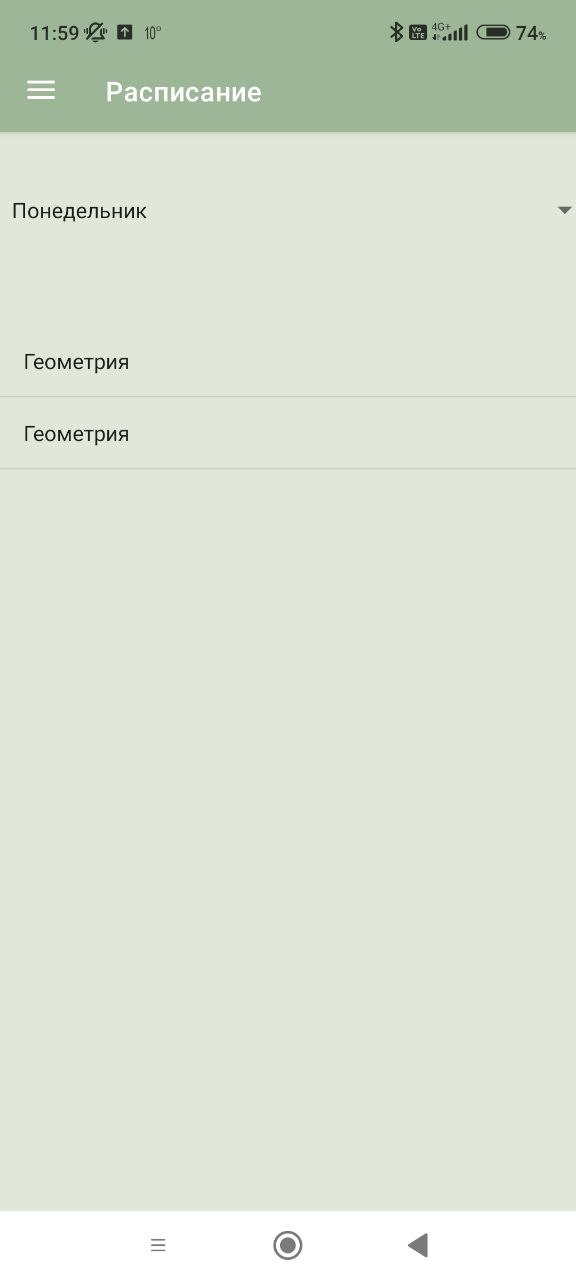
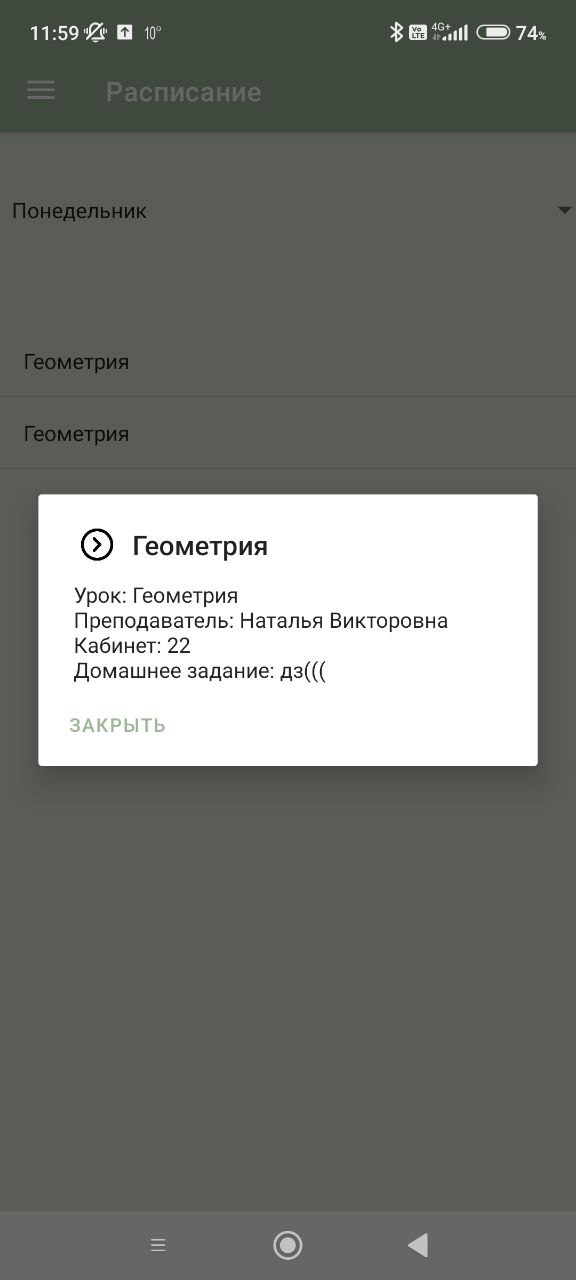
# СПИСОК ЛИТЕРАТУРЫ.

