Министерство науки и высшего образования РФ
Федеральное государственное автономное
образовательное учреждениевысшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Хакасский технический институт – филиал ФГАОУ ВО
«Сибирский федеральный университет»

Кафедра прикладной информатики, естественно-научных
и гуманитарных дисциплин

УТВЕРЖДАЮ
Заведующий кафедрой
___ О. В. Папина
подпись
«____» _____ 2023 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.03 Прикладная информатика

Разработка мобильного приложения «Тренажер ЕГЭ по математике»

Руководитель _____ доцент, канд. физ.–мат. наук М. А. Буреева
подпись, дата

Выпускник _____ Я. Р. Драганов
подпись, дата

Консультанты
по разделам:

Экономический _____ М. А. Буреева
подпись, дата

Нормоконтролер _____ А. Н. Кадычегова
подпись, дата

Абакан 2023

Министерство науки и высшего образования РФ
Федеральное государственное автономное
образовательное учреждениевысшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Хакасский технический институт – филиал ФГАОУ ВО
«Сибирский федеральный университет»

Кафедра прикладной информатики, естественно-научных
и гуманитарных дисциплин

УТВЕРЖДАЮ
Заведующий кафедрой
___ О. В. Папина
подпись
«____» _____ 2023 г.

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
в форме бакалаврской работы

Абакан 2023

Студенту Драганову Ярославу Руслановичу

Группа ХБ 19-02

Направление 09.03.03 Прикладная информатика

Тема выпускной квалификационной работы:Разработка мобильного приложения «Тренажер ЕГЭ по математике»

Утверждена приказом по институту № 283 от 11.05.2023 г.

Руководитель ВКР: М. А. Буреева, доцент, канд. физ.–мат. наук, ХТИ – филиал СФУ

Исходные данные для ВКР: заказ ХТИ – филиала СФУ.

Перечень разделов ВКР:

1. Анализ предметной области. Выбор средств проектных решений.
2. Описание разработки игры.
3. Расчёт затрат и оценка экономической эффективности реализации мобильного приложения

Перечень графического материала: нет

Руководитель ВКР _____ М. А. Буреева
подпись

Задание принял к исполнению _____ Я. Р. Драганов
подпись

«11» мая 2023г.

РЕФЕРАТ

Выпускная квалификационная работа по теме: Разработкамобильного приложения «Тренажер ЕГЭ по математике»содержит 69 страниц текстового документа, 35 иллюстраций, 8 таблиц, 5 формул, 8 использованных источников, 3 приложения.

МОБИЛЬНОЕ ПРИЛОЖЕНИЕ, ANDROIDSTUDIO, XML, ANDROIDSDK, JAVA, FIREBASE, DFD, UML, IDEF3, ТАБЛИЦЫ, SQLITE, КАПИТАЛЬНЫЕ ЗАТРАТЫ, СТОИМОСТЬ, ТСО, РИСКИ.

Объект выпускной квалификационной работы – повышение интереса абитуриентов к ХТИ – филиалу СФУ и мотивирование в подготовке к ЕГЭ.

Предмет выпускной квалификационной работы – приложение-тренажер ЕГЭ по математике для мобильных устройств под управлением ОС Android.

Цель выпускной квалификационной работы – создание мобильного приложения-тренажера для операционной системы Android, с целью повышения интереса абитуриентов к ХТИ – филиалу СФУ.

В первом разделе проведен анализ предметной области и охарактеризована организация ХТИ – филиал СФУ. Проведен анализ аналогичных программных продуктов. Проведен анализ программных средств разработки и обоснован выбор ПО «Android Studio». Спроектирована база данных.

Во втором разделе описана реализация мобильного приложения с подключением к сетевой базе данных Firebase и возможностью обновления банка заданий в файловой системе средствами SQLite и программным языком Java, навигацией между окнами.

В третьем разделе проведен расчет затрат на реализацию проекта методикой ТСО. Отдельно произведен расчёт капитальных, эксплуатационных и прямых затрат. Проанализированы риски, выявлены вероятности и уровни влияния рисков соответствия, операционных рисков, реализационных рисков, приведены возможные пути минимизации каждого из рисков.

SUMMARY

The theme of the graduation thesis is «IT-project management “Mobile application-simulator for Unified State Examination in Mathematics” ». It contains 69 pages of, 35 figures, 8 tables, 5 formulae, 8 reference items, 3 appendices.

MOBILE APP, ANDROID STUDIO, XML, ANDROID SDK, JAVA, FIREBASE, DFD, UML, IDEF3, TABLES, SQLITE, CAPEX, COST, TCO, RISKS

The object of the graduation thesis is to increase the interest of applicants for KHTI - a branch of the Siberian Federal University and to motivate them in preparing for the Unified State Examination.

