МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт **математики, физики и информационных технологий**

Кафедра **«Прикладная математика и информатика»**

02.03.03 МАТЕМАТИЧЕСКОЕ ОБЕСПЕЧЕНИЕ И АДМИНИСТРИРОВАНИЕ ИНФОРМАЦИОННЫХ СИСТЕМ

ТЕХНОЛОГИЯ ПРОГРАММИРОВАНИЯ

БАКАЛАВРСКАЯ РАБОТА

на тему «Разработка мобильного приложения для электронной библиотеки с полнотекстовым поиском»

Студентка	Д.С. Сарвилина		
	(И.О. Фамилия)		(личная подпись)
Руководитель	С.В. Мкртычев		
	(И.О. Фамилия)		(личная подпись)
Консультанты	А.В. Москалюк		
	(И.О. Фамилия)		(личная подпись)

Допустить к защите

Заведующий кафедрой к.т.н., доцент кафедры ПМИ, А.В. Очеповский
(ученая степень, звание, И.О. Фамилия)

_____
(личная подпись)

«_____»_____20_____г.

Тольятти 2018

АННОТАЦИЯ

Тема выпускной квалификационной работы Разработка мобильного приложения для электронной библиотеки с полнотекстовым поиском.

Ключевые слова: мобильное приложение, электронная библиотека, полнотекстовый поиск, операционная система.

Объектом исследования является полнотекстовый поиск.

Предметом данной работы является мобильное приложение для электронной библиотеки. В ходе работы были выполнены задачи: рассмотрение вопросов о разработке приложений для ОС Android; проводился анализ операционных систем; проектировалось, разрабатывалось и тестировалось мобильное приложение на базе ОС Android.

В теоретической части рассматривается вопрос о разработке мобильных приложений на базе ОС Android. Так же, между ОС Android, ОС iOS и ОС Windows Phone проводится сравнение критериев, таких как производительность, доступность (цена), безопасность, многозадачность и так далее. Итогом этого сравнения является то, что наиболее полно сформулированным требованиям соответствует ОС Android. Практическая часть посвящена концептуальному и логическому моделированию мобильного приложения для электронной библиотеки с полнотекстовым поиском. С помощью данного приложения можно просматривать тексты книг, искать по содержанию, наименованию и имени автора.

По итогу проделанной работы реализовано мобильное приложение на базе ОС Android, в котором была автоматизирована система библиотеки с полнотекстовым поиском. Данное приложение может быть использовано в различных учебных заведениях, как альтернатива научной библиотеке.

Данная работа состоит из 63 страниц, включая в себя введение, три главы, заключение, список используемой литературы из 27 источников и 4 приложений.

ABSTRACT

The topic of the graduation work is Development of a Mobile Application for an Electronic Library with Full-Text Search.

The object of research is full-text search.

The subject of this work is a mobile application for the electronic library. In the course of the work the following tasks are completed: examination of issues concerning development of applications for the Android OS, analysis of operating systems, development and testing of the mobile application based on the Android OS.

The theoretical part deals with the development of mobile applications based on the Android OS. We also conduct a comparative analysis of the Android OS, iOS and Windows Phoned based on such criteria as performance, availability (price), security, multitasking, etc. We come to a conclusion that Android fully meets the requirements formulated. The practical part is devoted to the conceptual and logical modeling of the mobile application for an electronic library with full-text search. This application makes it possible to view the texts of books, search by content, title and author's name.

As a result of the work done, a mobile application based on the Android OS is implemented with automated full-text search library system. This application can be used in various educational institutions, as an alternative to a scientific library.

This work consists of 63 pages, including the introduction, three chapters, conclusion, and a list of used literature from 27 sources and 4 applications.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Сегодня почти не осталось людей, которые хотя бы раз в жизни не пользовались смартфоном, планшетом или другим мобильным устройством, ведь это очень удобное средство для общения, работы, организации рабочего процесса, создания заметок.

Устройство смартфона в основном состоит из нескольких блоков, таких как: память, память для хранения данных, процессор, который отвечает за организацию вычислений и радио-модуль, состоящий из передатчика и приемника, и отвечает за связь. Все возможности устройства зависят от операционной системы и ее версии.

Мобильные продажи растут во всем в мире, и в связи с этим, так же, растет и спрос на приложения для них. В XXI веке любую компанию нельзя вообразить без специализированных и мобильных программ, при помощи которых можно руководить базами данных либо держать под контролем состояние продукта на рынке.

Библиотека очень важна в учебно-воспитательной и научно-исследовательской работе любого учебного заведения. Электронная библиотека обычно предполагает собой набор упорядоченных документов, находящихся на сервере, к которым возможно получить доступ в любой момент. Электронные библиотеки имеют такую важную отличительную черту, как неотъемлемое присутствие заданной структуры и навигационно-поисковые средства, которые гарантируют ориентирование в документах.

В наше время, печатные издания теряют свое предназначение из-за того, что неудобно изменять информацию, которая устарела, быстро. На смену печатным изданиям приходят электронные библиотеки, которыми люди могут воспользоваться в любой момент. Про электронные библиотеки рассказывается подробнее во второй главе.

Тема выпускной квалификационной работы – разработать мобильное приложение для электронной библиотеки с полнотекстовым поиском.

Предмет исследования: мобильное приложение для электронной библиотеки.

Объект исследования: полнотекстовый поиск.

Цель работы: разработать мобильное приложение для электронной библиотеки с полнотекстовым поиском.

Чтобы достичь установленной цели выпускной квалификационной работы следует выполнить следующие задачи:

1. Рассмотреть вопрос о разработке программ для операционной системы Android и использовать полученные знания для реализации мобильного приложения.
2. Выполнить описание техники безопасности программиста при разработке мобильного приложения.
3. Анализ существующих средств разработки мобильного приложения.
4. Изучить литературу по разработке мобильного приложения на Android.
5. Проектирование, разработка и тестирование мобильного приложения.

В первой главе выполнен обзор имеющихся мобильных платформ, рассмотрение электронных библиотек и полнотекстовых баз данных и рассматривается ОС Android, стадии ее развития и архитектура.

Во второй главе происходит проектирование мобильного приложения на базе ОС Android. Также формируются требования к будущему приложению, и выполняется логическое и функциональное моделирование разрабатываемого мобильного приложения.

В третьей главе происходит реализация разрабатываемого приложения. Описывается выбор средств разработки приложения, происходит реализация, описание работы и тестирование мобильного приложения.

В заключении подводятся итоги проделанной выпускной квалификационной работы.

ГЛАВА 1 АНАЛИЗ ИНФОРМАЦИИ И ПОСТАНОВКА ЗАДАЧИ

1.1 Обзор существующих мобильных платформ

Мобильное приложение (англ. «Mobile app») – программное обеспечение, которое предназначено для работы на планшетах, смартфонах и различных мобильных устройствах.

Мобильное приложение – это специальный пакет, устанавливаемый пользователем через магазины приложений (Play Market, App Store) на мобильное устройство.

Существует очень важная особенность, отличающая обычный мобильный телефон от смартфона – это наличие операционной системы. Наиболее используемыми операционными системами и платформами для смартфонов являются: [19]

1. Symbian OS. До конца 2010 года данная операционная система была крайне востребована на рынке операционных систем для смартфонов. В начале 2010 года на базе данной операционной системы осталась лишь одна платформа под названием Series 60, которая используется в мобильных устройствах Nokia и каких-то определенных моделях Samsung.
2. BlackBerry OS (RIM) – мобильные устройства, созданные на данной системе обладают широким спросом в США. Спецслужбы некоторых стран не достаточно заинтересованы в использовании подобных смартфонов в своей стране, так как вся входящая и исходящая информация шифруется специально разработанными методами шифрования AES.
3. Windows Mobile и Windows CE. Данная операционная система создана компанией Microsoft, работающая и выпускается с 1996 года. До 2010 года была очень популярна на рынке операционных систем для смартфонов.
4. Windows Phone 7 – разработка компании Microsoft, которая кардинально отличается от Windows Mobile.
5. Palm OS – данная платформа была очень популярной несколько лет назад. Сегодня же, на рынке изредка встречаются телефоны на данной платформе.

6. Linux. Данная операционная система не занимала широкое распространение, однако считается перспективным направлением. Смартфоны, созданные на базе Linux, в основном распространены в Азии.

7. Bada – это мобильная платформа, которая разработана компанией Samsung, и, первоначальным телефоном на данной платформе стал S8500 Wave.

8. Android – это мобильная операционная система, широко используемая во множестве цифровых проигрывателей, научных часов и нетбуков. Данная операционная система базируется на ядре Linux и собственной реализации виртуальной машины Java от Google. Сначала система находилась под разработкой компании Android, и лишь в 2005 году, её выкупила компания Google. На Android можно вести разработку приложений, написанных на языке Java, а также, подключения сторонних библиотек от Google.

9. iOS – это мобильная операционная система, разрабатываемая и выпускаемая компанией Apple. Как показывает статистика, число пользователей операционной системы iOS стремительно растёт. Согласно последнему отчету компании Canalys, основной компании по анализу high-tech устройств, предоставленная операционная система занимает ориентировочно 70% рынка мобильных устройств. Таким образом, можно смело утверждать, что платформа зарекомендовала себя с весьма хорошей стороны и благодаря этому, большинство разработчиков выпускают приложения именно на платформе iOS.

Таким образом, проанализировав и сравнив плюсы и минусы различных мобильных платформ, сделан вывод, что Android является наиболее доступной и эффективной платформой для данной работы.

1.2 Существующие электронные библиотеки и полнотекстовые базы данных

Первые электронные библиотеки были разработаны еще в 60-х годах в США, когда возникала необходимость в упорядочивании больших объёмов данных.

В 1950-1970-х годах было положение, характеризующееся термином, который был весьма популярен в то время – «информационный взрыв». Значение этого явления заключается в достижении максимального порога, который ограничивает способности изучения информационного массива возможным потребителем информации. Количество различных книг, которые принадлежат к какой-то определённой сфере деятельности, превысило физические способности человека к изучению новых знаний.

С точки зрения профессиональной информационной деятельности, именно подобные электронные библиотеки являются главной сферой деятельности в сети Интернет. Так как именно в аналогичных библиотеках собрана основная текстовая информация.

С точки зрения возможностей использования в информационной деятельности, электронные библиотеки можно разделить на два типа: бесплатные источники информации, такие как обучающие статьи, различные блоги, коллекции текстов и платные (коммерческие) полнотекстовые базы данных, доступ к которым возможно приобрести только при приобретении подписки. [21]

Подобные базы данных начали разрабатываться в начале 80-х годов, когда массово началась их разработка в разных странах. Присутствовали фактографические базы данных, которые содержали в себе фактические сведения, библиографическую информацию и полнотекстовые. Среди наиболее известных производителей в "до интернетовский" период выделялись:

1. LEXIS/NEXIS. Это один из основных и крупнейших комплексов баз данных, который включает в себя, в общей сложности, свыше 31 тысячи источников.

2. Dialog. Это первая онлайновая информационно-поисковая система в мире, включающая 573 базы данных.

3. Silver Platter. Представлено наиболее 250 баз данных, принадлежащие естественным и общественным наукам, бизнесу, финансам, сельскому хозяйству, медицине и фармакологии.

4. EBSCO Information Services. В составе данных информационных ресурсов находится наиболее 50 полнотекстовых баз данных, в числе которых содержатся материалы буквально по всем отраслям знаний.

5. STN Internetional. Более 200 баз данных исключительно академической направленности.

Среди выше перечисленных электронных библиотек не было найдено существующего мобильного приложения для просмотра и чтения книг, содержащихся в базах.

1.2.1 Развитие библиотеки: от традиционной до электронной

Библиотека – это организация культуры, которое организует сбор, сохранение и общественное использование печатных произведений. В настоящее время, та эпоха, когда основной объем знаний сберегался на печатных носителях, подходит к своему завершению. На смену ей приходит более удобное решение – электронные библиотеки. Основным преимуществом электронного аналога подобных библиотек является то, что они могут содержать не только текстовые данные, но и графические, включая анимацию, звук и видео. Также, неудобство классических библиотек и бумажных носителей заключается в том, что информация быстро устаревает, и нет возможности быстро её обновить. Высокая стоимость материалов тоже играет немаловажную роль.

Все библиотеки можно разделить на три типа:

1. Аналоговая (бумажная) библиотека – это классическая библиотека с карточными каталогами.

2. Автоматизированная (смешанная) библиотека – это аналоговая библиотека, которая содержит компьютерный каталог.

3. Электронная (цифровая) библиотека – это автоматизированная библиотека, в которой существенная доля информации располагается в электронном формате.

Электронные библиотеки с каждым днём становятся всё более и более популярными, ведь это очень эффективная альтернатива обычным, классическим библиотекам, ввиду снижения затрат на бумагу, чернила для печати. Однако активное развитие подобных электронных библиотек оказывает значительное воздействие на развитие классических библиотек.

В современном мире очень активно развиваются Интернет-ресурсы с полнотекстовым видом поиска. Благодаря подобным Интернет-ресурсам, пользователь получает уникальную возможность производить полнотекстовый поиск, а также искать большое количество информации в интернете. Благодаря таким сервисам, пользователю становятся доступны наиболее точные источники информации, а также, книги.

В функции электронных библиотек входят возможности поиска, редактирования и сохранения текстовой информации. На данный момент, очень много мобильных библиотек имеют меню иерархического вида. Таким образом, доступ к нужной информации упрощается. Благодаря механизмам полнотекстового поиска, пользователь имеет возможность найти любой интересующий его текст по ключевым словам, которые в нём находятся. Такой механизм реализуется при помощи поиска вхождений строки во всех имеющихся книгах.

На данный момент в Интернете существует множество открытых и бесплатных электронных библиотек. Наиболее крупными электронными библиотеками с полнотекстовым поиском являются: [21]

- библиотека художественной литературы Максима Мошкова (www.lib.ru);
- студенческую библиотеку (www.lib.students.ru);
- электронную библиотеку на сайте Эдварда Радзинского (www.radzinski.ru);
- библиотеку детской литературы «Сказки» (www.skazki.com.ru);
- Классика.ру (www.klassika.ru);
- Проза.ру (www.proza.ru);

- BestBooks.RU (www.bestbooks.ru) и так далее.

Таким образом, сравнив традиционную библиотеку и электронную, выбор очевиден. Ведь электронная библиотека – это не только удобно, но и более выгодно.

