Муниципальное автономное образовательное учреждение

«Лицей №38»

Секция «Прикладная информатика»

**Создание приложения для мобильных устройств с ОС Android «Читательский дневник»**

Выполнил: Дубенков Родион,

учащийся 11 класса 11 группы

МАОУ «Лицей №38»

Руководитель: Дудина О.В.,

г. Нижний Новгород

2021 г.

**Содержание:**

Введение…………………………………………………………………………1

**Глава I.** Теоретические основы создания приложения «Читательский дневник»………………… …………...………………………………………...…8

1. Основные понятия…………………………………………………………..
2. О операционной системе Android ………………………………………….
3. Общая схема работы приложения Android……………………………………………………………………….

**Глава II.** Выбор средств реализации поставленных задач и избрание концепции разрабатываемого программного обеспечения

* 1. Подбор инструментов……………………...
  2. Обзор аналогичных продуктов
  3. Описание концепции разрабатываемого программного обеспечения
  4. Требования к приложению

Глава **III.** Программная реализация

1. Архитектура ……………………………
   1. Объяснение применению UI – фрагментов
   2. Архитектура MVC

2. Список модулей…..

3. Описание функциональности…

4. Тестирование……………………………………

5. Возможные варианты развития программного приложения…………

Заключение………………………………………………………………………

Список использованной литературы…………………………………………...

Приложения……………………………………………………………………...13

**Введение**

С каждым годом телефон приобретает новые функции, регулярно обновляется, совершенствуется и становится все мощнее и мощнее. На данный момент можно найти немало мобильных устройств, которые могут конкурировать в производительности с компьютерами и ноутбуками нового поколения. Тем более телефон удобный и практичный в повседневном использовании. Таким образом, приложения под мобильные устройства с каждым годом становятся востребование.

В наше время, самая распространенная оперативная система ОС Android. Android поддерживает большое количество устройств, от разных производителей. Главная причина распространения OCAndroid–бесплатные средства разработки, в то время как разработка под систему IOS требует высоких начальных затрат.

Исходя из выше перечисленного, я решил создать приложение “Читательский дневник”для мобильных устройств с ОС Android. Разработанный проект облегчит людям ведение читательского дневника. В отличие от бумажной его версии, приложение сможет быстро находить нужный элемент, показывать статистику прочитанного, а также перестановка, удаление, репорт, все необходимые характеристики произведения, удобный интерфейс - все это будет доступно в приложении. И, конечно, телефон удобнее в ведении читательского дневника, потому что он всегда под рукой. Таким образом, приложение поможет вести отсчет о прочитанном не только учащимся, но и взрослым.

**Целью** моей работы является создание приложения для мобильных устройств с ОС Android, которое поможет следить за прогрессом в чтении произведений и облегчит человеку ведение читательского дневника.

Для решения поставленной цели нужно решить следующие **задачи**:

1. Познакомиться с операционной системой Android, узнать общую схему работы приложения на ОС Android.
2. Подобрать инструменты для работы над приложением и описать концепцию разрабатываемого программного обеспечения.
3. Реализовать приложение, которое сможет взаимодействовать с элементами списка, а именно сохранять, сортировать, удалять, перемещать, изменять и показывать общую статистику по всем элементам.

**Глава I. Теоретические основы создания приложения «Читательский дневник»**

* 1. **Основные понятия**

***Android*** — операционная система для смартфонов, интернет-планшетов, электронных книг, цифровых проигрывателей, наручных часов, игровых нетбуков, смартбуков, очков Google, телевизоров и других устройств.

***XML*** (расширяемый язык разметки) — это язык программирования, который состоит из объявлений в виде информации и определяющих тегов. С его помощью удобно хранить и передавать любые данные.  
Язык не зависит от операционной системы и среды обработки. XML служит для представления неких данных в виде структуры, которую вы можете сами разработать или подстроить под программу или сервис. Именно поэтому данный язык называют расширяемым, и в этом его главное достоинство, за которое его так ценят.

***Java***—представляет собой язык программирования и платформу вычислений, которая была впервые выпущена SunMicrosystems в 1995 г.

***Java Development Kit*** (сокращенно JDK) — бесплатно распространяемый компанией OracleCorporation (ранее SunMicrosystems) комплект разработчика приложений на языке Java, включающий в себя компилятор Java (javac), стандартные библиотеки классов Java, примеры, документацию, различные утилиты и исполнительную систему Java (JRE).

***Android Studio*** — это интегрированная среда разработки (IDE) для работы с платформой Android, анонсированная 16 мая 2013 года на конференции GoogleI/O. Инструментарий программной разработки Android SDK находится в свободном доступе и включает в себя интерфейсы прикладного программирования (API) на языке Java.

***API*** (интерфейс программирования приложений, интерфейс прикладного программирования) (англ. *application programming interface*) — набор готовых классов, процедур, функций, структур и констант, предоставляемых приложением (библиотекой, сервисом) или операционной системой для использования во внешних программных продуктах. Используется программистами при написании всевозможных приложений.

***База данных*** — представленная в объективной форме совокупность самостоятельных материалов (статей, расчѐтов, нормативных актов, судебных решений и иных подобных материалов), систематизированных таким образом, чтобы эти материалы могли быть найдены и обработаны с помощью электронной вычислительной машины (ЭВМ).

***Система управления базами данных*** (СУБД) — совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных.

SQLite - компактная встраиваемая СУБД. Исходный код библиотеки передан в общественное состояние.

Google Play — магазин приложений, а также игр, книг, музыки и фильмов от компании Google, позволяющий сторонним компаниям предлагать владельцам устройств с операционной системой Android устанавливать и приобретать различные приложения.

**1.2 О операционной системе Android**

Android - операционная система для смартфонов, планшетных компьютеров, электронных книг, цифровых проигрывателей, "умных" наручных часов, игровых приставок, нетбуков, смартбуков, очков Google, телевизоров, ситем автоматического управления автомобилем и других устройств. ОС основана на ядре Linux и собственной реализации виртуальной машины Java от Google. Изначально разрабатывалась компанией AndroidInc., которую в 2005 году купила Google. Впоследствии Google инициировала создание альянса OpenHandsetAlliance (OHA), который сейчас занимается поддержкой и дальнейшим развитием платформы. Android позволяет создавать Java-приложения, управляющие устройством через разработанные Google библиотеки. AndroidNativeDevelopmentKit позволяет портировать (но не отлаживать) библиотеки и компоненты приложений, написанные на Си и других языках. ОС Android установлена на 86% смартфонов (2014).