The subject of the graduation thesis is the USE simulator application in Mathematics for mobile devices running the Android OS.

The purpose of the graduation thesis is to create a mobile simulator application for the Android operating system to increase the interest of applicants for KHTI, a branch of the Siberian Federal University.

The first section analyzes the subject area and characterizes the organization of KHTI, a branch of the Siberian Federal University. The analysis of similar software products has been carried out. The analysis of software development tools has been provided and the choice of software “Android Studio” has been verified. The database has been designed.

The second section describes the implementation of a mobile application with a connection to the Firebase network database and the ability to update the job bank in the file system using SQLite and the Java programming language, navigation between windows.

The third section considers the calculation of the costs for the implementation of the project through the TCO method. Moreover, the calculation of capital, operating and direct costs has been given. The risks have been analyzed; the probabilities and levels of influence of compliance risks, operational risks, and implementation risks are identified, possible ways to minimize each of the risks are provided.

English language supervisor _____ N.V. Chezybaeva

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В современном мире мобильные технологии имеют высокий уровень влияния на нашу повседневную жизнь. Они упрощают многие процессы и облегчают нашу работу. В этой связи разработка мобильного приложения для профориентационных мероприятий ХТИ – филиала СФУ может существенно улучшить качество и процесс организации таких мероприятий.

Разработка мобильного приложения для образовательных услуг в России для школьников является перспективной. Реализация такого проекта может сопровождаться рисками, которые необходимо учитывать и минимизировать.

Цель выпускной квалификационной работы – создание мобильного приложения-тренажера для операционной системы Android с целью повышения интереса абитуриентов к ХТИ – филиалу СФУ.

Задачи:

1. Провести анализ предметной области и разработать концепцию IT-проекта.

2. Выполнить анализ аналогичных программных продуктов и обосновать необходимость собственной разработки;

3. Выполнить структурное моделирование для описания бизнес-процесса, подлежащего автоматизации;

4. Обосновать выбор программных средств разработки информационной системы.

5. Разработать мобильное приложение «Тренажер ЕГЭ по математике».

6. Составить план дальнейших работ по реализации проекта.

В первом разделе выполнен анализ предметной области и охарактеризована деятельность ХТИ – филиал СФУ в организации профориентационных мероприятий для школьников. Проведен анализ программных продуктов, являющихся аналогами мобильного приложения-тренажера для подготовки к ЕГЭ. Проведен анализ программных средств

разработки и обоснован выбор ПО «AndroidStudio». Спроектирована база данных

Во втором разделе описана реализация мобильного приложения с подключением к сетевой базе данных Firebase и возможностью обновления банка заданий в файловой системе средствами SQLite и программным языком Java, навигацией между окнами.

В третьем разделе проведен расчет затрат на реализацию проекта по методике ТСО. Отдельно произведен расчёт капитальных, эксплуатационных и прямых затрат. Проанализированы риски, выявлены вероятности и уровни влияния рисков соответствия, операционных рисков, реализационных рисков, приведены возможные пути минимизации каждого из рисков.

В результате выполнения выпускной квалификационной работыбыло разработаномобильное приложение «Тренажер ЕГЭ по математике».

1 Анализ предметной области проекта «Тренажер ЕГЭ по математике»

1.1 Характеристика основной деятельности ХТИ – филиала СФУ

Хакасский технический институт – филиал федерального государственного автономного образовательного учреждения высшего образования «Сибирский федеральный университет» – является обособленным подразделением федерального государственного автономного образовательного учреждения высшего образования «Сибирский федеральный университет» [1]. Институт был создан приказом Министерства высшего и среднего специального образования РСФСР в 1967 году и изначально именовался как общетехнический факультет Красноярского политехнического института в г. Абакане.

Миссией ХТИ – филиала СФУ является создание передовой образовательной, научно-исследовательской и инновационной инфраструктуры, продвижение новых знаний и технологий для решения задач социально-экономического развития Сибирского федерального округа, а также формирование кадрового потенциала – конкурентоспособных специалистов по приоритетным направлениям развития Сибири и Российской Федерации, соответствующих мировым стандартам [1].

Институт осуществляет подготовку по очной, заочной и очно-заочной формам обучения по направлениям специалитета, бакалавриата и магистратуры.

Ведётся подготовка специалистов и бакалавров в следующих областях: электроэнергетика; машиностроение; строительство; экономика; транспорт; информатика. Широко развиты программы дополнительного профессионального образования; повышение квалификации, профессиональная

переподготовка, тематические семинары для руководителей и специалистов предприятий и организаций.

Основными структурными подразделениями института являются:

- 3 кафедры;
- центр подготовки юного инженера;
- научно-образовательная лаборатория «Дендроэкология и экологический мониторинг»;
- отдел довузовской подготовки и нового набора;
- отдел информационных технологий;
- библиотека;
- центр студенческой культуры.