1.3 Существующие мобильные приложения для электронных библиотек

Мобильный Интернет расширяет возможности пользователей. Ведь благодаря нему, пользователь может заходить и просматривать сайт библиотеки из любого места. Множество различных сайтов подобных ресурсов сильно перегружены различной ненужной информацией. Таким образом, различные мобильные приложения предусматривают создание удобной версии сайта, которая была бы отображена правильно на дисплее телефона. К сожалению, на данный момент ещё не все сайты перешли на подобную мобильную версию. Это означает, что на мобильном устройстве просматривать подобный контент будет не очень удобно. Для того, чтобы страница того или иного сайта верно отображалась на экране смартфона пользователя, необходимо создать отдельную версию для мобильных устройств, которая будет включать в себя ту же страницу, но немного в другом формате и разрешении.

Вместо того, чтобы делать отдельную веб-страницу в другом формате для мобильного устройства, разрабатывается мобильное приложение. При помощи такого приложения можно повысить удобство работы пользователя, а также, предоставить удобный доступ к различным функциям в мобильном формате, таким как: электронная почта, личный кабинет и так далее. [15]

Примерами можно назвать продукты компаний:

- arXiv;
- ARTstor;
- Royal Society of Chemistry;
- Oxford University Press;
- Wiley;
- Association for Computing Machinery;

- American Chemical Society;
- American Mathematical Society.

Для пользователя существует множество мобильных приложений (мобильных библиотек) для комфортного чтения и поиска книг. Ниже представлены популярные приложения электронных библиотек: [15]

1. Cool Reader. Приложение предоставляет простой дизайн и функции ч целью создания индивидуальной мобильной библиотеки. Так же, в данном мобильном приложении есть возможность изменять шрифт, яркость подсветки и внешний вид страницы. При желании, настраиваются действия для всех кнопок и зон экрана.

2. eRSL – это приложение Российской государственной библиотеки, электронные каталоги которой можно просматривать не только через веб-браузер, но также через специальное мобильное приложение. Данное мобильное приложение дает возможность просматривать предельно полную информацию о разных библиотечных изданиях или различных других документах.

3. Aldiko Book Reader. Предоставленное мобильное приложение имеет дневную, и ночную тему, удобный контроль яркости, изменение шрифта и визуализированную книжную полку.

4. Ebook Search. Данное приложение содержит более 8 миллионов книг в бесплатном доступе. Приложение на английском языке.

5. FB Reader. Приложение способно функционировать со словарями, использовать магазин «Литрес», подключать OPDS-каталоги сетевых библиотек и, кроме того, комфортно и четко отображать содержимое библиотек с разбивкой согласно создателям, жанрам и категориям.

6. iReader. Приложение с приятным интерфейсом, книжной полкой, анимированным перелистыванием и фоном древесного цвета.

Все рассмотренные приложения предоставляют достаточно удобный для пользователя интерфейс для чтения книг. Приложение, разрабатываемое в ходе данной работы, будет основано на приведенных выше.

1.4 Сравнение мобильных операционных систем

iOS. Это операционная система, созданная компанией Apple. Данная операционная система содержит систему закрытого типа.

Данная операционная система обладает следующими особенностями: [10]

- стабильность работы, что обусловлено жесткой связкой железа с программным обеспечением;
- незначительное количество вирусов, и по этой причине можно не ломать голову над тем, какой антивирус установить на смартфон;
- простота изучения операционной системы, ведь спустя несколько часов работы с системой уже можно с легкостью назвать себя опытным пользователем;

Минусы данной операционной системы:

- Цена. Наиболее высокая цена, если сравнивать с другими устройствами-аналогами, а также, обслуживание;
- Если необходимо скачать какое-либо приложение, приходится делать это с помощью специальной программы. Обычные приложения будут закрыты для пользователя из соображений безопасности операционной системы;
- Большинство приложений и программ платные;
- Невозможность передать какой-либо файл по технологии Bluetooth из соображений безопасности;
- Отсутствует возможность использования карты памяти.

Android. Платформа, разработанная компанией Google. Данная операционная система открытого типа, что делает её практически незаменимой для разработчика приложений и игр. Для данной операционной системы существует множество сторонних программ, улучшающих работу вашего смартфона. И для этого не нужно никаких разрешений.

В операционной системе Android существует больше возможностей для настройки интерфейса под себя, чем в операционной системе iOS. Устройства на базе данной операционной системы более производительны: в ней частота процессора выше и количество ядер больше. Кроме того, в устройствах на базе

операционной системы Android имеется возможность расширения памяти, то есть, вероятность применять карты памяти. [10, 18]

Стоимость устройств на базе ОС Android значительно ниже, нежели устройства на базе операционной системы iOS, что дает возможность подобрать модель под свой кошелек.

Особенности ОС Android:

- личный магазин приложений;
- абсолютная поддержка Google (собственно, вся работа в Android происходит с помощью аккаунта в Google);
- возможность установки модификаций системы;
- возможность установки различных приложений, как с официального Google-Play, так и из других источников;

Минусы ОС Android:

- система нестабильна и часто функционирует некорректно;
- наличие большого количества вредоносных программ, которые легко можно скачать в интернете;
- малая ёмкость аккумулятора, по сравнению со смартфонами-конкурентами.

Операционная система Android удобна простым пользователям, которые не используют телефон, как рабочее пространство. К примеру, не выполняют ежедневно объёмные приложения и задачи.

Windows Phone. Относительно новейшая операционная система, разработанная компанией Microsoft. Windows Phone – это некая середина между iOS и Android, в плане закрытости системы. В Windows Phone больше возможностей персонализации чем в iOS, но меньше, чем в Android. [10]

Особенности операционной системы Windows Phone:

- при вводе текста с клавиатуры, происходит моментальная проверка на правописание;
- работа в сети Интернет совершается с помощью Internet Explorer Mobile;

- наличие уникальной возможности просмотреть информацию о каждом пользователе в телефонной книге;
- абсолютная синхронизация с социальными сетями;
- возможна полная синхронизация с ПК, на котором установлена операционная система Windows;
- более качественное и «интересное» оформление оболочки;
- значительная независимая работа устройств, работающих на основе данной ОС.

К минусам операционной системы Windows Phone относятся:

- недостаточное развитие операционной системы;
- по сравнению с операционной системой Android, отсутствует разнообразный выбор приложений.

У каждого человека, пользующего мобильным устройством, свои потребности и возможности. Ниже представлена таблица, в которой сравниваются мобильные операционные системы.

Таблица 1.1 – Сравнение мобильных операционных систем

	iOS	Android	Windows Phone
Доступность (цена)	-	+	+
Производительность	+	+	-
Оптимизация	+	-	+
Защита (безопасность)	+	+	-
Большой выбор бесплатных приложений	-	+	-
Дизайн пользовательского интерфейса	+	+	+
Многозадачность	+	+	+
Система уведомлений	+	+	+
Голосовое управление	+	+	+
Быстрое обучение	+	+	+
Итого	8	9	7

На основании данного анализа можно сделать вывод, что более полно сформулированным требованиям соответствует ОС Android, так как данная операционная система имеет большинство преимуществ в сравнении с другими мобильными операционными системами.

1.5 Стадии развития операционной системы Android

Android (рус. «Андроид») – это портативная операционная система, которая основана на ядре Linux. Первоначально данная операционная система была сформирована компанией Android Inc., которую в 2005 году приобрела компания Google.

Android предоставляет возможность разрабатывать java-приложения, которые управляют устройством с помощью библиотек, разработанных компанией Google.

На рисунке 1.1 показан логотип операционной системы Android. [18]

Рисунок 1.1 – Логотип Android

Android используется в большом ассортименте устройств, таких как:

- смартфоны;

- телевизоры;
- электронные книги;
- медиаплееры;
- фоторамки;
- часы;
- ноутбуки и так далее.

Отличительной особенностью Android является то, что некоторые приложения уже заранее предустановлены и интегрированы в систему, что позволяет повысить удобство использования за счёт синхронизации данных между устройством пользователя и серверами. К примеру, в браузере Chrome имеется возможность синхронизации устройств. То есть, Ваши закладки в браузере на телефоне, история просмотров и многое другое, могут быть синхронизированы с компьютером.

За кодовым названием версии ОС Android следует наименование какого-либо десерта. Первые буквы наименований версий по порядку, соответствуют буквам латинского алфавита. Ниже приведены версии ОС Android и их названия: [9, 19]

- 1.5 Cupcake («кекс»);
- 1.6 Donut («пончик»);
- 2.0/2.1 Éclair («эклер» или «глазурь»);
- 2.2 Froyo («сокращение от «замороженный йогурт»);
- 2.3 Gingerbread («имбирный пряник»);
- 3.0 Honeycomb («медовые соты»);
- 4.0 Ice Cream Sandwich («брикет мороженого»);
- 4.1/4.2/4.3 Jelly Bean («желейная конфета»);
- 4.4 KitKat («КитКат»);
- 5.0 Lollipop («леденец»);
- 6.0 Marshmallow («маршмэллоу»);
- 7.0 Nougat («нуга»);

- 8.0 Oreo («Орео»).

К достоинствам операционной системы Android можно отнести следующее:

- полностью открытая операционная система;
- доступность для различных аппаратных платформ;
- огромное количество программ и приложений;
- поддержка многопользовательского режима;
- наличие различных прошивок, в том числе и неофициальных.

Минусы операционной системы Android:

- операционная система может подвисать;
- недостаточный уровень безопасности;
- необходимость использования закрытых приложений и заключения контракта разработчиками для доступа к Google Play.

Также, к достоинствам ОС можно отнести легкость создания приложения благодаря среде разработки Android Studio, предоставляющей удобный интерфейс.

1.5.1 Архитектура ОС Android

На рисунке 1.2 представлена компонентная модель Android в виде иерархии. [8]

Архитектура ОС Android стоится из таких уровней, как: ядро операционной системы, набор библиотек, фреймворки приложений и уровень приложений. Также, на одном уровне с набором библиотек (Libraries) находится Android Runtime – среда выполнения прикладных программ.

Все приложения, которые запускаются на Android, написаны на языке Java, однако, есть вероятность разрабатывать программы на C/C++ (с помощью Native Development Kit).

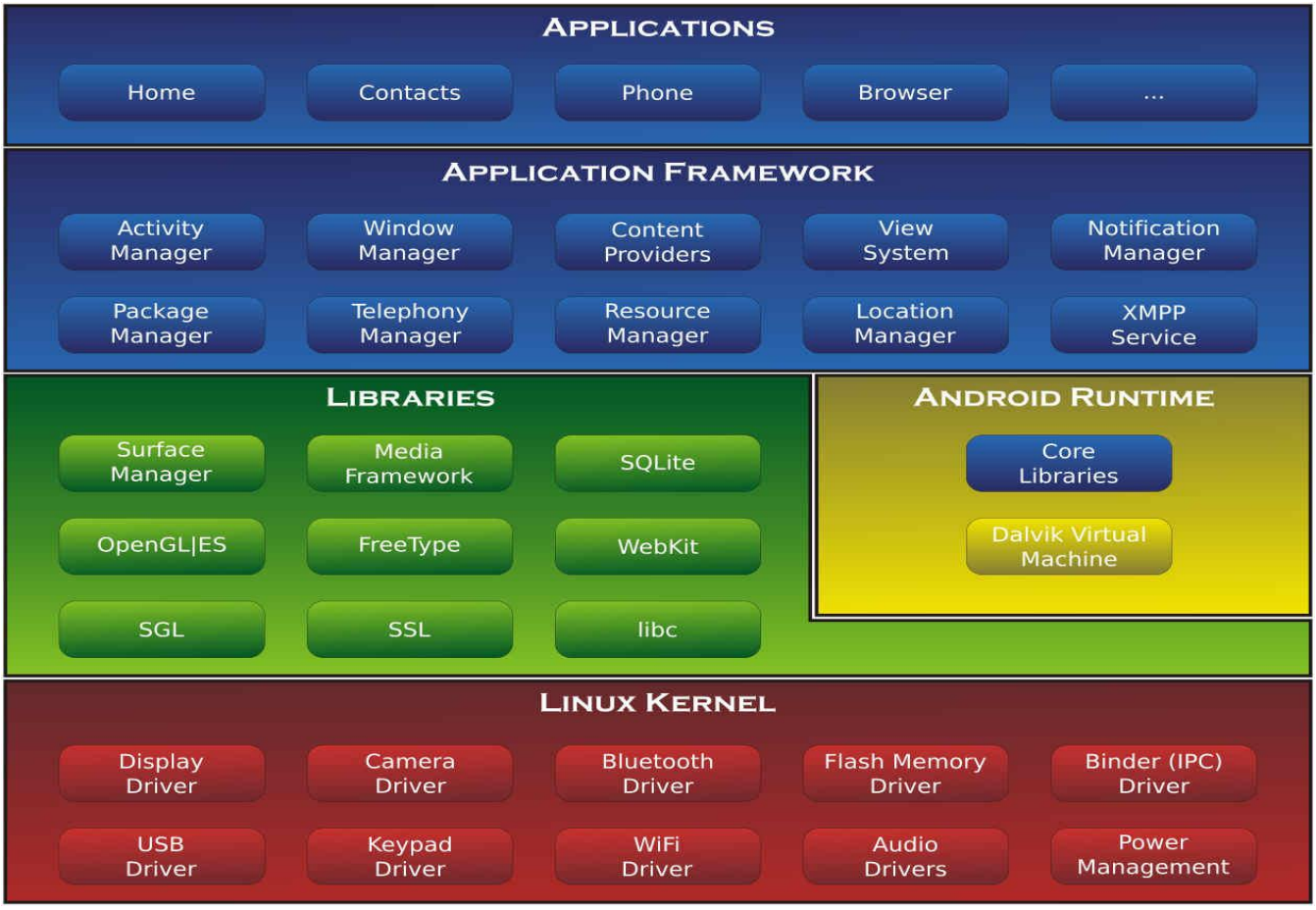

Рисунок 1.2 – Архитектура ОС Android

В самой основе располагается ядро операционной системы. Ядро обеспечивает ключевые функции системы, действует как уровень между аппаратным и программным обеспечением. Также, ядро отвечает за безопасность использования смартфона, защиту от вирусов и вредоносных программ.

Выше ядра находится Libraries (набор библиотек), который специализирован для решения типовых задач, требующие высокой производительности. Этот уровень дает возможность осуществлять алгоритмы для вышележащих уровней, таких как поддержку файловых форматов, осуществление кодирования и декодирования данных и многое другое. Далее представлены некоторые из низкоуровневых библиотек:

1. Surface manager – в ОС Android применяется наиболее упрощенный композитный менеджер окон, наподобие Compiz (Linux). Вместо отрисовки напрямую на дисплей, поступающие команды выполняются в закадровом

буфере, в котором происходит накопление различных кадров, создавая последовательность кадров, и только лишь затем выводит их на экран. Таким образом, получается создать более качественные эффекты.