Android является распространенной операционной системой (ОС) для мобильных устройств – телефонов и планшетов. Данная система имеет множество отличительных черт, которые делают ее узнаваемой и привлекательной для большого количества пользователей по всему миру.

Отличительной особенностью Android является его интегрированность с сервисами Google – Gmail, Hangouts, Voice Search и т.п. На Android официально реализована поддержка Chrome, что позволяет синхронизировать открываемые в браузере вкладки на смартфоне с компьютерным браузером.

Например, вы можете начать просмотр страниц с вашего телефона и при желании продолжить изучать информацию, открыв эту же вкладку на компьютере, не прибегая к помощи повторного поиска.

«Андроид» имеет достаточно простой и интуитивно понятный интерфейс. Все нужные приложения размещаются одновременно на главном 8

экране и в меню аппарата, которое вызывается нажатием на центральную сенсорную клавишу или соответствующую кнопку на экране. Все настройки располагаются в секции «Настройки», а каждое действие пользователя поясняется комментариями и подсказками при первом запуске аппарата. Операционная система быстро реагирует на нажатия пользователя и производит установку и скачивание нужных программ и файлов со скоростью, которая не проигрывает другим современным мобильным ОС.

Платформа Android не различает по правам предустановленные и сторонние приложения, что позволяет менять программную конфигурацию устройства, в том числе на самостоятельно разработанные. Основным языком разработки под Android является Java, однако, можно так же использовать другие языки программирования, например, С++. Все функциональные возможности системы открыты, так же имеется возможность создания новых компонент.

Система приложений включает различные службы, курирующие работу составляющих системы (Activity Manager, Resource Manager и другие).

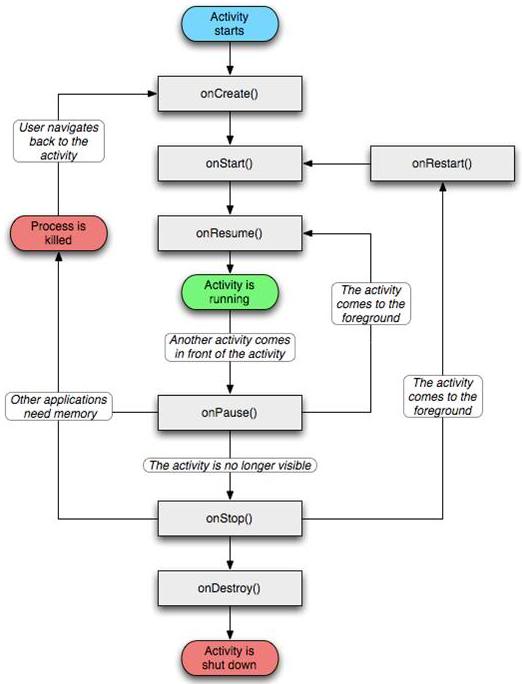
Activity Manager — диспетчер активности, который отвечает за функционирование приложения и его жизненный цикл.

Resource Manager — диспетчер ресурсов необходим для доступа к используемым внутренним ресурсам (строковым, графическим и т.п.).

* 1. **Общая схема работы приложения Android**

Приложения для Android в своей работе использует окна, однако в данной системе вышеуказанные окна носят иное название – Activity. Активность – это экран (по аналогии с web-формой), который приложение может показывать пользователям. Чем сложнее создаваемое приложение, тем больше экранов (Активностей) потребуется. При создании приложения потребуется, как минимум, начальный (главный) экран, который обеспечивает основу пользовательского интерфейса приложения.

При создании нового окна вызывается метод onCreate(), при разработке данный метод переопределяется и в нем происходит инициализация приложения и его компонентов. Далее вызываются методы onStart() и onResume(). Оба метода вызываются перед отображением окна при его создании, либо восстановлении(при переключении из другого приложения, при разворачивании свернутого приложения и тп). При сворачивании вызываются методы onPause() и onStop(). При закрытии приложения и окна вызывается onDestory(), в данном методе можно сохранить пользовательские данные и параметры.



*Рисунок 1. Состояния активностей*

**Глава II. Выбор средств реализации поставленных задач и избрание концепции разрабатываемого программного обеспечения**

**2.1 Подбор инструментов**

Существует достаточно много программ для создания приложений на ОС Android. Например, Eclipse, Embarcadero JBuilder, JDeveloper и т.д, была выбрана среда программирования – Android Studio. Причины выбора: удобный интерфейс и быстродействие программы.

В программе Android Studio на данный момент есть выбор среди трёх языков программирования. Это java, С++ и Kotling. В данном случае выбор языка не повлияет на результат. Приложение можно написать на разных языках. Был выбрал - java, потому что его я знаю лучше всех остальных.

Приложение “Читательский дневник” в любом случае потребует базы данных, потому что записи нужно будет сохранять на долгое время. Есть такие варианты СУБД: SQLite, MySQL, Oracle. Выбор пал на SQLite

Преимущества СУБД SQLite:

* Файловая структура - вся база данных состоит из одного файла, поэтому её очень легко переносить на разные машины
* Используемые стандарты - хотя может показаться, что эта СУБД примитивная, но она использует SQL. Некоторые особенности опущенны (RIGHT OUTER JOIN или FOR EACH STATEMENT), но основные все-таки поддерживаются
* Отличная при разработке и тестировании - в процессе разработки приложений часто появляется необходимость масштабирования. SQLite предлагает всё что необходимо для этих целей, так как состоит всего из одного файла и библиотеки написанной на языке C.

Cтоит заметить, что благодаря популярности этой СУБД, существует огромное количество различных плагинов и расширений, облегчающих работу с системой. СУБД SQLite достаточно примитивная, но для однопользовательского мобильного приложения она в самый раз. Тем более, что масштабируемость в данном приложении не нужна. А также создание таблицы SQLite происходит проще, чем в других базах данных: вам не нужно создавать тип столбца в момент создания.