Имеется также 11 компьютерных классов (с выходом в сеть Интернет), медпункт, 2 спортивных зала, столовая, 2 студенческих общежития.

Институт располагает 2 учебными корпусами:

- корпус «А» - г. Абакан, ул. Щетинкина, 27;
- корпус «Б» - г. Абакан, ул. Комарова, 15.

Образовательные программы ХТИ – филиала СФУ направлены на обучение специалистов в сфере технической направленности.

Базовой дисциплиной для освоения всех специальных технических дисциплин является математика, которую студенты изучают на всех направлениях бакалавриата и специалитета вуза:

- 08.03.01 Строительство.
- 08.05.01 Строительство уникальных зданий и сооружений.
- 09.03.03 Прикладная информатика.
- 13.03.05 Конструкторско-технологическое обеспечение машиностроительных производств.
- 23.03.03 Эксплуатация транспортно-технологических машин и комплексов.

По окончании обучения лицам, успешно прошедшим государственную итоговую аттестацию, выдается документ об образовании и о квалификации, подтверждающий получение высшего образования.

Структурная схема ХТИ – филиала СФУ представлена на рисунке 1 [1].

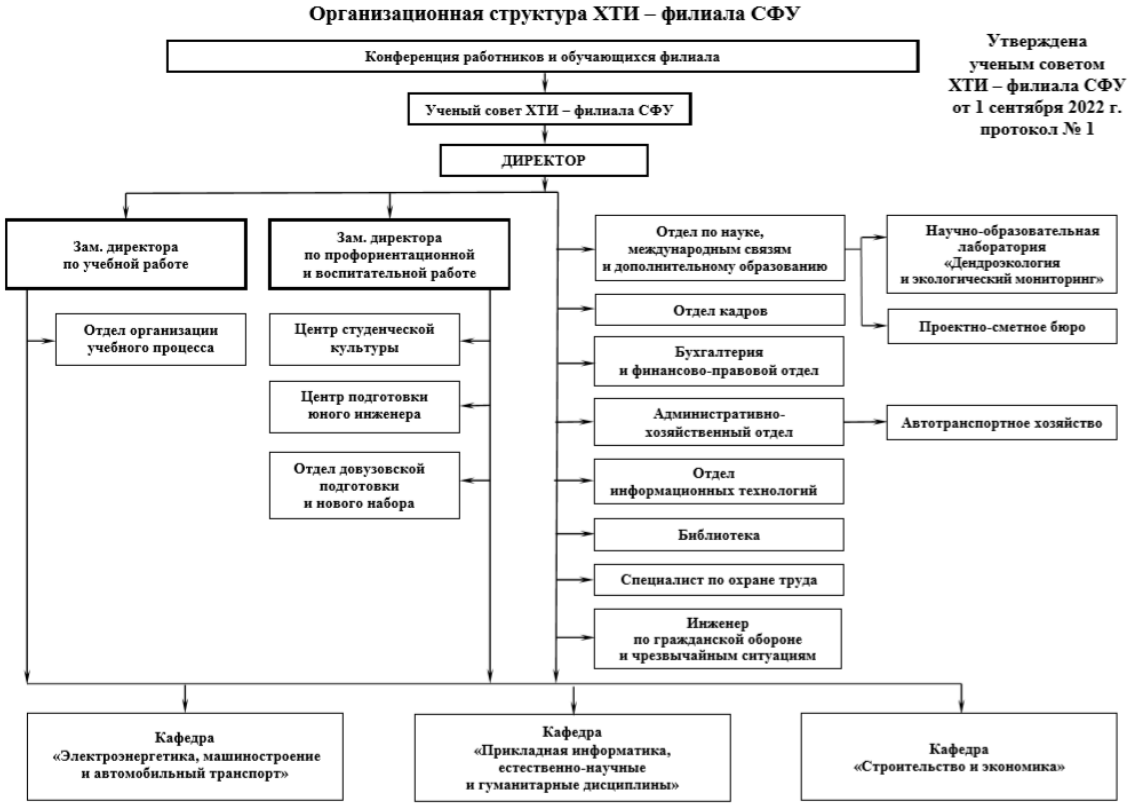


Рисунок 1 – Организационно-структурная схема ХТИ – филиала СФУ

Для обеспечения набора студентов на первый курс ХТИ – филиал СФУ регулярно проводит профориентационные мероприятия. Занимается этим отдел довузовской подготовки и нового набора (ОДПиНН).