2. Media Framework – это библиотеки, реализованные на основе PacketVideo OpenCORE. Данные библиотеки позволяют проигрывать аудио и видео контент, а также осуществлять вывод статических изображений. Поддерживаются большое количество форматов, к примеру, MPEG4, H.264, MP3, AAC, AMR, JPG, PNG.

3. SQLite – лёгкая и мощная СУБД, используемая как на Android, так и на Windows для разработки приложений.

4. OpenGL | ES – библиотека, широко используемая разработчиками для отрисовки 3D графики и визуальных эффектов.

5. FreeType – происходит обработка битовых карт, растеризация шрифтов, различные операции над шрифтами и текстом. Этот движок позволяет отображать текст с наложением эффектов.

6. LibWebCore – это библиотеки знаменитого браузерного движка WebKit, который также применяется в десктопных браузерах Google Chrome и Apple Safari.

7. SGL (Skia Graphics Engine) – движок для работы с 2D-графикой.

8. SSL – библиотеки, требуемые для поддержки одноименного криптографического протокола в других их программах.

9. libc – это библиотека стандартных вызовов языка C для небольших устройств. Носит название Bionic.

На этом же уровне, как было сказано выше, располагается Android Runtime. Ее составляющими элементами являются Core Libraries (набор стандартных библиотек) и Dalvik Virtual Machine (виртуальная машина Dalvik, создателем которой считается Дэн Бронштейн). Всякое мобильное приложение в ОС Android запускается в собственном экземпляре виртуальной машины Dalvik. Таким образом, все без исключения работающие процессы отделены и изолированы от ОС и друг от друга.

Структура Android Runtime: работа программ исполняется строго в рамках окружения виртуальной машины. Таким образом, происходит защита операционной системы от вероятного ущерба со стороны других ее элементов.

Выше Libraries находится Application Framework (фреймворки приложений).

Благодаря архитектуре фреймворка, предоставляются дополнительные функции для приложений. К примеру, получение доступа к другим приложениям. Основой всех приложений считается набор систем и служб. В основной набор сервисов и систем, которые лежат в основе всякого мобильного приложения и представляющихся частями фреймворка, входят:

1. Менеджер действий (Activity Manager) распоряжается жизненным циклом приложений и предоставляет систему навигации согласно работе с действиями.
2. Менеджер окон (Window Manager) распоряжается окнами, и предоставляет для приложений наиболее высокую степень абстракции библиотеки Surface Manager.
3. Контент-провайдеры (Content Providers) – службы, которые осуществляют доступ к другим приложениям и к их данным.
4. Система представлений (View System) – некоторый набор представлений, который позволяет построить внешний вид оболочки, и содержит различные компоненты для различных графических компонентов.
5. Менеджер извещений (Notification Manager) позволяет мобильному приложению в строке статуса отображать пользовательские уведомления.
6. Менеджер пакетов (Package Manager) управляет установленными пакетами на устройстве, отвечает за установку новейших и удаление имеющихся.
7. Менеджер телефонии (Telephony Manager) содержит API с целью взаимодействия со способностями телефонии, такие как звонки, смс и прочее.
8. Менеджер ресурсов (Resource Manager) необходим для доступа к строковым ресурсам, графическим ресурсам и другим типам.

9. Менеджер местоположения (Location Manager) предоставляет возможность мобильным приложениям время от времени получать обновленные сведения о текущем географическом положении мобильного устройства.

На вершине программного стека архитектуры ОС Android находится уровень приложений (Application). В составе ОС Android лежит комплект базовых приложений, таких как: календарь, карты, SMS и клиенты электронной почты, браузер, менеджер контактов и так далее.

Список интегрированных приложений способен изменяться в зависимости от версии Android и модели устройства.

1.5.2 Эмулятор для разработки мобильного приложения

Эмулятор – это программа, которая позволяет на компьютере пользоваться всеми функциями Android устройства. На основании этого можно выделить, что эмулятор считается виртуальной копией Android устройства. [12]

В основном, эмуляторы используются для разработки мобильного приложения, в частности, для выявления ошибок и различных тестов.

Недостатки эмуляторов:

- требование множества системных ресурсов;
- по причине того, что в архитектуре процессоров компьютера и смартфона имеются отличия, эмуляторы медленно запускаются;
- в некоторых случаях обычного эмулятора не хватает.

Ниже представлены эмуляторы среди самых популярных мобильных операционных систем для разработки, тестирования приложений и исправления ошибок:

- Google Android Emulator. Данный эмулятор запускается как отдельное приложение;
- Android SDK Emulator содержит в себе эмулятор мобильного устройства, который реализует все аппаратные и программные особенности стандартного устройства;

- MobiOne – это mobile Web IDE для Windows, разрабатывающая, проводящая отладку, тестирующая, упаковывающая и внедряющая мобильные веб-приложения на устройства как: iPhone, BlackBerry, устройства на Android и Palm Pre;
- TestiPhone – это симулятор, который основан на веб-браузере для быстрого тестирования веб-приложений для iPhone;
- BlackBerry Simulator. С любым из множества официальных эмуляторов BlackBerry возможна проверка того, как программное обеспечение, экран, клавиатура устройства будут взаимодействовать с приложением;
- Genymotion Android Emulator – это эмулятор Android, включающий в себя готовые и настроенные образы Android.

В ходе данной работы был выбран эмулятор Android SDK Emulator, так как он является наиболее удобным и простым.

Выводы по первой главе и постановка задачи

В данной главе был проведен анализ информации и постановка задачи. Выполнены обзоры существующих мобильных платформ, мобильных приложений для электронных библиотек, электронных библиотек и полнотекстовых баз данных. Проведен анализ сравнения мобильных операционных систем, по итогу которого видно, что наиболее популярной и простой в использовании является операционная система Android. Рассмотрена и описана архитектура операционной системы Android.

Также, рассматривались эмуляторы для разработки мобильных приложений. Был выбран наиболее эффективный и быстрый эмулятор для тестирования мобильного приложения непосредственно в момент разработки. Ввиду удобства использования прямо в среде разработки был выбран Android SDK Emulator.

Данное мобильное приложение позволит пользователю, находясь в любой точке, просматривать электронную библиотеку, установленную на сервере учебного заведения или любой другой организации.

В рамках данной работы были поставлены, а также реализованы следующие задачи:

- структурирование информации в базах данных;
- написание простейшего Android-приложения;
- реализация канала связи с сервером;
- оптимизация связи приложения с сервером и базой данных.

Поставленные задачи были распределены по срокам выполнения, а также на подзадачи для удобства.

ГЛАВА 2 ПРОЕКТИРОВАНИЕ МОБИЛЬНОГО ПРИЛОЖЕНИЯ НА БАЗЕ ОС ANDROID ДЛЯ ЭЛЕКТРОННОЙ БИБЛИОТЕКИ С ПОЛНОТЕКСТОВЫМ ПОИСКОМ

2.1 Формирование требований к разрабатываемому мобильному приложению

Мобильное приложение должно быть достаточно простым, практичным и удобным в использовании.

Требования к мобильному приложению описаны по классификации FURPS+. Данная классификация была предложена и разработана в 1992 году Робертом Грэйди Hewlett-Packard. Аббревиатура FURPS+ расшифровывается как:

- функциональные требования (Functionality);
- требования к удобству использования (Usability);
- требования к надежности (Reliability);
- требования к производительности (Performance);
- требования к возможности сопровождения (поддержка, гибкость, модифицируемость, модульность, расширяемость) (Supportability) [20].

Символ «+» обозначает дополнительные условия, которые описаны ниже:

- проектные ограничения (ограничения на технологии, средства разработки);
- требования управления системой (ресурсы и стандарты разработки, стандарты качества ПО, языки и стили программирования, ограничения на техническое (аппаратное) обеспечение);
- требования к графическому интерфейсу пользователя – ограничения, накладываемые потребностью взаимодействия с другими системами (форматы сведений, протоколы взаимодействия, внешние системы);
- физические требования – ограничения, на технические средства и окружение системы (форма, объем, температурный режим, влажность и так далее);

- юридические требования (международные соглашения, авторское право, договор о лицензировании, законодательство, отраслевые стандарты).

Функциональные требования (Functionality) содержат свойства и функции системы. К ним можно отнести:

- лицензирование – средства для отслеживания, приобретения, установки и контроля над использованием лицензий;
- почта – службы отправки и получения сообщений;
- помощь – техническая поддержка пользователей в любой момент времени;
- печать – возможность распечатать документы;
- отчетность – возможность генерирования отчётов;
- безопасность – получение и открытие доступа к какой-либо информации.

Удобство использования (Usability) делится на следующие виды требований:

- логичность и завершённость интерфейса пользователя;
- защита от человеческого фактора;
- руководство пользователя, стандартизированная документация, соответствующая госту;
- высокая квалификация пользователей и их обучение;
- справочная информация в системе.

Надежность (Reliability) содержит такие характеристики, как:

- сбои (периодичность сбоев, среднее время и серьезность сбоев, возможность восстановления системы);
- предсказуемость действия;
- период готовности системы к работе (режим работы либо период доступности);
- точность вычислений.

К требованиям производительности (Performance) относятся характеристики, как:

- скорость работы (время ответа системы);
- эффективность (результативность);
- пропускная способность (общее и возможное количество в то же время работающих пользователей, число запросов пользователя, число обращений системы к базе данных и размер передаваемых сведений в единицу времени);
- скорость восстановления;
- скорость запуска и завершения работы;
- потребление ресурсов.

К требованиям возможности сопровождения (Supportability) относятся возможности:

- тестирования;
- масштабирования;
- конфигурирования (регулярной настройки, переопределения характеристик);
- совместимости;
- сервисного обслуживания и ремонта;
- установки;
- расширения (наращивания вспомогательного функционала системы);
- приспособления к применению в установленной среде;
- портативность;
- соответствие международным стандартам.

Требования предполагают то, что должно быть реализовано.

После того, как требования к системе определены и сделан выбор мобильной операционной системы для реализации приложения, нужно выполнить моделирование мобильного приложения, для того, чтобы определить

ключевые функции, которые будут реализованы в предоставленном мобильном приложении.

2.2 Моделирование мобильного приложения для электронной библиотеки на базе ОС Android

2.2.1 Выбор архитектуры мобильного приложения

При реализации мобильного приложения главным компонентом является выбор архитектуры. Мобильные приложения в современное время разрабатываются в архитектуре «клиент-сервер». При реализации мобильного приложения используется трехзвенная архитектура. [16]

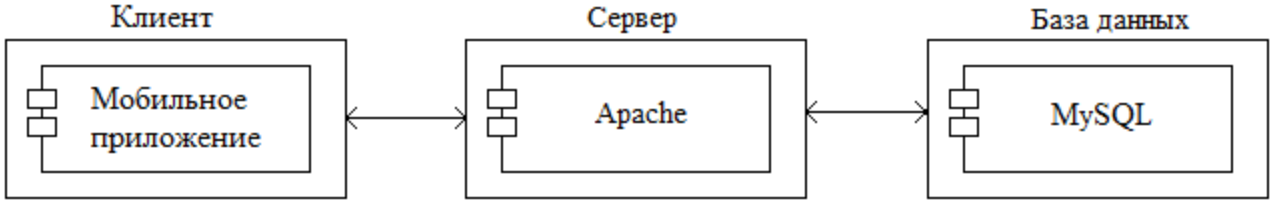

Рисунок 2.1 – Трехзвенная архитектура мобильного приложения

Архитектура разрабатываемого мобильного приложения для электронной библиотеки представлена на рисунке 2.1.

В качестве клиента представляется мобильное устройство. С целью взаимосвязи среди компонентов выступает сеть Интернет.

В качестве серверной части используется набор дистрибутивов Denwer, который предоставляет необходимые в рамках данной работы сервисы, такие как: сервер Apache, база данных MySQL. [17]

Сервер базы данных содержится на персональном компьютере руководителя электронной библиотеки, напрямую не взаимодействует с клиентом, и поэтому, повышается безопасность системы.

2.2.2 Функциональное моделирование мобильного приложения

Проектируемое мобильное приложение для библиотеки, обязано соответствовать конкретным функциональным требованиям. На рисунке 2.2

показана функциональная модель мобильного приложения, где отображены основные функции приложения.

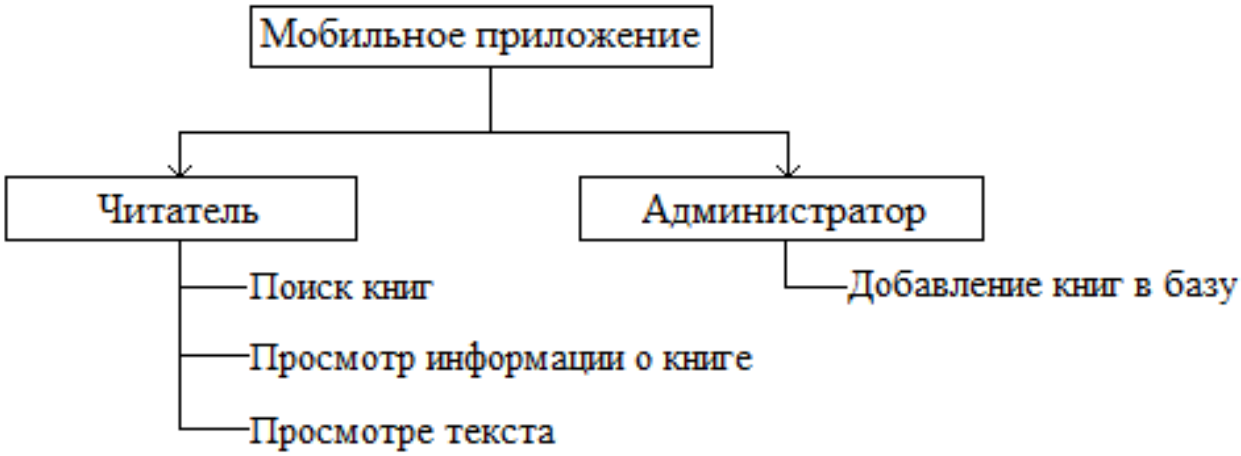

Рисунок 2.2 – Модель мобильного приложения

Основные функции мобильного приложения:

- поиск книг по ключевым словам;
- поиск книг по автору/названию;
- просмотр информации о книге;
- просмотр текста.