**2.2 Обзор аналогичных продуктов**

На просторах GooglePlay на данный момент очень мало приложений с функциями читательского дневника. Тем не менее самое крупное приложение называется BookLover. Рассмотрим функции предоставляемые в приложении BookLover, а также плюсы и минусы этого приложения.

Функции: 1) сортировка элементов

2) отображение статистики

3) отображение информации о произведении

4) публикация произведения

Плюсы: 1) качественная сортировка элементов

2) удобное главное меню

Минусы: 1) неполная статистика

2) нет возможности выставить дату окончания чтения (всегда принимается за день добавления книги)

3) нет поля для краткого содержания или анализа произведения

4) неудобный просмотр записи о книге после ее добавления

5) загруженность приложения рекламой

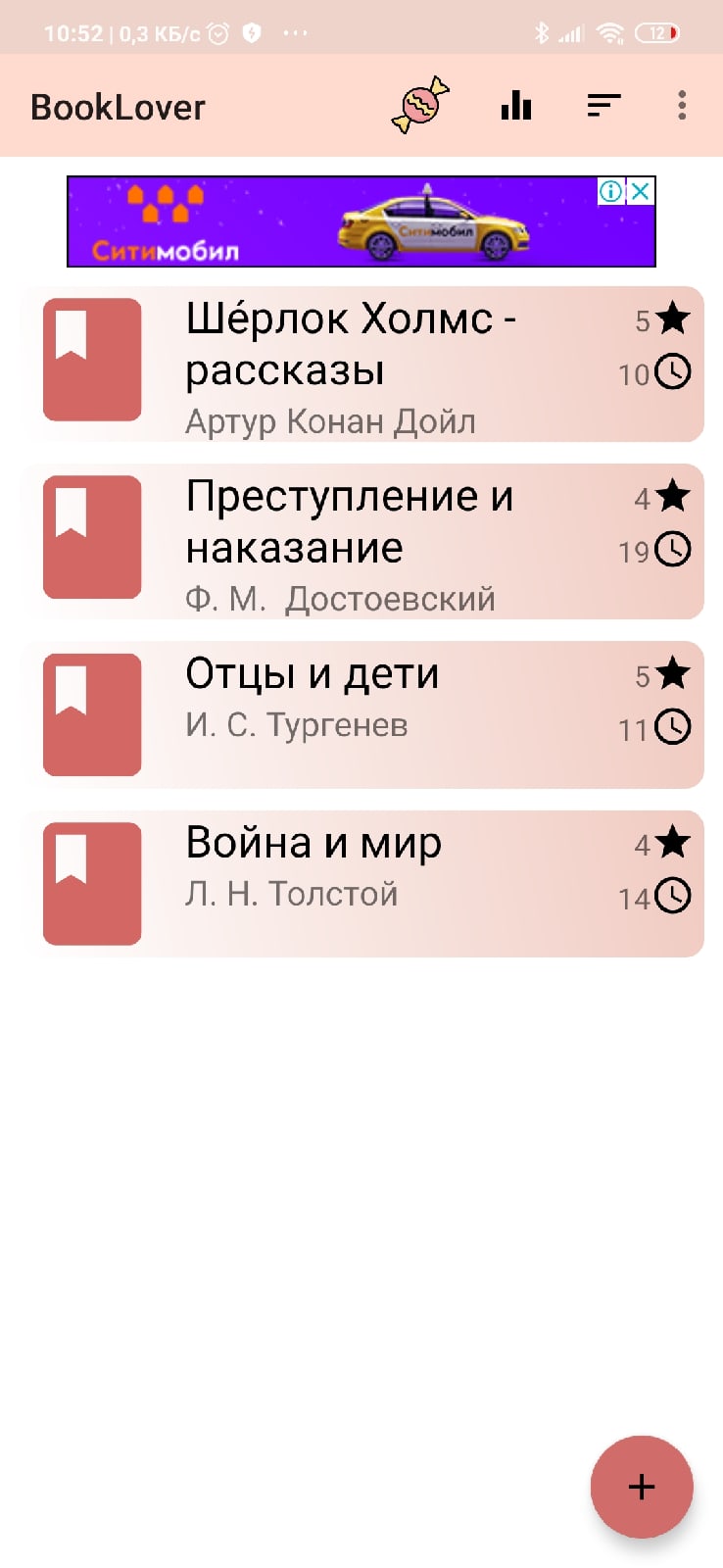
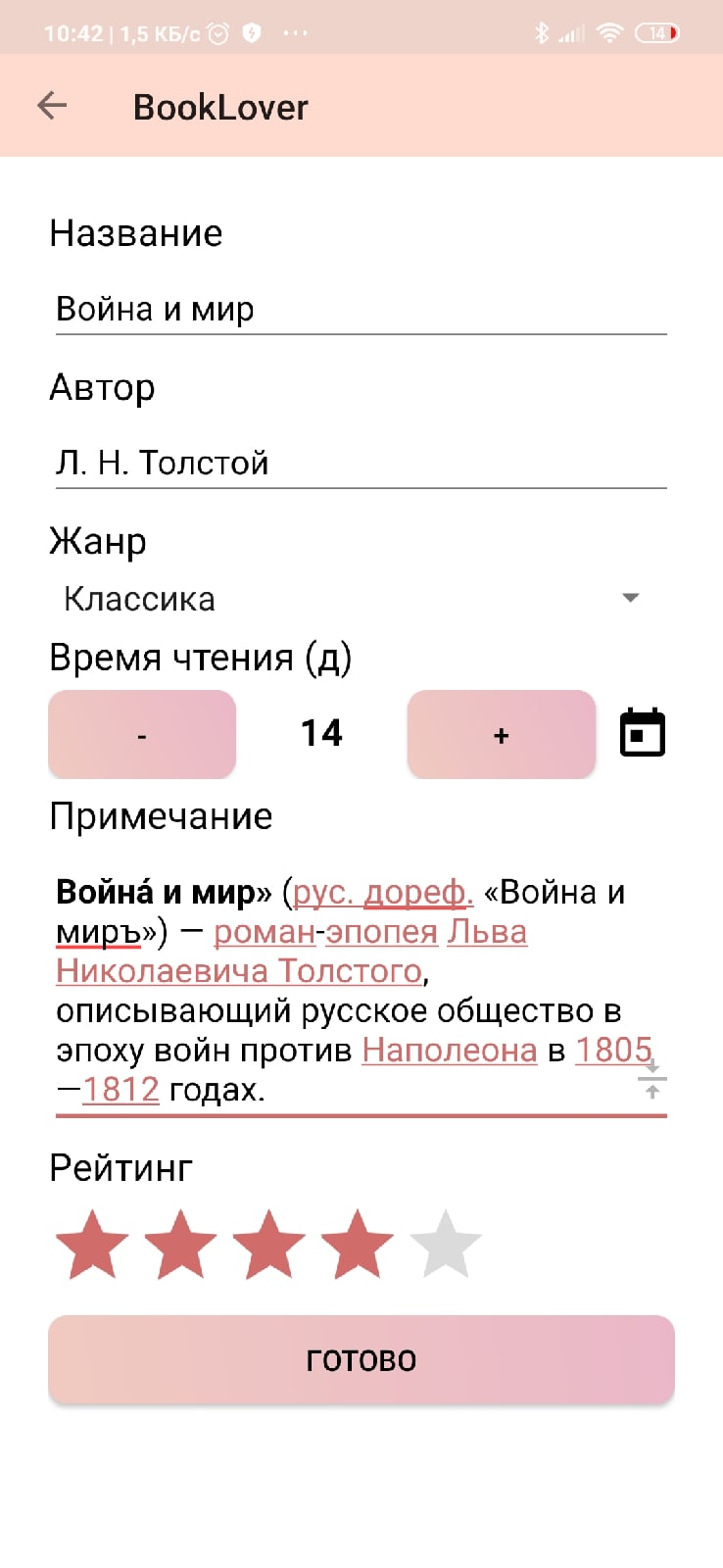
 

