**Министерство образования Республики Беларусь**

**БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**МЕХАНИКО-МАТЕМАТИЧЕСКИЙ ФАКУЛЬТЕТ**

**Кафедра Веб-технологий**

**РАЗРАБОТКА МОБИЛЬНОГО ПРИЛОЖЕНИЯ ДЛЯ ПЛАНИРОВАНИЯ ВРЕМЕНИ**

Курсовая работа

Беляковича Дмитрия Александровича

студента 2 курса

специальность «Математическое и программное обеспечение мобильных устройств»

Научный руководитель:   
старший преподаватель  
Пономаренко Юлия Игоревна

Минск 2020

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc40982472)

[1 АНАЛИЗ АНАЛОГОВ ПРИЛОЖЕНИЯ 4](#_Toc40982473)

[1.1 Microsoft To Do 4](#_Toc40982474)

[1.2 Todoist 5](#_Toc40982475)

[2 ЦЕЛИ И ЗАДАЧИ 7](#_Toc40982476)

[2.1 Цель 7](#_Toc40982477)

[2.2 Задачи 7](#_Toc40982478)

[2.3 Необходимые средства 7](#_Toc40982479)

[3 ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА 8](#_Toc40982480)

[3.1 Структура приложения 8](#_Toc40982481)

[3.2 Model 9](#_Toc40982482)

[3.3 View 9](#_Toc40982483)

[3.4 Controller 11](#_Toc40982484)

[4 ТЕСТИРОВАНИЕ 12](#_Toc40982485)

[5 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ 13](#_Toc40982486)

[5.1 Необходимые требования 13](#_Toc40982487)

[5.2 Начало работы 13](#_Toc40982488)

[ЗАКЛЮЧЕНИЕ 15](#_Toc40982489)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 16](#_Toc40982490)

# **ВВЕДЕНИЕ**

Современный мир диктует свои правила. Ему нужны универсальные люди, которые не забывают выполнять очень важную работу. Например, пользователю надо написать отчёт, подготовить курсовую и так далее. Держать всё это в голове рано или поздно станет очень сложно, а следовательно список дел и сроков сдачи может перепутаться. Пользователь, который организовывает свои списки дел, с целью запоминания списка в случае перенасыщения задачами, нуждается в средстве упорядочивания списка на своём устройстве.

Далеко не всегда есть возможность воспользоваться физическими средствами хранения информации такими как, лист бумаги или блокнота, ввиду возможного отсутствия места хранения информации (листа бумаги). Да, иногда такой способ хранения информации очень удобен, но, когда список становится очень большим, искать информацию на листах блокнота становится очень неудобно.

Целью данной курсовой работы является разработка мобильного программного средства для хранения списка задач. Программа будет включать в себя возможности организации и редактирования списка. Приложение должно функционировать без регистрации пользователя, что необходимо для лёгкости использования.

# **АНАЛИЗ АНАЛОГОВ ПРИЛОЖЕНИЯ**

В настоящий момент существует множество аналогов разрабатываемого приложения. Поиск осуществляется исключительно среди приложений, которые можно скачать в Play Store.

## **Microsoft To Do**

Приложение предоставляет возможность создавать задачи путём организации отдельных списков по темам. Есть возможность группировать списки. Для каждой отдельной задачи есть возможность установить дату выполнения, добавить подзадачи и поставить метку “Важное”. Из созданных списков надо добавлять задачи в другой список “Мой день” (Рисунок 1.1).

Приложение отличается приятной цветовой палитрой. Чтобы начать пользоваться обязательно надо авторизоваться. При наличии интернета данные синхронизируются с сервером. Присутствуют элементы управления для печати списка и распространения копии списка в различных известных сервисах, например, таких как VK, Telegram, Facebook.

Чтобы начать пользоваться этим приложением необходимо потрать некоторое время, чтобы разобраться в его логике (рисунок 1.2). Список подзадач реализован не очень хорошо. Возможности сортировки спрятаны очень далеко. При добавлении задачи вне определённого списка, она добавляется в отдельный список, поэтому в конечном итоге получается много списков, за которыми надо следить. Всё это не позволяет эффективно пользоваться реализованным функционалом.