При открытии приложения, система выводит на экран список книг, имеющихся в базе данных. Пользователь может выбрать необходимую книгу из предложенного списка. Если нужной книги в списке нет, пользователь может ввести ключевое слово в поле для поиска. Введённая пользователем информация в виде запроса отправляется на удалённый сервер. Сервер обрабатывает запрос от пользователя и в зависимости от типа запроса, сервер обращается в базу данных и получает ответ. После чего, сервер обрабатывает ответ из базы и отправляет данные клиенту.

2.2.3 Логическое моделирование мобильного приложения

Логическое моделирование – это осуществление испытания функционирования логической схемы.

Главная задача логического моделирования заключается в осуществлении проведения проверки функции проектируемой логической схемы без абсолютной реализации на данном этапе разработки. [22]

Преимуществом предоставляемой модели считается то, что осуществляется проверка, как логических функций приложения, так и её временных соотношений.

Диаграмма вариантов использования представляет взаимоотношения между вариантами использования и действующими лицами. Варианты использования – это описание взаимодействий между пользователем и системой.

Действующее лицо, то есть пользователь, является источником, взаимодействующим с системой с помощью варианта использования.

На рисунке ниже представлена диаграмма вариантов использования мобильного приложения.

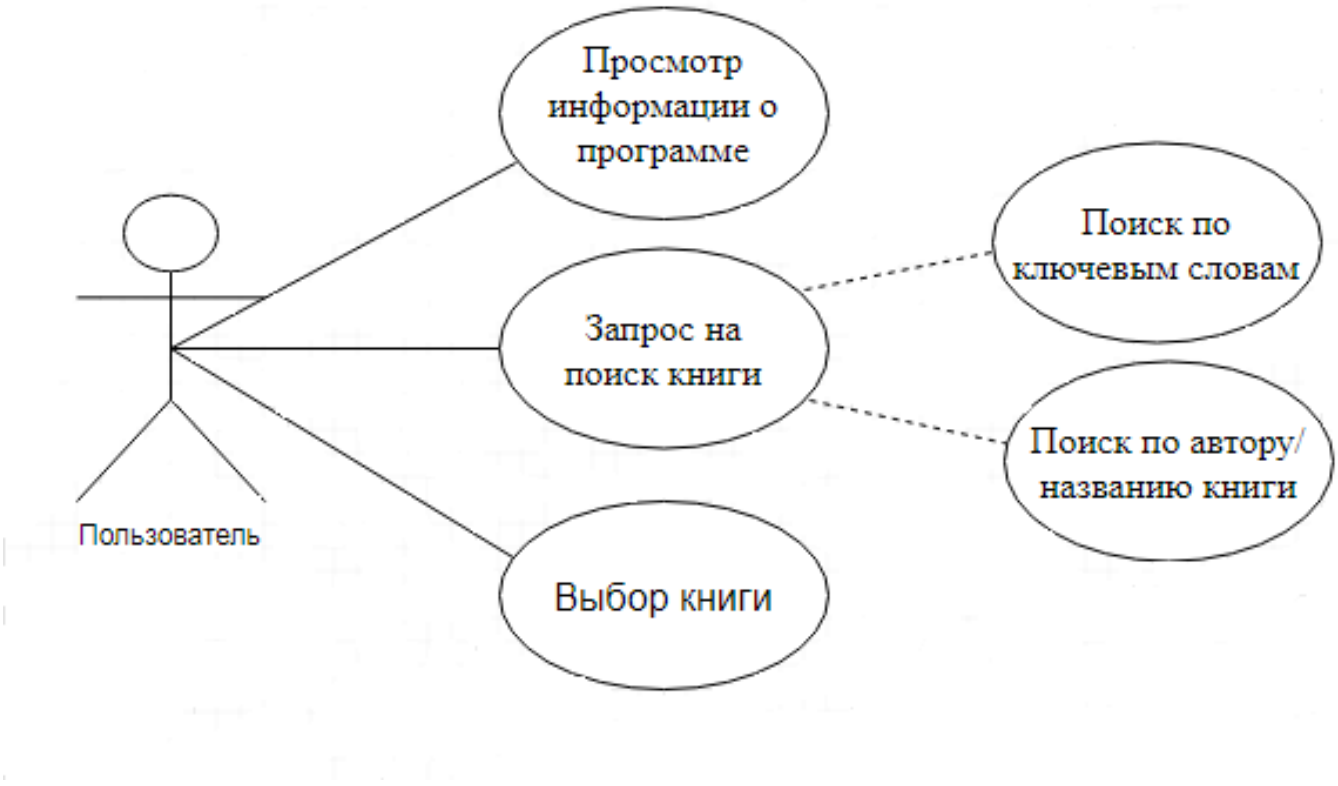

Рисунок 2.3 – Диаграмма вариантов использования

С целью реализации логического моделирования следует создать диаграмму классов мобильного приложений.

Диаграмма классов специализирована с целью отображения классов, их атрибутов и операций, связей между классами с указанием кратности.

Диаграмма классов приложения представлена на рисунке 2.4.

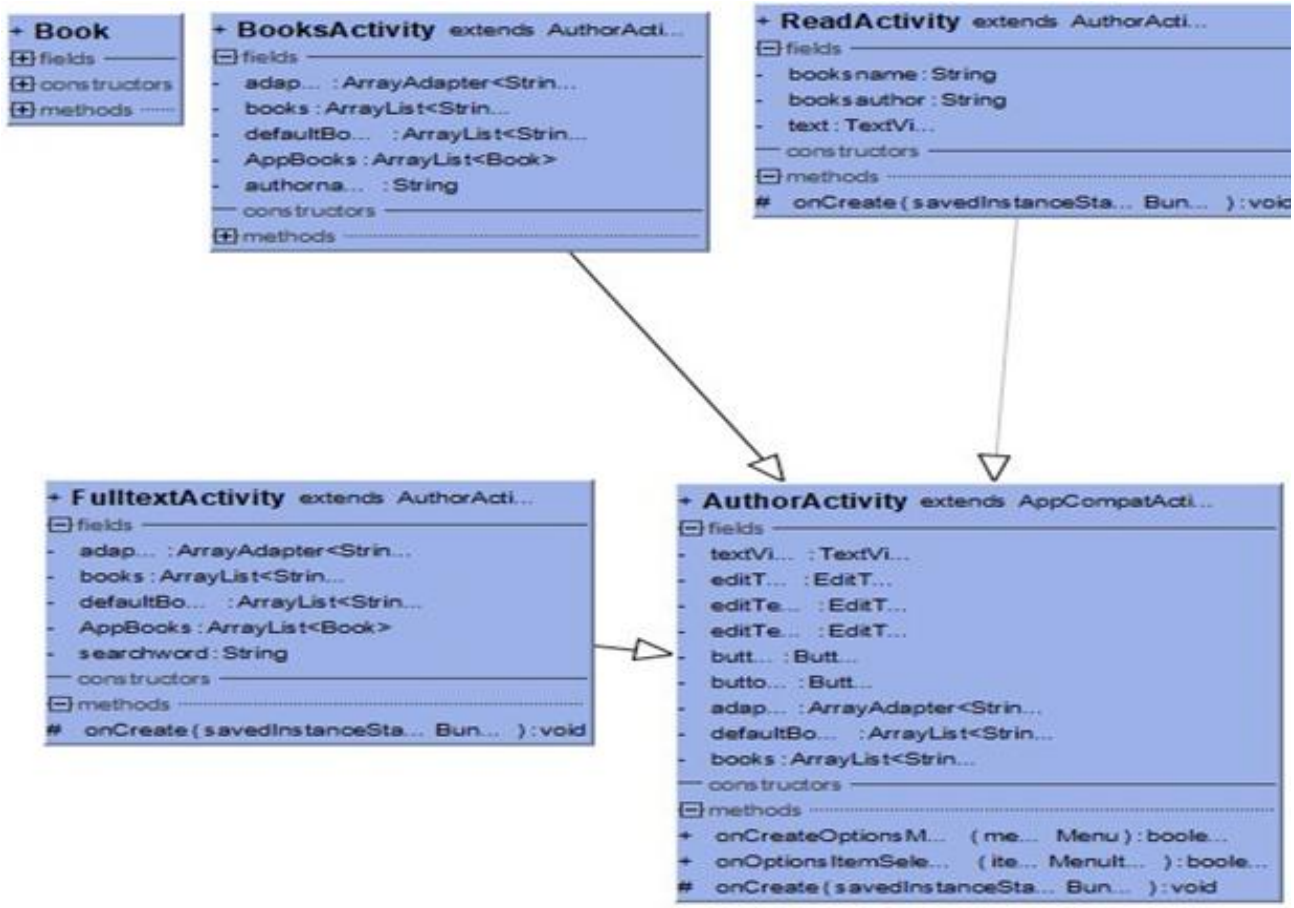

Рисунок 2.4 – Диаграмма классов

На диаграмме классов представлены классы «BooksActivity», «ReadActivity», «FulltextActivity», «AuthorActivity», «Book».

Класс «BooksActivity» – вывод списка книг на экран, который является родительским классом «AuthorActivity» – вывод списка авторов на экран.

Класс «ReadActivity» отвечает за вывод текста выбранной книги на экран. Содержит поля «booksName» – имя выбранной книги, «booksAuthor» – имя автора выбранной книги и «text» – текст книги.

Класс «FulltextActivity» содержит поля: «adapter» – адаптер для списка, выводимого на экран, «books» – сортированный список всех названий книг,

«defaultBooks» – список, используемый для временного хранения списка книг, отсортированных по названию, «AppBooks» – поле для хранения тех книг, которые были получены вследствие применения фильтра, «searchword» – слово, вводимое в поле поиска для сортировки книг.

Класс «Book» используется для создания объектов-книг с полями «name», «author», «content», для хранения имени книги, имени автора книги и текста книги соответственно.

Диаграмма последовательности – это диаграмма, на которой для определенного набора объектов на единой временной оси представлен жизненный цикл некоторого конкретного объекта и взаимодействие актеров.

На рисунке 2.5 показана диаграмма последовательности работы мобильного приложения, на которой отображено взаимодействие объектов.

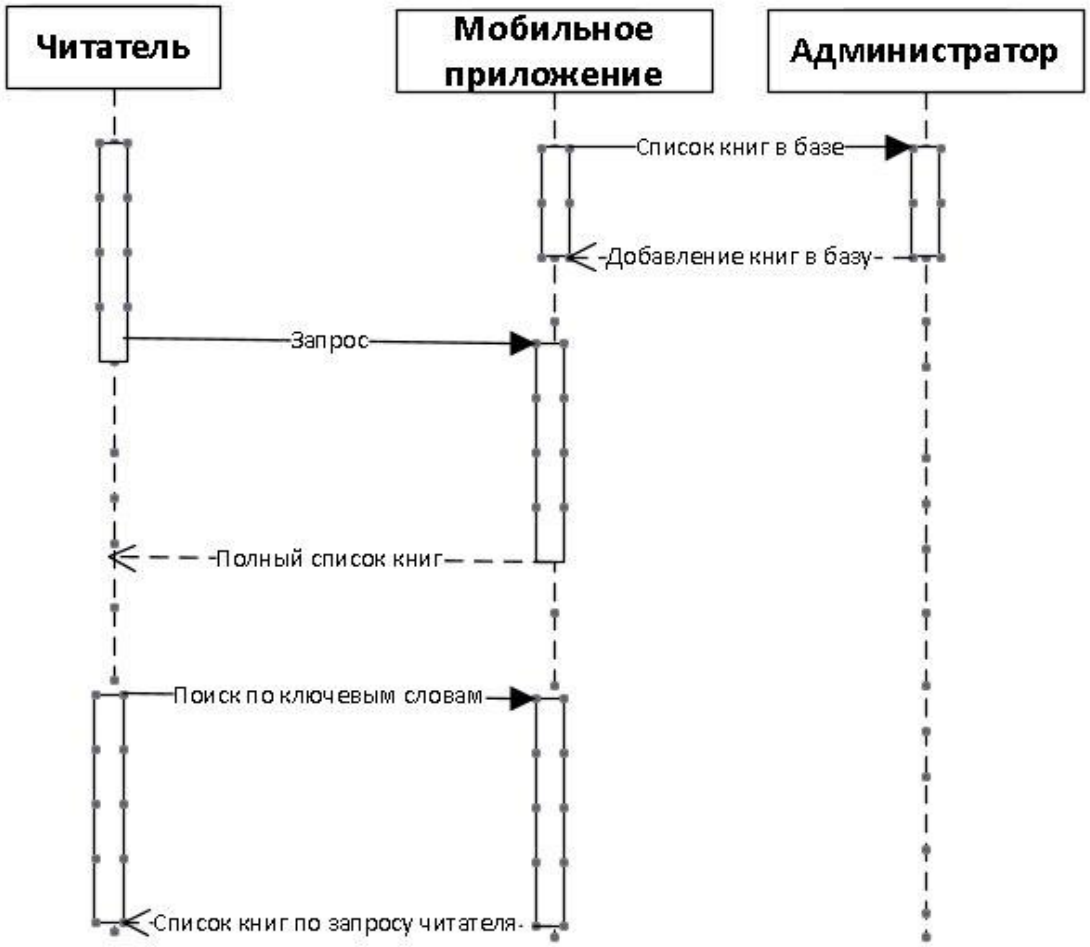

Рисунок 2.5 – Диаграмма последовательности работы мобильного приложения

Контакт между объектами и субъектами происходит следующим образом:

1. Администратор добавляет книги в базу.
2. Приложение выводи книги, находящиеся в базе.
3. Читатель делает запрос на поиск любой книги по ключевым словам.
4. Приложение выводит список книг по запросу читателя.

Таким образом, проверка логики функционирования мобильного приложения была осуществлена. Так же, описана взаимосвязь и последовательность работы в приложении.

Выводы по второй главе

В данной главе были разработаны функциональные модели и диаграммы:

- вариантов использования;
- классов:
- последовательности.

Также были сформулированы требования к мобильному приложению.

ГЛАВА 3 РЕАЛИЗАЦИЯ МОБИЛЬНОГО ПРИЛОЖЕНИЯ ДЛЯ ЭЛЕКТРОННОЙ БИБЛИОТЕКИ С ПОЛНОТЕКСТОВЫМ ПОИСКОМ

3.1 Функциональные требования к мобильному приложению

Функциональная структура мобильного приложения содержит основные подпрограммы, которые выполняют задачи обработки полученных данных, сбора, предназначенные с целью реализации процессов сбора данных от пользователя приложения. Далее перечислены технические требования к разрабатываемому мобильному приложению:

1. Разрешения при установке приложения.