Рисунок 2 – Главное меню Рисунок 3 – Страница создания записи

приложения BookLover

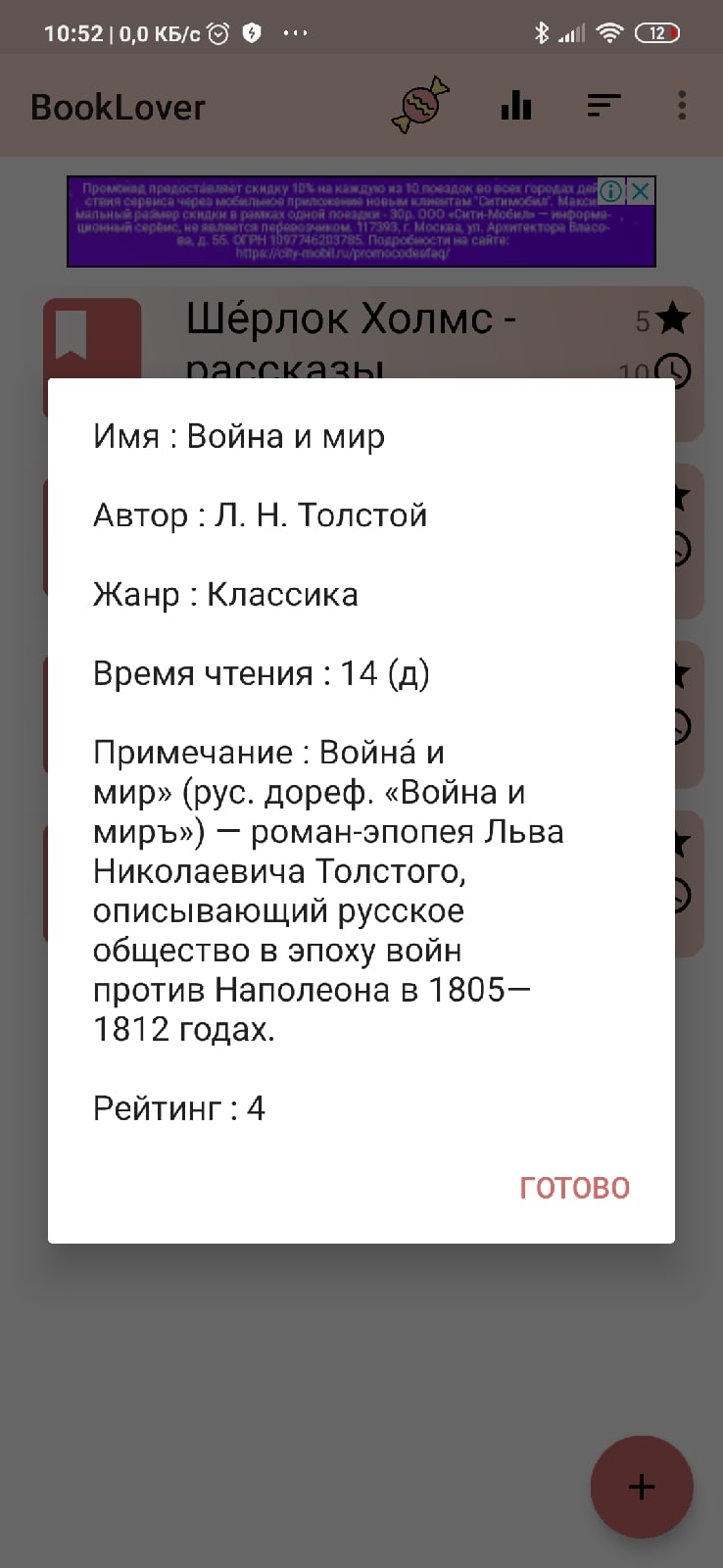
 

Рисунок 4 - записи о книге Рисунок 5 – просмотр

после ее добавления статистики

Таким образом, приложение BookLover имеет достаточно много минусов. Поэтому есть необходимость создания нового приложения – читательского дневника, которое исправит все минусы приложения BookLover и добавит новые необходимые детали.

* 1. **Описание концепции разрабатываемого программного обеспечения**

Концепция разрабатываемого программного продукта представляет собой приложение для OCAndroid, предоставляющее следующие самые главные возможности:

1. Создание записи о произведении
2. Сохранение этой записи
3. Поиск записи с помощью сортировки

Все записи будут отображаться на главном меню в виде списка, который можно скролить. Все записи будут сохранены в базу данных SQLite. Также есть возможность охватить большую аудиторию. Чтобы этим приложением могли пользоваться абсолютно все (знающие английский или русский язык), я локализую приложение “Читательский дневник” под английский язык (английский – это международный универсальный язык).