Плюсом данного приложения является его кроссплатформенность. Все списки синхронизируются с учетной записью.

Другим преимуществом данного приложения является локализация. Сейчас оно может адаптироваться под множество регионов и пользователей.

|  |  |
| --- | --- |
|  |  |
| Рисунок 1.1 – список “Мой день” | Рисунок 1.2 – много списков |

## **Todoist**

Todoist – ещё один аналог разрабатываемого приложения.

Данное приложение отличается от ранее описанного аналога красочностью и, как мне кажется, неудачной стартовой палитрой. Есть шаблоны создания каких-то проектов и очень много разных стикеров и смайликов, которые пытаются расположить к себе пользователя, но из-за этого и так непонятный интерфейс становится более нагромождённым и, следовательно, более непонятным. (Рисунок 1.3)

Огромным плюсом является то, что Todoist предоставляет возможность отслеживать продуктивность пользователя. (Рисунок 1.4)

Есть возможность составлять список на неделю, что тоже является плюсом. (Рисунок 1.5)

|  |  |  |
| --- | --- | --- |
|  |  |  |
| Рисунок 1.3 – интерфейс Todoist | Рисунок 1.4 – отслеживание продуктивности | Рисунок 1.5 |

Так же, как и в Microsoft To Do, необходимо обязательно авторизоваться, что подразумевает синхронизацию с аккаунтом и, возможно, кроссплатформенность

На рынке существует большое количество приложений как платных, так и бесплатных, позволяющих создавать списки задач, однако каждый из рассмотренных аналогов предоставляет свой главный функционал – управление списками, предполагая, что пользователь потратит время на то, чтобы разобраться в интерфейсе, и на то, что пользователь обязательно создаст аккаунт в приложении, а эти два пункта не удовлетворяют критерию простоты использования. Именно по этой причине необходимо разработать приложение лишенное описанных ранее недостатков.

# **ЦЕЛИ И ЗАДАЧИ**

## **Цель**

Целью данной курсовой работы является создание мобильного приложения на операционную систему Android для контроля времени и задач.

## **Задачи**

Проанализировав описанные выше аналоги приложений, были поставлены следующие задачи:

1) реализовать приложение, предоставляющее возможность для контроля выполнения задач, не требующее регистрации и аутентификации.

2) разработать простой в использовании графический интерфейс приложения, предоставляющий возможность использования основного функционала программы с минимальным количеством необходимых действий для его управления.

## **Необходимые средства**

Для разработки программного средства использован язык программирования Java. Среда разработки – Android Studio. Для обеспечения работоспособности всех функций приложения на устройствах с разной версией операционной системы Android был использован класс AppCompatActivity. Для обеспечения удобства использования были использованы Fragments.

# **ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА**

## **Структура приложения**

Архитектура приложения представлена в виде Model-View-Controller [1], что позволяет обособить такие функции как загрузку и представление данных от, непосредственно, отображения, что в свою очередь даёт возможность построения расширяемого приложения, компоненты которого между собой слабо связаны.

Всё программное средство реализовано в рамках одного проекта в Android Studio. Активити-хостом для списка задач является класс SingleFragmentActivity, для активной задачи – TaskPagerActivity. В целях осуществления большей гибкости пользовательского интерфейса было принято решение использовать технологию Fragment [2] из библиотеки поддержки. Ввиду существования двух способов внедрения фрагментов, было решено внедрять фрагменты в коде активити, а не макета (Листинг 3.1), что предоставляет больший контроль управления над фрагментами.