Для того чтобы осуществить установленную задачу, приложение должно иметь доступ к необходимым разрешениям с целью выполнения определенных операций с устройством. Список запрашиваемых разрешений выводится на экран пользователя при установке, для того, чтобы он смог решить, стоит ли устанавливать данное мобильное приложение на свое мобильное устройство.

В файле AndroidManifest.xml описаны основные настройки, запросы и свойства на разрешения мобильного приложения, представленные в формате XML. Далее приведено разрешение, описанное в файле AndroidManifest.xml:

<uses-permission android:name="android.permission.INTERNET" />

2. Требования к эргономике и технической эстетике: реализация графического режима, простой и удобный интерфейс, который будет понятен самому не опытному пользователю;

3.2 Выбор средств разработки мобильного приложения

С целью исследования темы, для разработки мобильного приложения электронной библиотеки с полнотекстовым поиском был выбран язык программирования Java, серверная часть на PHP, MySQL и среда разработки Android Studio. [24, 27]

Java – высокоуровневый язык программирования, который разработан компанией Sun Microsystems и в последующем приобретен компанией Oracle.

Мобильные приложения, реализованные на языке Java, как правило, транслируются в специальный байт-код, и поэтому, с помощью виртуальной Java-машины, имеют шансы функционировать на любой компьютерной архитектуре. [1, 2]

Android Studio – интегрированная среда разработки софта для устройства на платформе Android: смартфоны, планшеты, часы и много другое, которая представляет не только удобное пространство для разработчика, но и которая обеспечивает быстрый запуск рабочего процесса. [27]

Android Studio используется для создания различных типов приложений:

- приложения и игры, которые выполняются не только на платформе Android, но и на Windows;
- веб-службы на основе ASP.NET, JQuery, AngularJS и других популярных платформ;
- приложения для работы с виртуальной реальностью.

Android Studio обладает умным редактором программного кода, который обеспечивает анализ кода и специальные подсказки, позволяющие завершить определенные части программного кода. Это помогает составить структурированный код, тем самым наиболее увеличивая продуктивность написания программного продукта.

Серверная часть написана на языке PHP, встроенном в пакет Denwer, который позволяет обрабатывать GET и POST запросы с Android-клиента. Также, язык позволяет отправлять запросы к базе данных MySQL при помощи языка SQL. Язык PHP работает с данными в JSON-формате. Позволяет запаковывать, отправлять и распаковывать их.

Так как предоставленная среда создана с целью разработки приложений для устройств на платформе Android, в ней имеются эмуляторы для различных типов устройств и несколько режимов работы, такие как debug, test и run.

Среда сама адаптируется под изменения в программе и сразу запускает, без нужды перезапускать или переустанавливать приложение.

3.3 Разработка мобильного приложения

3.3.1 Реализация мобильного приложения

Программа Android Studio дает возможность выбора мобильного устройства, для которого будет разрабатываться мобильное приложение. В Android Studio возможно создавать мобильные приложения для многих устройств, например, смартфон, планшет, умные часы, или Android TV. [4, 5, 6, 7]

На рисунке ниже представлено окно выбора шаблона. Данные шаблоны задают структуру и поведение приложения для определенной задачи. Например, при создании шаблона Empty Activity, создаётся пустой шаблон – пустое окно. Есть также, шаблон с кнопкой, шаблон с вкладками для выбора, а также, шаблон со встроенным сервисом Google Maps. При помощи таких шаблонов можно быстро перейти непосредственно к разработке приложения, не тратя время на добавление кнопок и других интерактивных элементов.

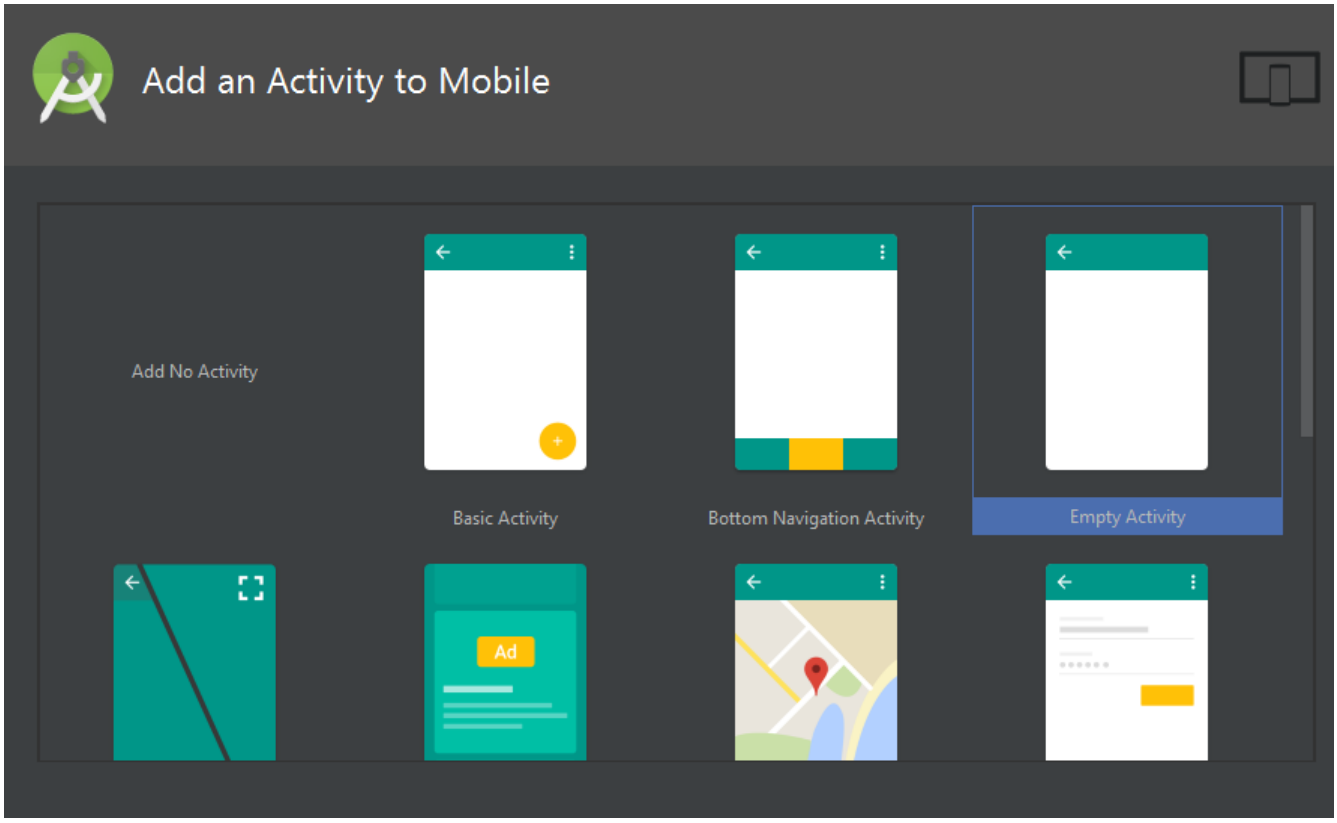

Рисунок 3.1 – Окно выбор шаблона

Задается имя шаблона и после этого, проект автоматически собирается благодаря Gradle. Garadle – это система автоматической сборки. Она выполняет

все необходимые действия: сборка проектов, компиляция кода, выполнение тестирования и многое другое.

На рисунке 3.2 представлено первоначальное окно после загрузки и сборки проекта. Слева отображена полная структура проекта. Весь написанный код будет располагаться в папке «app». В файле «build.gradle» содержится информация, используемая при построении проекта.

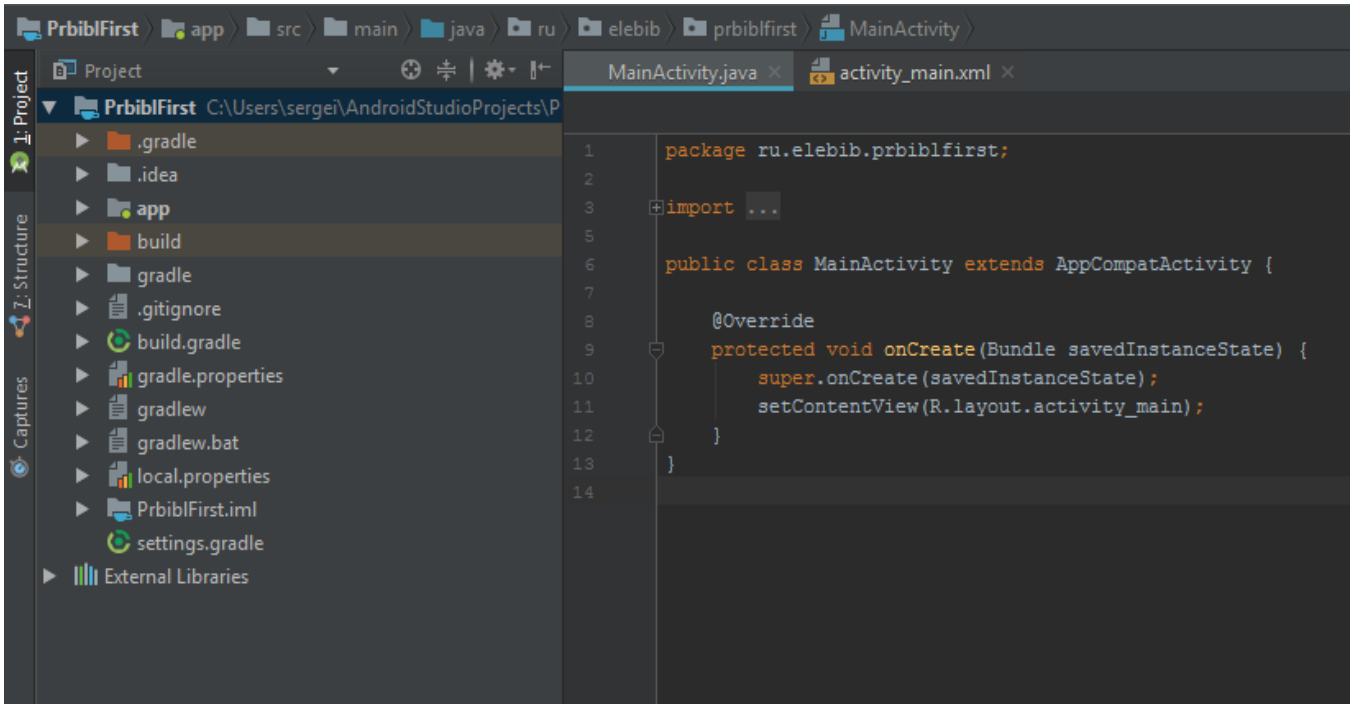

Рисунок 3.2 – Первоначальное окно проекта

Разработанное мобильное приложение позволяет пользователям видеть список имеющихся в базе книг, читать их и осуществлять поиск нужной книги по автору, названию или ключевым словам. При выборе необходимой пользователю книги, открывается новое окно, в котором отображается полный текст книги, имя автора и название самой книги.

Добавлять книги в базу может только администратор данной электронной библиотеки. Для этого на серверной части предусмотрена специальная СУБД для работы с MySQL. Также, добавление книг возможно посредством SQL-запросов из командной строки сервера, где необходимо ввести информацию о книге: Имя автора, название книги и её содержание. После чего, книга будет автоматически внесена в базу и станет доступна для полнотекстового поиска.

Для того чтобы книга появилась в самом приложении, необходимо будет обновить страницу, или перезапустить приложение.

Ниже представлен фрагмент кода, с помощью которого происходит вывод списка авторов на экран мобильного приложения.

```
@Override
protected String doInBackground(String... params) {
    HttpClient httpclient = new DefaultHttpClient();
    HttpGet httpget;
    try {
        httpget = new HttpGet( uri: server + "?req=getAuthors");//Запрос к серверу
        HttpResponse response = httpclient.execute(httpget); //Исполнение запроса к серверу

        if (response.getStatusLine().getStatusCode() == 200) {
            HttpEntity entity = response.getEntity();
            answerHTTP = EntityUtils.toString(entity);
        }
    }
    catch (ClientProtocolException e) {
    }
    catch (IOException e) {
    }
    catch(RuntimeException e){
        System.err.println(e.getMessage());
    }
    return null;
}

public void initList(){

    try {
        JSONObject jo=new JSONObject(answerHTTP);

        JSONArray jsonNames=jo.getJSONArray( name: "names");//Парсинг JSON полученного с сервера
        String[] authors = null;

        for(int i=0;i<jsonNames.length();i++){
            defaultBooks.add(jsonNames.getString(i));//Добавление в List defaultBooks
            books.add(jsonNames.getString(i));//Добавление в визуальный список на экране
            adapter.notifyDataSetChanged();//Адаптер обновляет список, выводимый на экран
        }

    } catch (JSONException e) {
        e.printStackTrace();
    }
}
```

Рисунок 3.3 – Вывод списка авторов на экран

Изначально, клиент посылает запрос серверу с помощью функции «execute()». Запрос с параметром – «getAuthors», означает, что в ответе сервер

пришлет список авторов в формате JSON. Затем, после получения ответа, JSON-массив просматривается и поэлементно заносится в список «books», который в свою очередь, заполняет экран фамилиями авторов.

Далее представлен алгоритм проверки на вхождения, то есть имеется ли слово в тексте, поиск по названию книги или по имени автора, эти алгоритмы аналогичны. Также, полнотекстовый поиск основан на подобном алгоритме, изображенном на рисунке 3.4.

```
editText.addTextChangedListener(new TextWatcher() {
    @Override
    public void beforeTextChanged(CharSequence s, int start, int count, int after) {

    }

    @Override
    public void onTextChanged(CharSequence s, int start, int before, int count) {
        if(s.toString().equals("")){
            // reset listview
            books.clear();
            books.addAll(defaultBooks);
        } else {
            // perform search
            searchItem(s.toString());
        }
        adapter.notifyDataSetChanged();
    }

    public void searchItem(String textToSearch){

        ArrayList<String> tempBooks=new ArrayList<>();

        //adapter.clear();
        for(int i=0;i<defaultBooks.size();i++){
            if(defaultBooks.get(i).startsWith(textToSearch)){//выполняется поиск из списка книг
                tempBooks.add(defaultBooks.get(i));//найденные книжки
            }
        }
        books.clear();//визуальная часть
        books.addAll(tempBooks);
        //adapter.notifyDataSetChanged();
    }

    @Override
    public void afterTextChanged(Editable s) {

    }
```

Рисунок 3.4 – Алгоритм поиска по введенному тексту

Ниже представлен фрагмент кода алгоритма подключения к серверу и отправка GET-запроса на сервер. Для определения, какая именно информация нужна в ответе, используются параметры, такие как getContent, getAllBooks,

getAuthor – соответственно, данный метод возвращает текст книги, список всех книг и список книг по имени автора.