Очень важным моментом для всех приложений и игр является совместимость. Важно, чтобы приложение не только могло поддерживаться различными устройствами, но и быть удобным для каждого из них. Для того чтобы приложение максимально удобно использовалось как для телефонов, так и для планшетов, будет создана специальная ориентация только для телефонов и специальная ориентация только для планшетов.

Теперь выделим основные функции приложение “Читательский дневник” под ОС Android. В список основных функций моего приложения входят:

1. Создание и сохранение записи о произведении
2. Сортировка элементов
3. Перемещение и удаление элементов списка
4. Отображение статистики
5. Публикация произведения

Таким образом, приложение составит прямую конкуренцию для BookLover, имея при себе ряд преимуществ.

Самое главное преимущество в удобном расположении элементов и удобном перемещении пользователя по приложению. Созданное мною приложение исправит все минусы BookLover и добавит новые детали.

* 1. **Требования к приложению**

Требования целесообразно разделить на несколько групп.

*Функциональные характеристики приложения:*

1) корректное отображение всего содержимого на экране (объектов и текста), чтобы у пользователя не возникали трудности и раздражение при чтение текста или просмотра картинки;

2) корректный ввод команд и вывод данных, чтобы пользователь мог вводить данные и это не вызывало у него никаких затруднений, также важно чтобы информация выдаваемая приложением была правильной;

3) все данные “доставаемые”из базы данных должны быть правильные, неискажённые.

4) работа приложения должна быть стабильной. Взаимодействие с любыми его элементами и компонентами не должно приводить к резкому завершению процесса;

5) изначально при попадании в приложение сразу переходить в главное меню

6) приложение не должно зависеть от наличия интернет-соединения и работать без интернет-подключения;

7) сортировка элементов должна быть в соответствии с выбранным типом.

8) статистика должна быть правильная, неискаженная

9) все введенные данные о записи и расстановка элементов должны сохраняться в БД

*Требования для комфортного использования.*

1) программа правильно и корректно работает на разных входных данных

2) временные задержки не больше 0,1 с. (чтобы пользователю было комфортно работать);

3) программа должна сохранять работоспособность на протяжении всего сеанса пользователя вне зависимости от его длительности;

*Параметры технических средств:*

1) платформа ОС Android с версией выше 4.0.0;

2) техническое средство должно, как минимум, иметь одноядерный процессор с частотой выше 1Ггц, оперативную память ёмкостью в 512 Мб, сенсорный экран (не менее 640х480);

Итак, для того, чтобы мобильное приложение могло быть использовано, оно должно соответствовать определенным требованиям: функциональные характеристики, требования для комфортного использования, определенные технические параметры. Выполнение всех вышеуказанных требований позволит создать такое мобильное приложение, которое будет удобно для использования.

***Программная реализация***

***1. Архитектура***

* 1. ***Объяснение применению UI – фрагментов***

Приложение для ОС Android состоит из набора активностей, каждой из которых соответствует экран приложения. Каждая активность представлена в проекте классом, реализованном на языке Java, хранящемся в одноименном файле с расширением .java. Каждой активности соответствует xml файл-описание. В xml-файле описано в виде xml кода расположение визуализируемых объектов. При запуске активности система Android автоматически распознает размер экрана мобильного устройства и приводит выводимый контент в соответствие с разметкой, описанной в xml файле. Таким образом, одна и та же активность будет выглядеть одинаково независимо от диагонали используемого устройства. Также, для каждого приложения Android должен существовать xml файл, в котором в виде xml кода будут прописаны минимальные требования к системе, а также активность, вызываемая при запуске приложения.

В приложении “Читательский дневник” используется интерфейс типа “список / детализация”: на главном экране выводится список зарегистрированных произведений. Пользователь может добавить новое или выбрать существующее произведение для просмотра и редактирования информации.

Разумно предположить, что приложение типа “список / детализация” состоит из двух активностей: для управления списком и для управления детализированным представлением. Щелчок на произведении в списке запускает экземпляр детализированной активности. Нажатие кнопки BACK уничтожает активность детализации и возвращает на экран список, в котором можно выбрать другое произведение.

Такая архитектура работает, но что делать, если вам потребуется более сложная схема представления информации и навигации между экранами?

- Допустим, пользователь запустил приложение “Читательский дневник” на планшете. Экраны планшетов и некоторых больших телефонов достаточно велики для одновременного отображения списка и детализации, по крайней мере в альбомной ориентации.

-Пользователь просматривает описание произведения на телефоне и хочет увидеть следующее произведение в списке. Было бы удобно, если бы пользователь мог провести пальцем по экрану, чтобы перейти к следующему произведению без возвращения к списку. Каждый жест прокрутки должен обновлять детализированное представление информацией о следующем произведении.

Эти ситуации объединяет гибкость пользовательского интерфейса: возможность формирования и изменения представления активности во время выполнения в зависимости от того, что нужно пользователю или устройству.

Подобная гибкость в активности не предусмотрена. Представления активности могут изменяться во время выполнения, но код, управляющий этими изменениями, должен находиться в представлении. В результате активность тесно связывается с конкретным экраном, с которым работает пользователь.

Закон Android нельзя нарушить, но можно обойти, передав управление пользовательским интерфейсом приложения от активности одному или нескольким фрагментам.

Фрагмент представляет собой объект контроллера, которому активность может доверить выполнение операций. Чаще всего такой операцией является управление пользовательским интерфейсом – целым экраном или его частью.

Фрагмент, управляющий пользовательским интерфейсом, называется UI – фрагментом. UI – фрагмент имеет собственное представление, которое заполняется на основании файла макета. Представление фрагмента содержит элементы пользовательского интерфейса, с которыми будет взаимодействовать пользователь.

Представление активности содержит место, в которое вставляется представление фрагмента. Более того, хотя в нашем примере активность управляет одним фрагментом, она может содержать несколько мест для представлений нескольких фрагментов.

Фрагменты, связанные с активностью, могут использоваться для формирования и изменения экрана в соответствии с потребностями приложения и пользователей. Представление активности формально остается неизменным на протяжении жизненного цикла, и законы Android не нарушаются.