Листинг 3.1 – объявление фрагмента в коде программы

1. FragmentManager fm = getSupportFragmentManager();
2. Fragment fragmen = fm.findFragmentById(R.id.fragment\_container);
3. if(fragment == null){
4. fragment = createFragment();
5. fm.beginTransaction()
6. add(R.id.fragment\_container,fragment)
7. .commit();
8. }

Пакет с классами TaskBaseHelper, TaskCursorWrapper, TaskDbSchema предназначены для доступа к реляционной базе данных SQLite. Реляционная база данных SQLite [3] была выбрана в качестве места хранения данных на устройстве, так как в отличие от других баз данных она хранит данные в простых файлах, для чтения и записи которых была использована библиотека SQLite, а она в свою очередь входит в стандартную библиотеку Android.

## **Model**

В рамках проекта реализованы следующие классы уровня DAL: Task, TaskLab.

Модель Task представляет собой одно задание. Эта модель содержит primary key – uuid. Задание содержит заголовок, состоящий из текста, представленный полем String mTitle. Каждому заданию можно устанавливать дату, которая представлена полем Date mDate. Задание может содержать какое-либо описание, раскрывающее или поясняющее смысл этого задания. Описание представлено полем String mDescription. Должна быть возможность отслеживания выполнения поставленных задач. Это достигается путем добавления boolean поля mSolved.

Модель TaskLab представляет собой централизованное хранилище для объектов Task. В этом классе осуществляется доступ к базе данных с возможностью удаления и добавления новых записей.

## **View**

Для построения представления списка задач, было решено использовать технологию RecyclerView [4]. Этот выбор обусловлен тем, что, во-первых, не придётся управлять представлением всех добавленных задач самостоятельно, а во-вторых, представление View необходимо объекту Task только во время его нахождения на экране, и из-за этого держать на наготове все добавленные объекты нет необходимости (рисунок 3.1)

View активной задачи представляет собой подробную детализацию объекта. Она включает в себя UI для ввода информации по новой добавленной задаче, редактирования и удаления уже существующей задачи (рисунок 3.2). Также присутствует диалоговое окно для выбора даты (рисунок 3.3).

В целях увеличения удобства пользования был разработан двух панельный интерфейс для планшетов. На котором 1/3 экрана занимает отображение списка со всеми задачами, а остальное пространство детализация выбранной задачи (Рисунок 3.4).

|  |  |  |
| --- | --- | --- |
|  |  |  |
| Рисунок 3.1 – главная страница приложения | Рисунок 3.2 – детализация задания | Рисунок 3.3 – диалоговое окно для выбора даты |
|  | | |
| Рисунок 3.4 – двух панельный интерфейс для планшетов | | |

## **Controller**

В качестве контроллера выступают классы TaskListFragment и TaskListActivity.

TaskListaActivity минималистичен и содержит один метод, который запускает активити, выбранную пользователем из списка.

TaskListFragment выступает в роли “менеджера” всех задач. Он подготавливает все записи из базы данных к их представлению на View и отправляет запрос к TaskListaActivity, чтобы тот запустил активити с нужной задачей (рисунок 3.4).

|  |
| --- |
|  |
| Рисунок 3.4 – схема взаимодействия слоёв |

# **ТЕСТИРОВАНИЕ**

Приложение было протестировано с использованием двух физических устройств и двух эмуляторов [5]: Pixel 3 (API 29), Pixel C (API 29). Приложение показало ожидаемый результат: можно создавать список задач, который сохраняется локально на устройстве. Все функции приложения работают, как и ожидалось: жесты и разделение экрана на две части на большом экране. Приложение закрывается и работает стабильно без “вылетов” и ошибок.

# **РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ**

## **Необходимые требования**

Для успешной работы приложения необходимо иметь версию Android 4.4 и выше.

## **Начало работы**

Первым действием при эксплуатации разработанного приложения для управления задачами необходимо установить его на физическое устройство через устройство, на котором установлена Android Studio, или с помощью .apk файла.

По окончании установки, пользователь может перейти непосредственно к функционалу, разработанного программного средства.

После установки приложения будет видено пустое пространство (рисунок 5.1).