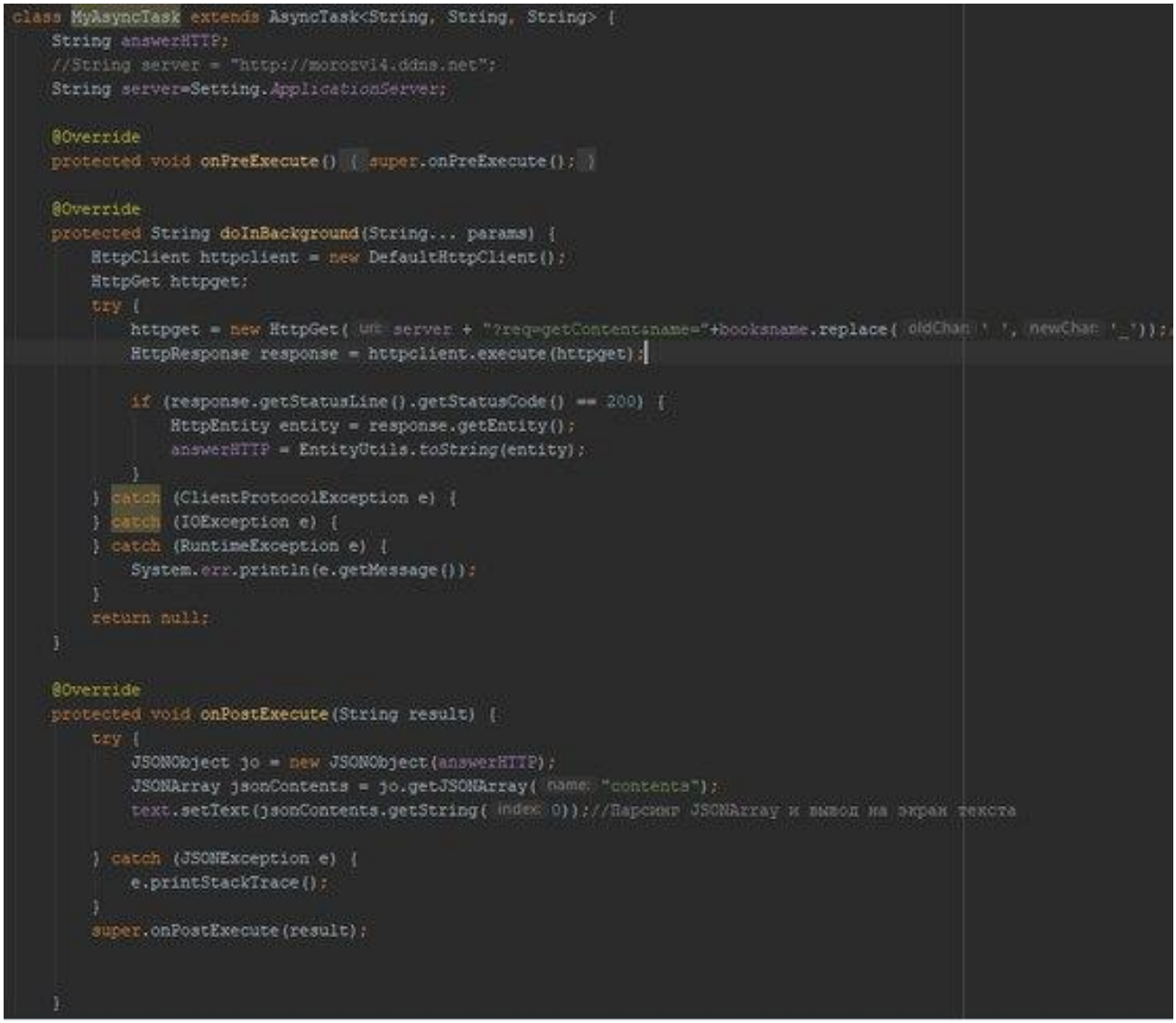

Рисунок 3.5 – Класс чтения книги

На рисунке 3.6 показана реализация класса «Book», класс для объектов-книг, в котором содержатся поля «bookname» – название книги, «bookauthor» – имя автора и «bookcontent» – текст книги. Также, имеются методы get, которые возвращают значения этих полей.

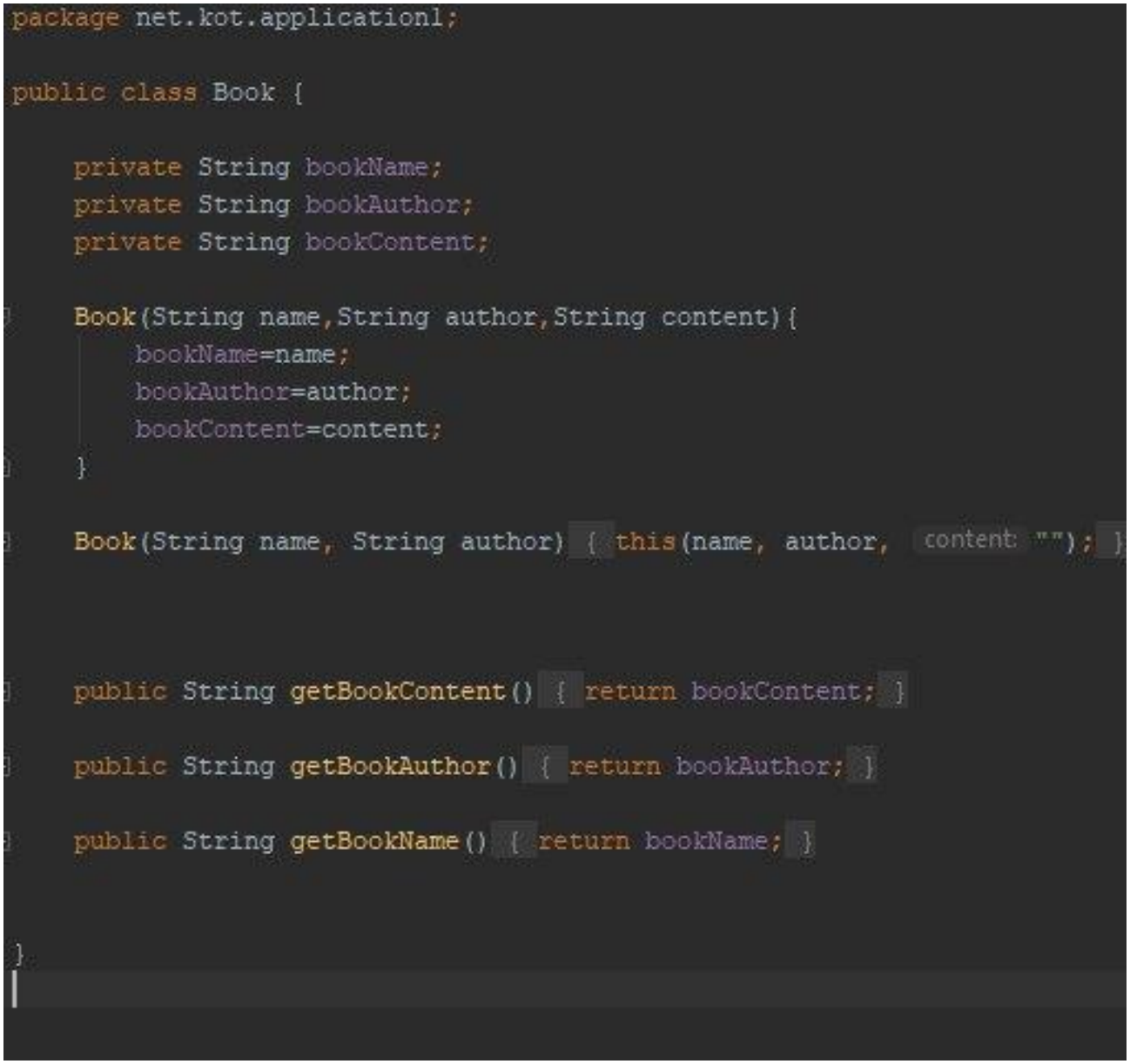

Рисунок 3.6 – Класс «Book»

Таким образом, для каждой книги, взятой из базы данных, создается отдельный объект, что позволяет потом удобно производить полнотекстовый поиск, поиск по названию и имени автора.

3.3.2 Описание работы мобильного приложения

При открытии приложения на главной странице пользователю сразу виден список авторов, книги которых имеются в базе. Пользователь может прочесть книгу, просто выбрав сначала автора, а затем его произведение. Главная страница приложения со списком авторов показана на рисунке 3.7. На рисунке 3.8 показан выбор автора и выводятся произведения выбранного автора.

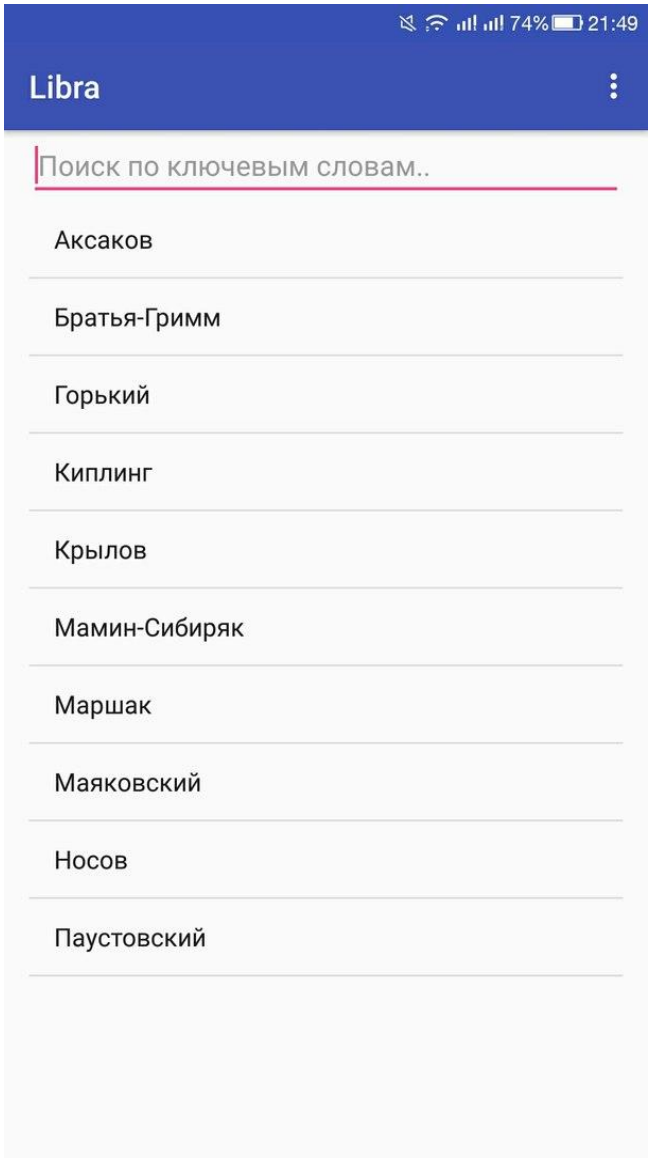

Рисунок 3.7 – Главная страница мобильного приложения

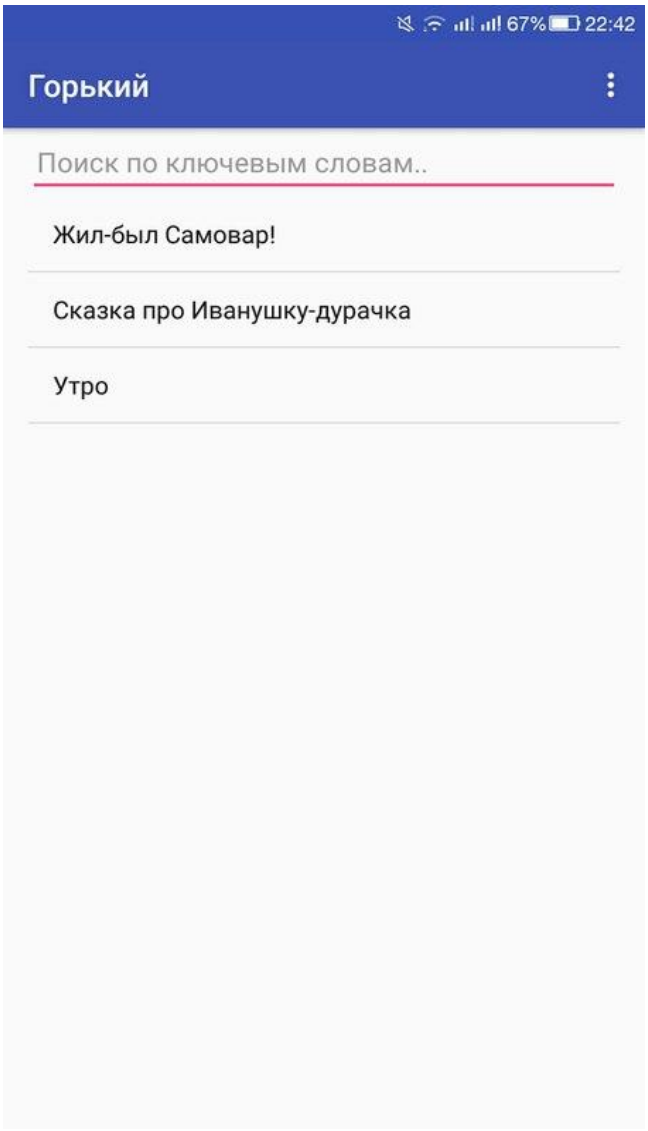

Рисунок 3.8 – Выбор автора

На рисунке 3.9 представлен выбор произведения.

Также, в функционал данного мобильного приложения для электронной библиотеки включена возможность полнотекстового поиска. Форма полнотекстового поиска представлена на рисунке 3.10.