Давайте посмотрим, как эта схема работает в приложении “список / детализация”. Представление активности строится из фрагмента списка и фрагмента детализации. Представление детализации содержит подробную информацию о выбранном элементе списка.

При выборе другого элемента на экране появляется новое детализированное представление. С фрагментами эта задача решается легко; активность заменяет фрагмент детализации другим фрагментом детализации. Это существенное изменение представления проходит без уничтожения активности.

Применение UI – фрагментов позволяет разделить пользовательских интерфейс вашего приложения на структурные блоки, а это полезно не только для приложений “список / детализация”. Работа с отдельными блоками упрощает построение интерфейсов с вкладками, анимированных боковых панелей и много другого.

* 1. **Архитектура MVC**

**Model-View-Controller** (**MVC**, «Модель-Представление-Контроллер», «Модель-Вид-Контроллер») — схема разделения данных приложения, пользовательского интерфейса и управляющей логики на три отдельных компонента: модель, представление и контроллер — таким образом, что модификация каждого компонента может осуществляться независимо.

Исходя из функций нашего приложения, архитектура MVC идеально подходит, потому что такая архитектура наиболее удобна для небольших, простых приложений. В нее легко добавляются новые функции, и она позволяет легко представить взаимодействие между взаимодействующими частями приложения. MVC закладывает надежную основу для разработки, позволяет быстро построить работоспособное приложение и эффективно работает на ранних стадиях работы над проектом. А такие архитектуры как MVVM используются при очень больших размерах программы.

Всего для успешной программной реализации приложения “Читательский дневник” потребовалось 15 классов (можно было сделать и меньше, но зато код легко читается, а это очень важно). Теперь подробнее, уровень модели представлен двумя классами BookLab и Book. BookLab предоставляет доступ ко всему списку элементов, позволяет удалить элемент, добавить элемент, а также добавить информацию об элементе в базу данных, найти элемент и обновить список элементов. Если кратко, то управляет данными. Класс Book хранит данные об элементе, с которым взаимодействует пользователь.

База данных представлена тремя классами: BookBaseHelper, BookCursorWrapper, BookDbSchema. BookDbSchema содержит константы – названия столбцов таблицы. BookBaseHelper создает базу данных. BookCursorWrapper “вынимает” данные о элементе из базы данных SQLite и отправляет в класс Book.

Объекты контроллеров – это классы BookListActivity, BookListFragment, BookFragment, BookPagerActivity, DatePickerFragment, SortFragment, StatisticsFragment, SimpleItemTouchHelperCallback. Все эти классы связывают объекты модели и объекты представления. Интерфейсом страницы со списком будет управлять UI – фрагмент с именем BookListFragment. Макет UI - фрагмента представляет из себя список элементов и несколько кнопок на панели инструментов. Здесь был использован список RecyclerView. RecyclerView гораздо **мощнее, гибче и значительно лучше по сравнению ,например,с ListView**. Главные преимущества: наличие шаблона ViewHolder, LayoutManager, качественных и простых для создания анимаций. BookListFragment будет обрабатывать нажатие кнопок, выбор элемента списка и выполнять операции в зависимости от действий пользователя. Хостом (host) экземпляра BookListFragment является активность с именем BookListActivity. По сути BookListActivity только запускает UI - фрагмент.

Интерфейсом страницы с детализацией элемента будет управлять UI – фрагмент с именем BookFragment. То есть BookListFragment будет обрабатывать нажатие кнопок, ввода текста и выполнять операции в зависимости от действий пользователя (например, отправлять данные на уровень модели). Хостом (host) экземпляра BookFragment является активность с именем BookPagerActivity. Макет активности будет состоять из экземпляра ViewPager. Включение виджета ViewPager в пользовательский интерфейс позволяет “листать” элементы списка, проводя пальцем по экрану. Эта очень удобная функция.

DatePickerFragment, SortFragment, StatisticsFragment это UI – фрагменты, макет которых представляет из себя диалоговые окна. DatePickerFragment – это класс, который управляет диалоговым окном, которое позволяет выбрать дату (вызывается он из UI – фрагмента BookFragment для выбора даты начала и окончания чтения). SortFragment – это класс, который управляет диалоговым окном, которое позволяет отсортировать элементы списка по определенным параметрам (вызывается в он из UI – фрагмента BookListFragment для выбора категории, по которой производится сортировка). StatisticsFragment - это класс, который управляет диалоговым окном, которое позволяет увидеть пользователю статистику списка элементов (вызывается он из UI – фрагмента BookListFragment).

Класс SimpleItemTouchHelperCallback позволяет внедрить в приложение функциональность удаления смахиванием и перестановки (при перетаскивании элемента), которая сделает работу с RecyclerView в приложении более удобной.

Также существует в структуре приложения абстрактный класс SingleFragmentActivity, который является суперклассом для класса BookListActivity. SingleFragmentActivity нужен для увеличения гибкости приложения. В структуре приложения также есть интерфейсы, например, ItemTouchHelperAdapter и Callbacks. Интерфейс Callbacks важен для реализации макета в различных видах в зависимости от диагонали экрана телефона или планшета пользователя. Таким образом, макет будет выглядеть на больших устройствах в виде двухпанельного интерфейса. Чтобы делегировать функциональность активности – хосту, фрагмент обычно определяет интерфейс обратного вызова с именем Callbacks. Этот интерфейс определяет работу, которая должна быть выполнена для фрагмента его “начальником” – активностью – хостом. Любая активность, выполняющая функции хоста фрагментов, должна реализовать этот интерфейс.

Объекты представления – это макеты UI –фрагментов и активностей, представленных в объекте контроллера, а именно activity\_book\_pager, activity\_fragment, activity\_twopane, dialog\_date, dialog\_sort, dialog\_statistics, fragment\_book, fragment\_book\_list, list\_item\_book. Каждый макет представлен XML – файлом и содержит виджеты для диалога с пользователем. А виджеты передают как раз информацию контроллеру, а он ей управляет. Более подробно о уровне представления рассказывается в следующем подразделе под названием “Список модулей”.

1. Список модулей

В папке проекта находится папка ресурсов “res”, где собраны все вспомогательные файлы для работы с проектом. Все они являются файлами XML. Такое формат создан для того, чтобы описывать объекты и частично описывать поведение программ, читающих XML –документы.