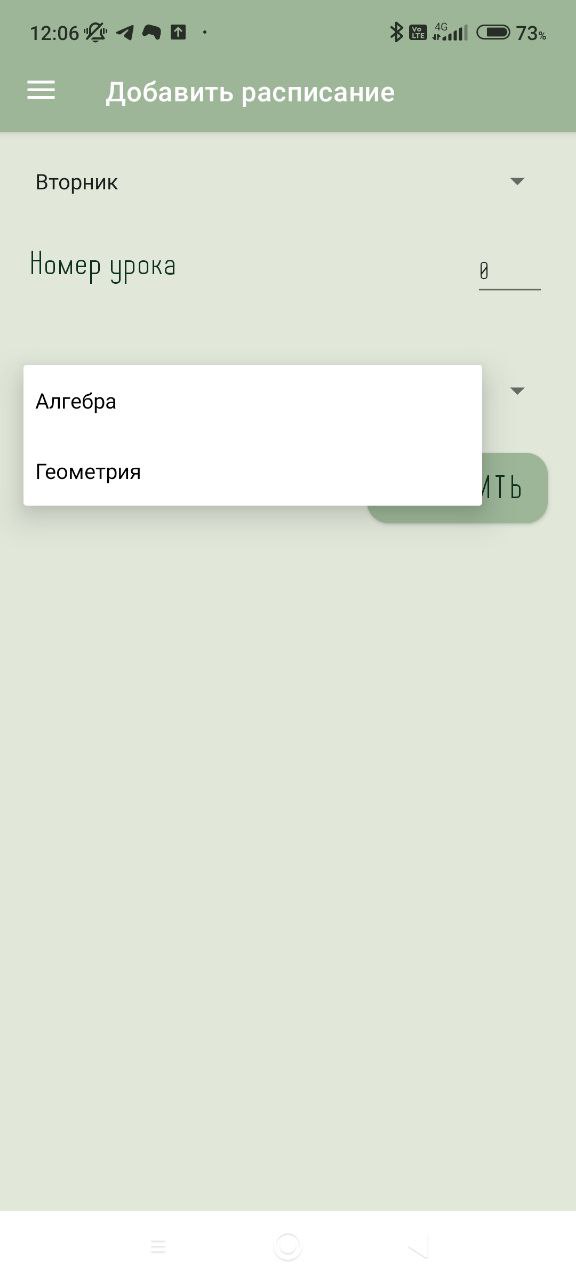
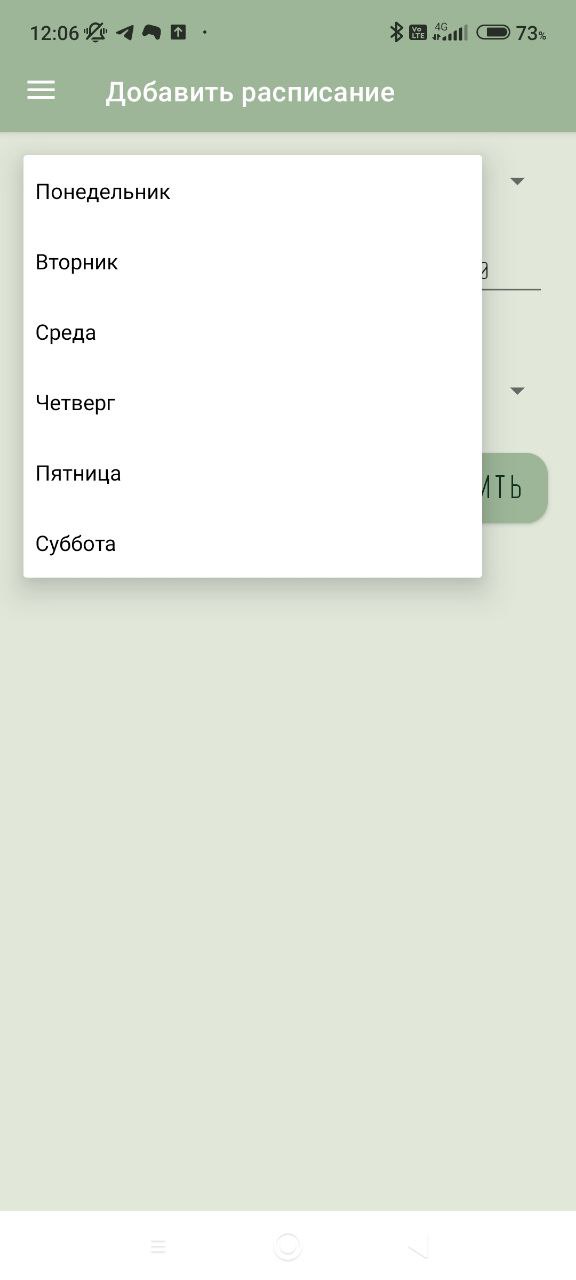
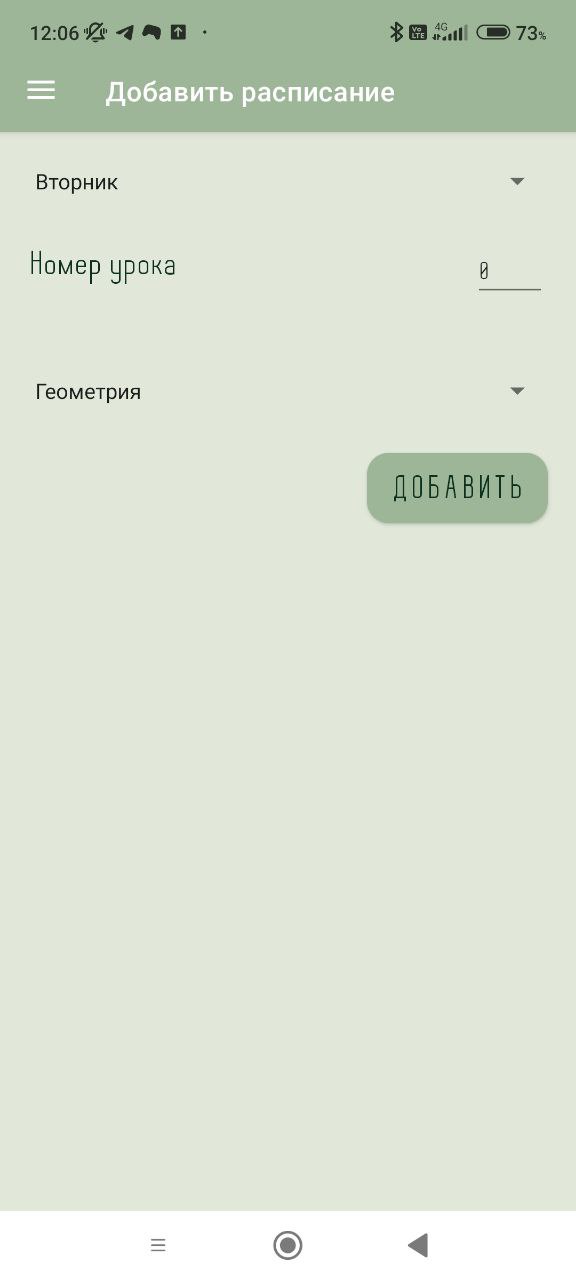
* Википедия URL <https://ru.wikipedia.org/wiki/%D0%97>
* Android for Developers <https://developer.android.com/>
* Сайт Александра Климова <https://developer.alexanderklimov.ru/android/>
* GitHub <https://github.com/>
* Harb <https://habr.com/ru/all/>
* Stackoverflow <https://ru.stackoverflow.com/>
* Fonts <https://ffont.ru/fonts>
* Flaticon <https://www.flaticon.com/ru/>

# ПРИЛОЖЕНИЕ

****



****

Посмотреть полученный проект можно по ссылке [https://disk.yandex.ru/d/nD0C\_X2AABOwQg](https://disk.yandex.ru/d/nD0C_X2AABOwQg.), код проекта загружен на GitHab, по ссылке https://github.com/Alenaaaaaaaa/Iskids.