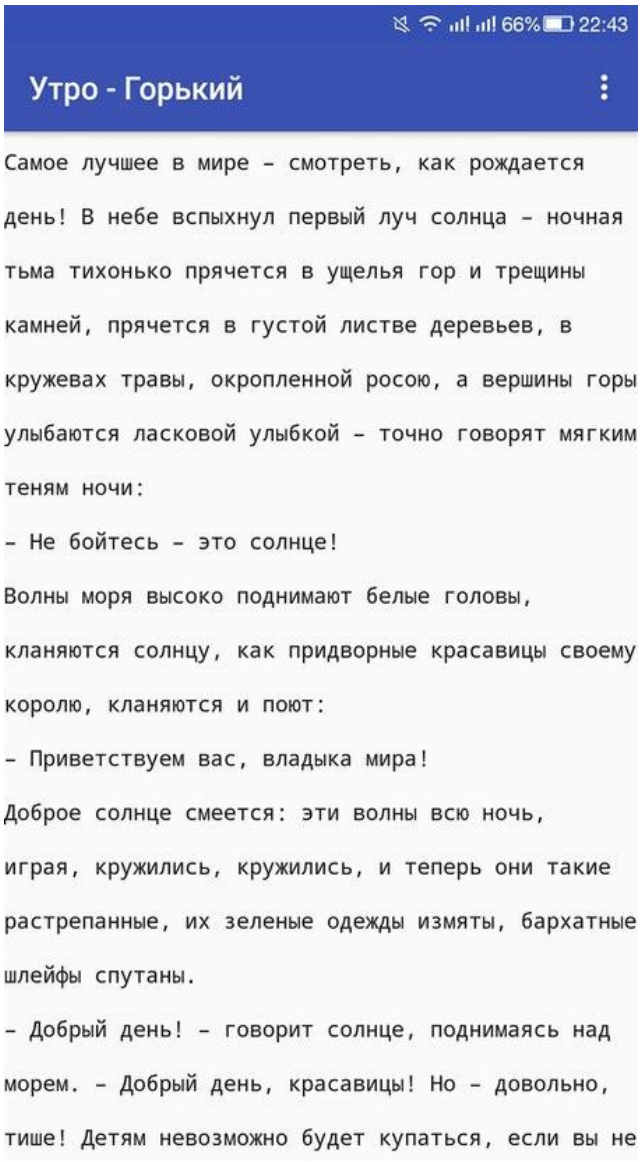

Рисунок 3.9 – Выбор произведения

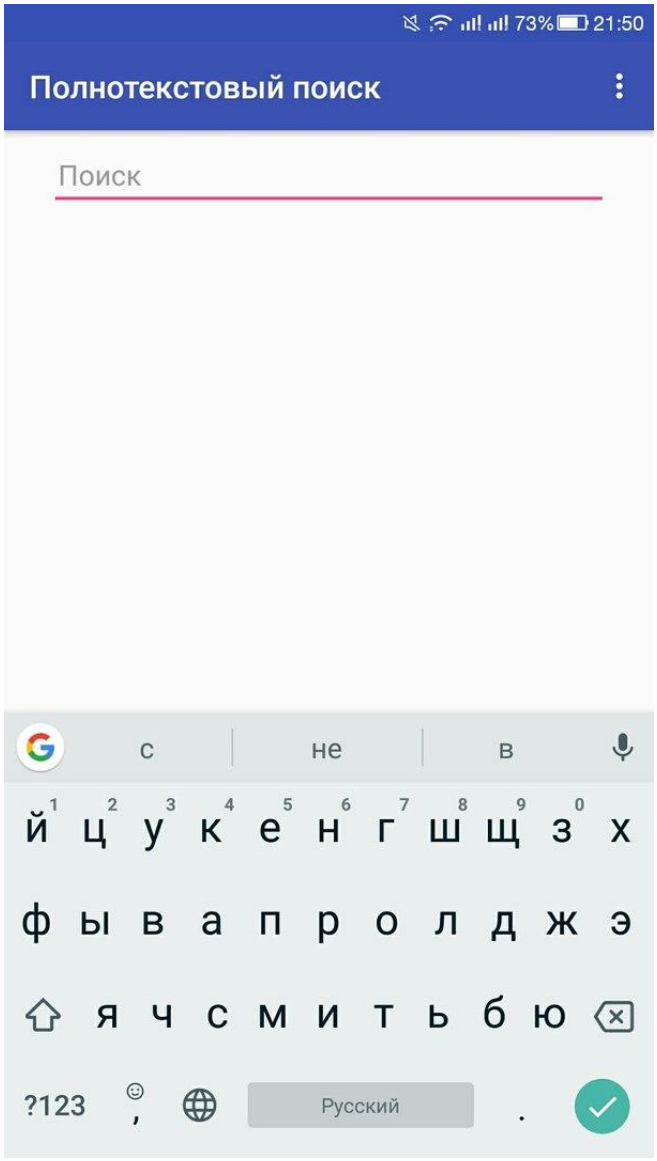

Рисунок 3.10 – Полнотекстовый поиск

При вводе любого ключевого слова или названия определенной книги, приложение откроет пользователю список книг, в которых содержится запрашиваемое слово. Это показано на рисунке ниже.

При полнотекстовом поиске, происходит проход по списку объектов класса «Book» и приложение ищет в поле «content» каждого объекта вхождения. В случае если вхождение найдено, приложение выводит на экран искомую книгу.

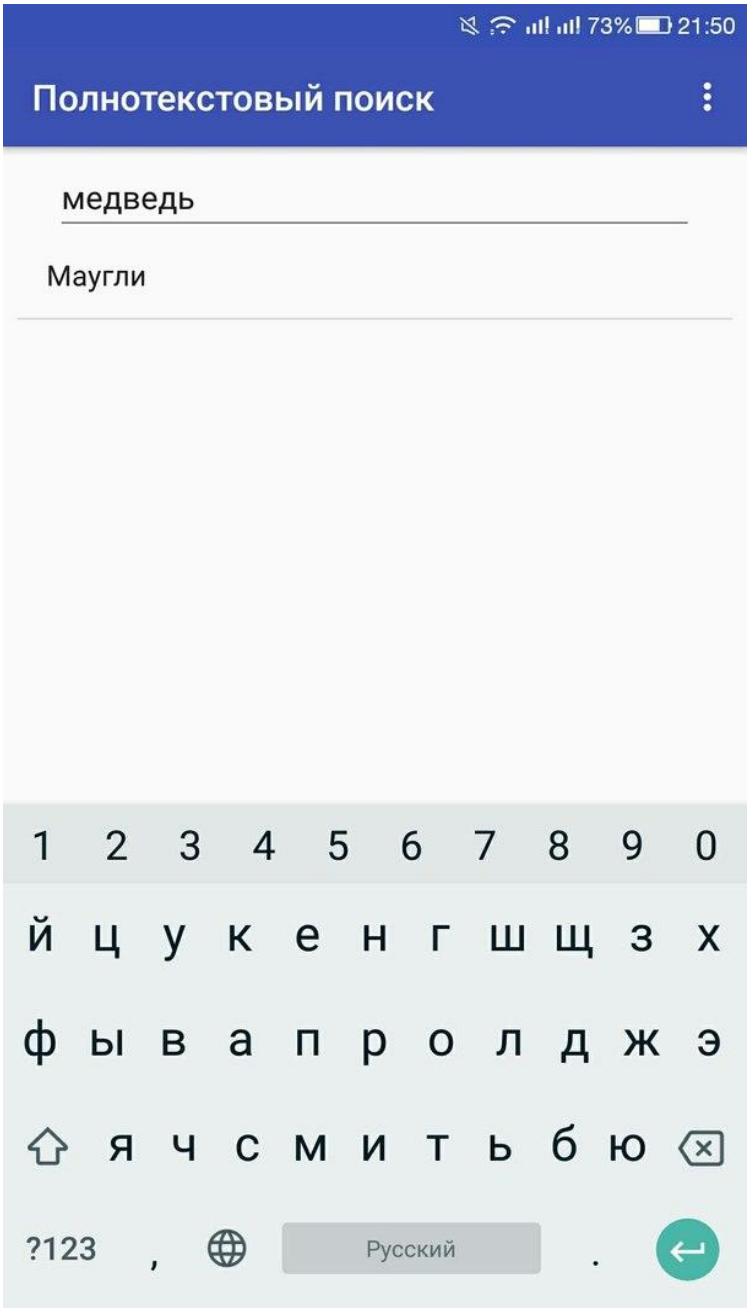

Рисунок 3.11 – Полнотекстовый поиск по ключевому слову

Таким образом, просмотрев список книг, их текст, форму полнотекстового поиска, можно сделать вывод о том, что данное мобильное приложение соответствует поставленным задачам и требованиям.

3.3.3 Тестирование мобильного приложения

Данное приложение было протестировано с разным количеством книг, загруженных на сервер. Данные о времени загрузки списка книг и списка авторов в приложение предоставлены в таблице 3.1.

Таблица 3.1 – Зависимость времени загрузки от количества книг в базе.

	Платформа тестирования		
Количество книг	Встроенный эмулятор	Le Eco CoolPad	ZTE Blade A510
50	4,3 сек.	2,3 сек.	3,1 сек.
100	4,5 сек.	2,4 сек.	3,2 сек.
250	4,6 сек.	2,4 сек.	3,2 сек.

Таким образом, согласно таблице, видно, что основная нагрузка идет на серверную часть, так как время обработки запросов при разном количестве книг почти не отличается. Также наблюдается небольшая зависимость от платформы тестирования: чем мощнее устройство, тем быстрее происходит обработка запроса от сервера.

Выводы по третьей главе.

В данной главе были выбраны средства проектирования мобильного приложения: язык программирования Java, интегрированная среда разработки Android Studio. Для серверной части использовался язык PHP, база данных MySQL и сервер, встроенный в пакет Denwer. Также, было реализовано, разработано и протестировано мобильное приложение электронной библиотеки.

ЗАКЛЮЧЕНИЕ

Целью данной выпускной квалификационной работы являлась разработка приложения Android для электронной библиотеки. В приложении реализованы функции поиска текста в базе данных, вывода текста на экран, вывода списка книг.

В процессе выполнения выпускной квалификационной работы был исследован теоретический материал по технологии разработки мобильных приложений для операционной системы Android. Также были определены требования к мобильному приложению и были выделены задачи, которые необходимо было реализовать.

Проводился анализ имеющихся средств разработки мобильного приложения. Так же, было представлено сравнение операционных систем, таких как Android, iOS и Windows Phone, и представлена архитектура ОС Android.

При создании электронной библиотеки были проанализированы уже существующие мобильные приложения для электронных библиотек. Были выбраны специальные программы и средства для реализации. Мобильное приложение реализовано на языке Java. В качестве базы данных использовалась MySQL.

Разработанное мобильное приложение позволяет пользователю делать запрос на поиск книги по автору и названию книги, а также по ключевым словам Достоинствами данного приложения является простота в использовании и приятный, понятный пользователю интерфейс.

В процессе выполнения выпускной квалификационной работы было разработано мобильное приложение для электронной библиотеки с полнотекстовым поиском, которое соответствует требованиям по разработке мобильного приложения.

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

Научная и методическая литература

1. Блох Д. Java. Эффективное программирование: Лори, 2014. – 440с.

2. Васильев, А.Н. Самоучитель Java с примерами и программами / А.Н. Васильев. – СПб.: Наука и техника, 2016. – 368с.

3. Вендров, А.М. Практикум по проектированию программного обеспечения экономических информационных систем. Учеб. Пособие. – М.: Финансы и статистика, 2014.

4. Голощапов, А. Google Android: программирование для мобильных устройств / А. Голощапов. – СПб.: БХВ-Петербург, 2010. – 448 с.

5. Гриффитс Д., Гриффитс Д. Head First. Программирование для Android. — СПб.: Питер, 2016. — 704 с.

6. Дейтел П., Дейтел Х., Дейтел Э., Моргано М. Android для программистов: создаём приложения. — СПб.: Питер, 2013. — 560 с.

7. МакГрат, М. Программирование на Java для начинающих [Текст] / М. МакГрат. – М.: Эксмо. – 2016. – 192 с.

8. Медникс З., Дорнин Л., Мик Б., Накамура М. Программирование под Android. – СПб.: Питер, 2013. — 560 с.

9. Роджерс, Р. Android. Разработка приложений, / Р. Роджерс, Д. Ломбардо. – М.: ЭКОМ Паблишерз, 2010. – 400 с.

10. Таненбаум, Э. Современные операционные системы / Э. Таненбаум. – СПб.: Питер, 2013. - 1120 с.

11. Фелкер Д. Android: Разработка для чайников / Д.Фелкер – М.: ООО “И.Д. Вильямс”, 2012. — 336 с.

12. Филлипс Б., Стюарт К., Марсикано К. Android. Программирование для профессионалов. 3-е изд. — СПб.: Питер, 2017. — 688 с.

13. Хорстманн К.С. Java. Библиотека профессионала, том 1 / К.С. Хорстманн. – М .: ООО "И.Д. Вильямс", 2016. — 864 с.

14. Чистов, Д. В. Проектирование информационных систем. Учебник и практикум / Д. В. Чистов, П. П. Мельников, А. В. Золотарюк, Н. Б. Ничепорук. -

М.: Юрайт, 2016. - 260 с.

Электронные ресурсы

15. 10 приложений-«читалок» с доступом к бесплатным книгам [Электронный ресурс]. – Режим доступа: https://informburo.kz/stati/10-prilozheniy-chitalok-s-dostupom-k-besplatnym-knigam.html (дата обращения 13.04.2018)

16. Архитектура информационных систем. Локальная, клиент-сервер, двух и трехуровневая архитектура [Электронный ресурс]. – Режим доступа: https://studopedia.su/18_23238_arhitektura-informatsionnih-sistem-lokalnaya-klient-server-dvuh-i-trehurovnevaya-arhitektura.html (дата обращения 20.04.2018)

17. Джентльменский Набор Веб-Разработчика [Электронный ресурс]. – Режим доступа: http://denwer.ru (24.04.2018)

18. История появления эмблемы (логотипа) Android [Электронный ресурс]. – Режим доступа: http://superandroid.ru/page/history-logo-android

19. Обзор мобильных платформ [Электронный ресурс]. – Режим доступа: https://studwood.ru/1936921/informatika/obzor_suschestvuyuschih_mobilnyh_platform. (дата обращения 1.04.2018)

20. Требования к системе: классификация FURPS+ [Электронный ресурс]. – Режим доступа: https://sysana.wordpress.com/2010/09/16/furps/. (дата обращения 19.04.2018)

21. Электронные библиотеки [Электронный ресурс]. – Режим доступа: https://studfiles.net/preview/5438745/page:10/. (дата обращения 20.03.2018)

Литература на иностранном языке

22. Alan Dennis, Barbara Haley Wixom, David Tegarden: Systems Analysis and Design with UML - 4th Edition, Wiley, 2012.

23. Alan Mark Davis. Just Enough Requirements Management: Where Software Development Meets Marketing. — Dorset House, 2015.