В папке “layout”, которая находится в папке ресурсов, есть файлы, отвечающие за представление приложения. Это макеты активностей, фрагментов, диалоговых окон и т. д. Они предназначены для описания “внешнего облика приложения”. В приложении очень важно, чтобы пользователю было удобно обмениваться информацией с приложением, поэтому основная задача: обеспечить комфорт в диалоге с приложением. Макеты были представлены нами ранее в подразделе “Архитектура”. Это activity\_book\_pager, activity\_fragment, activity\_twopane, dialog\_date, dialog\_sort, dialog\_statistics, fragment\_book, fragment\_book\_list, list\_item\_book.

Activity\_book\_pager, activity\_fragment, activity\_twopane являются макетами хоста для UI – фрагментов. Фрагмент не может вывести свое представление на экран сам по себе. Ему нужно место, которое как раз обеспечивает хост. То есть эти три представленных макета должны быть пустыми.

Dialog\_date, dialog\_sort, dialog\_statistics – внешний вид диалоговых окон. В каждом из них заложены нужные для данного окна виджеты. Так у dialog\_date находится виджет DatePicker, который нужен для просмотра или установки даты. У dialog\_sort находятся переключатели (кнопки выбора), которые нужны для выбора типа сортировки. У dialog\_statistics расположены вертикально поля, в которых будет заполняться текст – статистика о списке элементов.

Fragment\_book, fragment\_book\_list – макеты, которые связаны на прямую с контроллером. Они являются самыми важными макетами среди всех. Вводя данные в эти макеты и взаимодействуя с ними, пользователь передает информацию к контроллеру, а дальше она обрабатывается. Fragment\_book\_list содержит список RecycleView (рисунок № 6). Наибольшее количество виджетов находится в макете Fragment\_book. В нем находятся строки, поля ввода текста, кнопки обычные, кнопки выбора, выпадающий список и виджет, содержащий рейтинг (рисунок № 7) В макет фрагмента пользователь может ввести следующие параметры о книге: название и автор произведения, состояние (статус), в котором находится книга(прочитана вами, все еще читается или будете читать), дата начала и окончания чтения, количество страниц, жанр, краткое содержание, примечание (или отзыв), рейтинг произведения. Также функционал предоставляет возможность с помощью кнопок, которые находится внизу, удалить книгу или поделиться её характеристикой и своим отзывом о ней со знакомыми.

List\_item\_book – это макет элемента списка. То есть в ней мы задаем внешний вид элемента списка. Здесь очень важно расставить нужную информацию в нужных местах, чтобы пользователю было максимально удобно. Так, самым крупным шрифтом будет написано название произведения, под ним автор, справа статус.

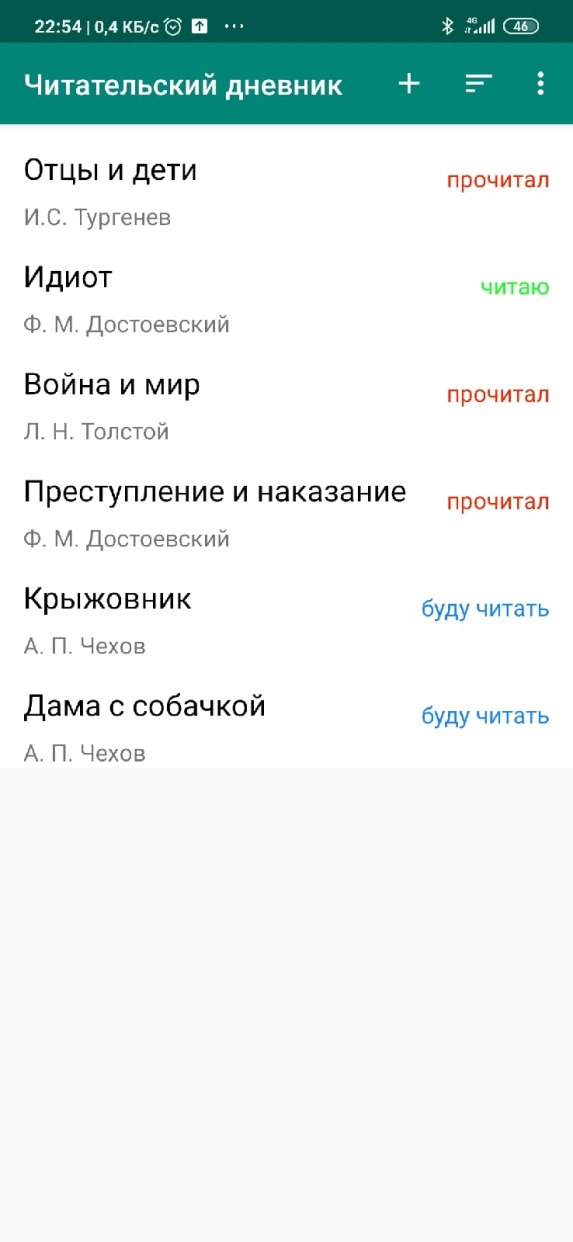
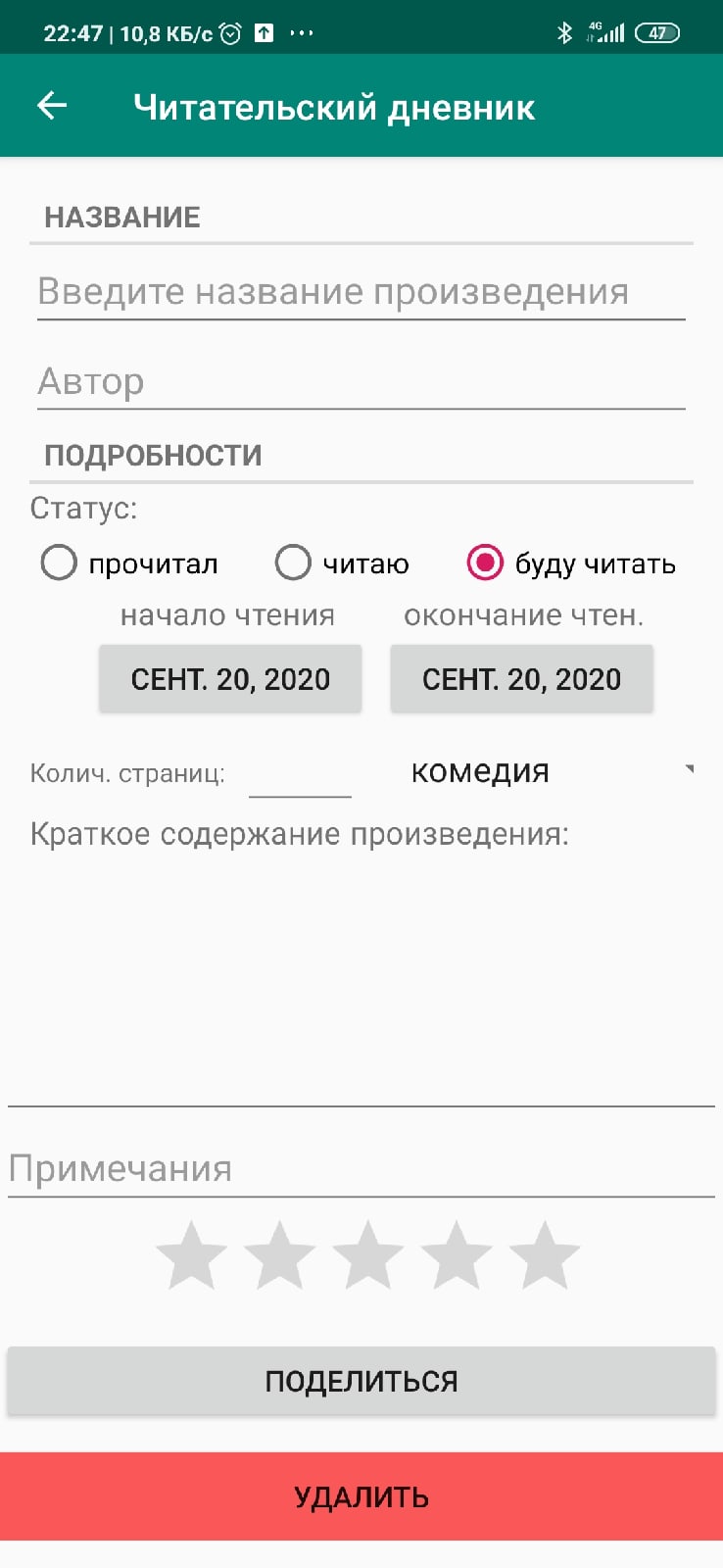
 