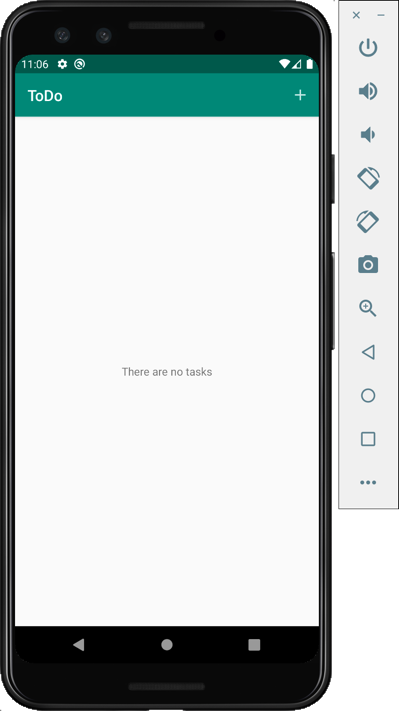


Рисунок 5.1 – пустое поле

После этого надо нажать на “плюс” в правом верхнем углу экрана для того, чтобы добавить новую задачу. После этого откроется Activity с детализацией самой задачи. После заполнения нужных полей, пользователь может выйти к основному списку двумя способами: нажатием кнопки “назад” или с помощью стрелочки на панели действий. На Activity с детализацией есть кнопка для удаления задачи из базы данных. Также удалить задачу можно при помощи свайпа на главной Activity.

Все данные приложения остаются на устройстве до тех пор, пока пользователь самостоятельно не удалит запись или само приложение с устройства.

# **ЗАКЛЮЧЕНИЕ**

Параллельно с разработкой описанного в данном проекте программного средства разрабатывались и другие приложения, во время чего многократно возникала необходимость быстрого отслеживания сделанных работ, что невозможно было невозможно сделать при помощи сервисов, требующих регистрацию и действий, отнимающих время и концентрацию. Этот факт является еще одним доказательством применимости и необходимости разрабатываемого программного средства.

В результате выполнения данной курсовой работы разработано приложение для отслеживания дел. Оно отличается простотой использования, минималистичным и приятным интерфейсом, а также, что не мало важно, быстротой работы, в плане необходимого количества взаимодействий пользователя с графическим интерфейсом для использования основного функционала.

Цели, которые ставились перед разработкой данного приложения, считаются успешно достигнутыми. Кроме одного пункта: при разработке данного курсового проекта планировалось, что будет присутствовать возможность добавления подзадач к основной задаче, однако, по итогу, она не была реализована.

В ходе разработки были изучены возможности IDE Android Studio. Получены навыки в разработке Android-приложений.

В силу выбора объектно-ориентированной парадигмы программирования приложение обладает свойством расширяемости, что сэкономит большое количество ресурсов и времени для возможных будущих разработок и улучшений существующего программного кода.

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Model-View-Controller – Википедия // Википедия - свободная энциклопедия [Электронный ресурс]. – 2017. - Режим доступа: <https://ru.wikipedia.org/wiki/Model-View-Controller>. - Дата доступа: 05.10.2020

Fragments | Android Developers // Android Developers [Electronic resource]. – 2019. - Mode of access: https://developer.android.com/guide/components/fragments. – Data of access: 05.16.2020

SQLiteDatabase | Android Developers // Android Developers [Electronic resource]. -2020. – Mode of access: <https://developer.android.com/reference/android/database/sqlite/SQLiteDatabase>. - Data of access: 05.16.2020

1. Create a List with Recyclerview | Android Developers // Android Developers [Electronic resource]. – 2020. - Mode of access: <https://developer.android.com/guide/topics/ui/layout/recyclerview>. – Date of access: 05.16.2020
2. Run apps on the Android Emulator | Android Developers // Android Developers [Electronic resource]. – 2020. – Mode of access: <https://developer.android.com/studio/run/emulator>. – Date of access: 05.16.2020
3. Start Android - Учебник по Android для начинающих и продвинутых. – 2020. - Режим доступа: <https://startandroid.ru/ru/>. - Дата доступа: 05.16.2020