24. Farrell J. Java Programming. — Course Technology, 2015.

25. Hahn J. Mobile augmented reality applications for library services / Jim Hahn / New Library World. –2012. –Vol. 113, No. 9/10. – P.429 – 438.

26. Michael J. Hernandez: Database Design for Mere Mortals: A Hands-On Guide to Relational Database Design - 3rd Edition, Addison-Wesley Professional, 2013.

27. Smyth N. Android Studio Development Essentials / Neil Smyth / Android 6 Edition – Payload Media, 2015.

ПРИЛОЖЕНИЕ А

Листинг алгоритма вывода списка авторов на экран

```
import android.annotation.SuppressLint;
import android.content.Intent;
import android.os.AsyncTask;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.text.Editable;
import android.text.TextWatcher;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ListView;
import android.widget.TextView;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.HttpClient;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.util.EntityUtils;
import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

import java.io.IOException;
import java.util.ArrayList;

public class AuthorActivity extends AppCompatActivity {

    private TextView textView;
    private EditText editText;
    private EditText editText2;
    private EditText editText3;
    private Button button,button2;

    private ArrayAdapter<String> adapter;
    private ArrayList<String> defaultBooks;
    private ArrayList<String> books;

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.menu_author, menu);
        return true;
    }
```

```
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        int id=item.getItemId();
        Intent switchIntent=null;

        switch (id){
            case R.id.action_settings:
                switchIntent = new Intent(AuthorActivity.this, SettingsActivity.class);
                break;
            case R.id.action_about:
                switchIntent = new Intent(AuthorActivity.this, AboutActivity.class);
                break;
            case R.id.action_fulltext:
                switchIntent = new Intent(AuthorActivity.this, FulltextActivity.class);
                break;
        }
        startActivity(switchIntent);
        return super.onOptionsItemSelected(item);
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_author);

        ListView listView=(ListView)findViewById(R.id.listView);
        editText = (EditText)findViewById(R.id.editText);

        defaultBooks=new ArrayList<>();
        books=new ArrayList<>();
        adapter=new ArrayAdapter<>(this,android.R.layout.simple_list_item_1,books);
        listView.setAdapter(adapter);

        listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
            @SuppressLint("ResourceType")
            @Override
            public void onItemClick(AdapterView<?> parent,View itemClicked,int position,long id) {
                Intent intent = new Intent(AuthorActivity.this, BooksActivity.class);
                //Toast.makeText(getApplicationContext(),"ID Clicked:
"+id,Toast.LENGTH_SHORT).show();
                intent.putExtra("name",((TextView) itemClicked).getText());
                startActivity(intent);
            }
        });

        editText.addTextChangedListener(new TextWatcher() {
            @Override
            public void beforeTextChanged(CharSequence s, int start, int count, int after) {

            }

            @Override
```

```
        public void onTextChanged(CharSequence s, int start, int before, int count) {
            if(s.toString().equals("")){
                // reset listview
                books.clear();
                books.addAll(defaultBooks);
            } else {
                // perform search
                searchItem(s.toString());
            }
            adapter.notifyDataSetChanged();
        }

        public void searchItem(String textToSearch){

            ArrayList<String> tempBooks=new ArrayList<>();

            //adapter.clear();
            for(int i=0;i<defaultBooks.size();i++){
                if(defaultBooks.get(i).startsWith(textToSearch)){
                    tempBooks.add(defaultBooks.get(i));//
                }
            }
            books.clear();
            books.addAll(tempBooks);
            //adapter.notifyDataSetChanged();
        }

        @Override
        public void afterTextChanged(Editable s) {

        }
    });

    new MyAsyncTask().execute();
}

class MyAsyncTask extends AsyncTask<String, String, String> {
    String a,answerHTTP;
    String server=Setting.ApplicationServer;

    @Override
    protected void onPreExecute() {
        super.onPreExecute();
    }

    @Override
    protected String doInBackground(String... params) {
        HttpClient httpclient = new DefaultHttpClient();
        HttpGet httpget;
        try {
            httpget = new HttpGet(server + "?req=getAuthors");
            HttpResponse response = httpclient.execute(httpget);
```

```
            if (response.getStatusLine().getStatusCode() == 200) {
                HttpEntity entity = response.getEntity();
                answerHTTP = EntityUtils.toString(entity);
            }
        }
        catch (ClientProtocolException e) {
        }
        catch (IOException e) {
        }
        catch(RuntimeException e){
            System.err.println(e.getMessage());
        }
        return null;
    }

    public void initList(){

        try {
            JSONObject jo=new JSONObject(answerHTTP);

            JSONArray jsonNames=jo.getJSONArray("names");
            String[] authors = null;

            for(int i=0;i<jsonNames.length();i++){
                defaultBooks.add(jsonNames.getString(i));
                books.add(jsonNames.getString(i));
                adapter.notifyDataSetChanged();
            }

        } catch (JSONException e) {
            e.printStackTrace();
        }
    }

    @Override
    protected void onPostExecute(String result) {
        initList();

        super.onPostExecute(result);

    }
  }
}
```

ПРИЛОЖЕНИЕ Б

Листинг алгоритма вывода списка книг на экран

```
public class BooksActivity extends AuthorActivity {

    private ArrayAdapter<String> adapter;
    private ArrayList<String> books;

    private ArrayList<String> defaultBooks;

    private ArrayList<Book> AppBooks=new ArrayList<>();

    private String authorname;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_books);

        authorname = getIntent().getStringExtra("name");
        setTitle(authorname);

        ListView listView = (ListView) findViewById(R.id.listView);
        EditText editText = (EditText) findViewById(R.id.editText);

        books = new ArrayList<>();
        adapter = new ArrayAdapter<>(this, android.R.layout.simple_list_item_1, books);
        defaultBooks=new ArrayList<>();


        listView.setAdapter(adapter);

        listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
            @SuppressLint("ResourceType")
            @Override
            public void onItemClick(AdapterView<?> parent, View itemClicked, int position, long id) {
                Intent intent = new Intent(BooksActivity.this, ReadActivity.class);

                String localtext=((TextView) itemClicked).getText().toString();

                for(int i=0;i<AppBooks.size();i++){
                    if(localtext.equals(AppBooks.get(i).getBookName())){
                        intent.putExtra("name",localtext);
                        intent.putExtra("author",AppBooks.get(i).getBookAuthor());
                        //intent.putExtra("content",AppBooks.get(i).getBookContent());
                    }
                }
                startActivity(intent);
            }
        });
```

```
    editText.addTextChangedListener(new TextWatcher() {
        @Override
        public void beforeTextChanged(CharSequence s, int start, int count, int after) {

        }

        @Override
        public void onTextChanged(CharSequence s, int start, int before, int count) {
            if(s.toString().equals("")){
                // reset listview
                books.clear();
                books.addAll(defaultBooks);
            } else {
                // perform search
                searchItem(s.toString());
            }
            adapter.notifyDataSetChanged();
        }

        public void searchItem(String textToSearch){

            ArrayList<String> tempBooks=new ArrayList<>();

            //adapter.clear();
            for(int i=0;i<defaultBooks.size();i++){
                if(defaultBooks.get(i).startsWith(textToSearch)){
                    tempBooks.add(defaultBooks.get(i));
                }
            }
            books.clear();
            books.addAll(tempBooks);
            //adapter.notifyDataSetChanged();
        }

        @Override
        public void afterTextChanged(Editable s) {

        }
    });

    new MyAsyncTask().execute();
}

public String getAuthorName() {
    return authorname;
}

class MyAsyncTask extends AsyncTask<String, String, String> {
    String answerHTTP;
    String server=Setting.ApplicationServer;
```

```
        @Override
        protected void onPreExecute() {
            super.onPreExecute();
        }

        @Override
        protected String doInBackground(String... params) {
          HttpClient httpclient = new DefaultHttpClient();
            HttpGet httpget;
            try {
                httpget = new HttpGet(server + "?req=getnew&author=" + authorname);
                HttpResponse response = httpclient.execute(httpget);

                if (response.getStatusLine().getStatusCode() == 200) {
                    HttpEntity entity = response.getEntity();
                    answerHTTP = EntityUtils.toString(entity);
                }
            } catch (ClientProtocolException e) {
            } catch (IOException e) {
            } catch (RuntimeException e) {
                System.err.println(e.getMessage());
            }
            return null;
        }

        @Override
        protected void onPostExecute(String result) {
            try {
                JSONObject jo = new JSONObject(answerHTTP);

                JSONArray jsonNames = jo.getJSONArray("names");

                for (int i = 0; i < jsonNames.length(); i++) {
                    AppBooks.add(new Book(jsonNames.getString(i),authorname));

                    defaultBooks.add(jsonNames.getString(i));
                    books.add(defaultBooks.get(i));
                    adapter.notifyDataSetChanged();
                }
            } catch (JSONException e) {
                e.printStackTrace();
            }
            super.onPostExecute(result);

        }
    }
}
```

ПРИЛОЖЕНИЕ В

Листинг алгоритма полнотекстового поиска

```
public class FulltextActivity extends AuthorActivity {

    private ArrayAdapter<String> adapter;
    private ArrayList<String> books;
    private ArrayList<String> defaultBooks;
    private ArrayList<Book> AppBooks=new ArrayList<>();
    private String searchword;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_fulltext);

        setTitle("Полнотекстовый поиск");

        ListView listView = (ListView) findViewById(R.id.listView);
        EditText editText = (EditText) findViewById(R.id.editText3);

        books = new ArrayList<>();
        adapter = new ArrayAdapter<>(this, android.R.layout.simple_list_item_1, books);
        defaultBooks=new ArrayList<>();

        listView.setAdapter(adapter);

        listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
            @SuppressLint("ResourceType")
            @Override
            public void onItemClick(AdapterView<?> parent, View itemClicked, int position, long id) {
                Intent intent = new Intent(FulltextActivity.this, ReadActivity.class);

                String localtext=((TextView) itemClicked).getText().toString();

                for(int i=0;i<AppBooks.size();i++){
                    if(localtext.equals(AppBooks.get(i).getBookName())){
                        intent.putExtra("name",localtext);
                        intent.putExtra("author",AppBooks.get(i).getBookAuthor());
                        intent.putExtra("content",AppBooks.get(i).getBookContent());
                        intent.putExtra("searchword",searchword);
                    }
                }
                startActivity(intent);
            }
        });

        editText.addTextChangedListener(new TextWatcher() {
            @Override
            public void beforeTextChanged(CharSequence s, int start, int count, int after) {
```

```
        }

        @Override
        public void onTextChanged(CharSequence s, int start, int before, int count) {
            if(s.toString().equals("")){
                // reset listview
                books.clear();
                books.addAll(defaultBooks);
            } else {
                // perform search
                searchItem(s.toString());
            }
            adapter.notifyDataSetChanged();
        }

        public void searchItem(String textToSearch){
            ArrayList<String> tempBooks=new ArrayList<>();
            for(int i=0;i<defaultBooks.size();i++){
                if(AppBooks.get(i).getBookContent().contains(textToSearch)) {
                    tempBooks.add(AppBooks.get(i).getBookName());
                    searchword = textToSearch;
                }
            }
            books.clear();
            books.addAll(tempBooks);
            //adapter.notifyDataSetChanged();
        }

        @Override
        public void afterTextChanged(Editable s) {
        }
    });

    new MyAsyncTask().execute();
}

class MyAsyncTask extends AsyncTask<String, String, String> {
    String answerHTTP;
    String server=Setting.ApplicationServer;

    @Override
    protected void onPreExecute() {
        super.onPreExecute();
    }

    @Override
    protected String doInBackground(String... params) {
        HttpClient httpclient = new DefaultHttpClient();
        HttpGet httpget;
        try {
            httpget = new HttpGet(server + "?req=getAllBooks");
            HttpResponse response = httpclient.execute(httpget);
```

```
            if (response.getStatusLine().getStatusCode() == 200) {
                HttpEntity entity = response.getEntity();
                answerHTTP = EntityUtils.toString(entity);
            }
        } catch (ClientProtocolException e) {
        } catch (IOException e) {
        } catch (RuntimeException e) {
            System.err.println(e.getMessage());
        }
        return null;
    }

    @Override
    protected void onPostExecute(String result) {
        try {
            JSONObject jo = new JSONObject(answerHTTP);
            JSONArray jsonNames = jo.getJSONArray("names");
            JSONArray jsonAuthors = jo.getJSONArray("authors");
            JSONArray jsonContents = jo.getJSONArray("contents");

            for (int i = 0; i < jsonNames.length(); i++) {
                AppBooks.add(new
Book(jsonNames.getString(i),jsonAuthors.getString(i),jsonContents.getString(i)));

                defaultBooks.add(AppBooks.get(i).getBookName());
                books.add(AppBooks.get(i).getBookName());
                adapter.notifyDataSetChanged();
            }
        } catch (JSONException e) {
            e.printStackTrace();
        }
        super.onPostExecute(result);

    }
  }
}
```

ПРИЛОЖЕНИЕ Г

Листинг алгоритма вывода содержимого книги на экран

```
public class ReadActivity extends AuthorActivity {

    private String booksname;
    private String booksauthor;
    private TextView text;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_read);

        booksname = getIntent().getStringExtra("name");
        booksauthor = getIntent().getStringExtra("author");

        text = (TextView) findViewById(R.id.textview);

        setTitle(booksname+" - "+booksauthor);

        //text.setText(bookscontent);
        new MyAsyncTask().execute();
    }

    class MyAsyncTask extends AsyncTask<String, String, String> {
        String answerHTTP;
        String server=Setting.ApplicationServer;

        @Override
        protected void onPreExecute() {
            super.onPreExecute();
        }

        @Override
        protected String doInBackground(String... params) {
            HttpClient httpclient = new DefaultHttpClient();
            HttpGet httpget;
            try {
                httpget = new HttpGet(server + "?req=getContent&name="+booksname.replace(' ','_'));
                HttpResponse response = httpclient.execute(httpget);

                if (response.getStatusLine().getStatusCode() == 200) {
                    HttpEntity entity = response.getEntity();
                    answerHTTP = EntityUtils.toString(entity);
                }
            } catch (ClientProtocolException e) {
            } catch (IOException e) {
            } catch (RuntimeException e) {
                System.err.println(e.getMessage());
            }
```

```
            return null;
        }

        @Override
        protected void onPostExecute(String result) {
            try {
                JSONObject jo = new JSONObject(answerHTTP);
                JSONArray jsonContents = jo.getJSONArray("contents");
                text.setText(jsonContents.getString(0));

            } catch (JSONException e) {
                e.printStackTrace();
            }
            super.onPostExecute(result);


        }
    }

}
```