Рисунок № 6 – макет fragment\_book\_list, Рисунок № 7 - макет fragment\_book

заполненный данными

3. Описание функциональности

При запуске приложения перед пользователем открывается главная страница приложения. На ней расположен список книг, добавленных пользователем. На каждом элементе списка обозначено название произведения, его автор и состояние (прочитано, читаете или будете читать в скором времени).

При нажатии на какой-нибудь элемент списка, открывается новая страница приложения, в которой представлена характеристика выбранного произведения. В данной активности представляются данные, введенные пользователем. Это: название, автор, состояние, дата начала и окончания чтения, количество страниц, жанр, краткое содержание, примечания, ваша оценка произведению (от 1 до 5). Также функционал приложения позволяет поделиться описанием книги с друзьями и, естественно, удалить ее из списка.

Вернемся на главную страницу приложения. Сверху, над списком, находится панель инструментов. Первая слева кнопка(“**+**”) позволяет добавить новое произведение в список. Далее идет кнопка сортировки. Если нажать на нее, то появится окно выбора типа сортировки. Список можно отсортировать по различным критериям, а именно, по названию книги, автору, статусу, дате начала и окончания чтения, жанру, вышей оценке, количеству страниц. Следующая кнопка панели инструментов – дополнительное меню, в котором находятся не столь важные по сравнению с остальными функциями приложения, но тем не менее необходимые пользователю элементы интерфейса.

* + 1. Кнопка, вызывающая окно статистики элементов списка.
    2. Кнопка, вызывающая окно – инструкцию о работе с приложением.
    3. Кнопка, нажатие на которую позволяет вам увидеть количество произведений в списке.

Важной функциональной возможностью приложения является жесты. С их помощью пользователь способен перетащить элемент от одного места в другое. Для этого достаточно, просто удерживая определенный элемент, перетащить его в другое место. Таким образом, человек сам выбирает порядок элементов в списке. А также есть возможность удалить элемент из списка, просто смахнув его вправо или влево.

1. Тестирование

В процессе разработки приложения производилось поэтапное тестирование с целью выявления ошибок в коде. Для этого был созданы эмуляторы смартфона и планшета с разными диагоналями экрана для разных версий Android. Тестируемый программный продукт последовательно запускался на этих эмуляторах, его поведение анализировалось, и при необходимости по результатам анализа вносились изменения в код.

В результате тестирования появилась необходимость сделать более удобный интерфейс для планшетов с большой диагональю экрана. Для такого типа устройств была произведена отдельная активность, которая помещала в себя два фрагмента. Первый фрагмент – с левого края экрана – выводит список элементов, а второй фрагмент, занимающий большую часть экрана, представляет детализацию выбранного объекта. Именно благодаря тестированию приложения были найдены и исправлены ошибки в коде и некоторые недочеты в представлении интерфейса.

*Были проведены приведенные ниже тесты.*

1. Каждая активность была подвергнута тестированию с целью выявления ошибок, вызванных несоответствием ожидаемых и полученных параметров. Для этого для каждой активности был создан специальный юнит-класс, посылающий в активность все возможные параметры (производился перебор всех возможных значений). При аномальном поведении активности или ее сбое, мною анализировалось поведение программы и ошибка исправлялась.

2. Приложение было запущено на устройствах, работающих под управлением разных версий Android с целью выявления особенностей работы приложения, запущенного в разных операционных системах.

3. После завершения цикла разработки, программный продукт тестировался на реальных устройствах.

1. Возможные варианты развития приложения
   * 1. Реализация интерфейса на различных языках (не только русский и английский язык)
     2. Внедрение более удобного способа внесения информации. Например, с помощью голосовых сообщений.
     3. Автоматический поиск информации в интернете по данной книге.
2. Заключение

**Было разработано** приложение для мобильных устройств с ОС Android, которое поможет следить за прогрессом в чтении произведений и облегчит человеку ведение читательского дневника.