Отдел довузовской подготовки и нового набора является частью Хакасского технического института – филиала Сибирского федерального университета. Руководство деятельностью ОДПиНН ведет начальник отдела, который подчиняется директору ХТИ-филиала СФУ. В структуру отдела входят начальник и специалисты. Отдел довузовской подготовки и нового набора разрабатывает планы и методы подготовки к поступлению,

профориентационных мероприятиям, дополнительной подготовке и вступительным экзаменам.

Основные направления деятельности ОДПиНН:

– координация работы структурных подразделений института в области профориентационной работы;

– определение общих принципов деятельности института по подготовке выпускников школ, организаций среднего профессионального образования, начального профессионального образования, выпускников прошлых лет к поступлению в ХТИ – филиал СФУ, ФГАОУ ВО «Сибирский федеральный университет»;

– работа по формированию нового набора;

– организация проведения рекламной кампании по набору на первый курс, разработка и утверждение рекламно-информационных материалов;

– разработка планов мероприятий, касающихся организации набора;

– организация приема документов от поступающих;

– организация приема посетителей по вопросам поступления в ХТИ – филиал СФУ, ФГАОУ ВО «Сибирский федеральный университет», проведение консультаций с поступающими, в том числе по вопросам выбора направления подготовки/специалитета, наиболее соответствующего их способностям, склонностям и подготовке;

– организация работы экзаменационной, апелляционной комиссий на площадке ХТИ – филиала СФУ;

– организация и контроль работы технического секретариата во время приемной кампании;

– осуществление работы с автоматизированной информационной системой (АИС) «Абитуриент» СФУ.

Каждый учебный год в ХТИ – филиале СФУ проводятся различные мероприятия, направленные на привлечение абитуриентов и обеспечение нового набора. Большая часть данных мероприятий направлена на

старшеклассников средних общеобразовательных школ, но также и на учащихся 1-8 классов, например, в дни открытых дверей.

Кроме того, ОДПиНН занимается привлечением лиц, желающих получить второе высшее образование по направлениям, реализуемым в ХТИ – филиале СФУ.

Мероприятия, направленные на привлечение абитуриентов и обеспечение нового набора, проведённые в 2022-2023 учебном году:

– День «открытых дверей»;

– «Технотворчество Хакасии-2023»;

– акция «Родная школа-вуз»;

– организация и проведение олимпиад: Ищем Ломоносовых, Учись строить будущее, Бельчонок, AIхимия будущего, Россети, и др. на базе ХТИ – филиала СФУ;

– участие в "Ярмарках профессий";

– день магистратуры;

– посещение закрепленных за кафедрами школ с рекламными проспектами для агитации школьников и проведение профориентационных работ (родительские собрания, классные часы);

– проведение научно-практической конференции школьников «Наука – наше будущее».

На основе данного списка мероприятий можно сделать вывод о том, что большинство мероприятий направлено на учащихся 10-11 классов, готовящихся сдавать ЕГЭ. Разработка мобильного приложения-тренажера позволит не только помочь абитуриентам подготовиться к сдаче ЕГЭ по математике, но и привлечь внимание к направлениям обучения ХТИ – филиала СФУ, проводящим набор по результатам данного экзамена.

1.2 Актуальность IT-проекта

Ежегодно в ХТИ – филиале СФУ проводятся стационарные и выездные профориентационные мероприятия, на которых необходимо применение интерактивных форм общения с учащимися средних образовательных школ, где школьникам показывают перспективы обучения в институте.

На данные мероприятия приглашаются дети от 1 до 11 классов в возрасте от 7 до 18 лет. Если младшим классам до выбора направления обучения ещё далеко, то обучающимся в 9-11 классах уже пора задуматься об этом выборе.

В связи с этим возникла идея разработки мобильного приложения по подготовке к единому государственному экзамену по профильной математике, что привлечёт интерес будущих абитуриентов к информационным технологиям и направлениям подготовки ХТИ – филиала СФУ. Интерес абитуриентов к вузу зачастую становится основополагающим критерием выбора последнего и положительно сказывается на новом наборе абитуриентов.

Мобильное приложение может активно использоваться не только во время проведения профориентационных мероприятий, но и на подготовительных курсах. Оно продемонстрирует абитуриентам возможности разработки актуальных для их жизненного периода образовательных приложений.

Основное преимущество приложения для подготовки к единому государственному экзамену по математике заключается в его ориентированности на мобильные устройства, что позволит абитуриентам готовиться к экзамену по математике в любом удобном месте.

1.3 Анализ аналогичных программных продуктов

Из похожих приложений можно выделить «ЕГЭ Математика» от разработчика «ЕГЭ 2023» и «Профмат – ЕГЭ Математика 2023» от

разработчицы Елены Скороходовой.

Приложение «ЕГЭ Математика» имеет минималистичный дизайн (рисунок 2). Что не отвлекает от процесса учёбы яркими цветами.

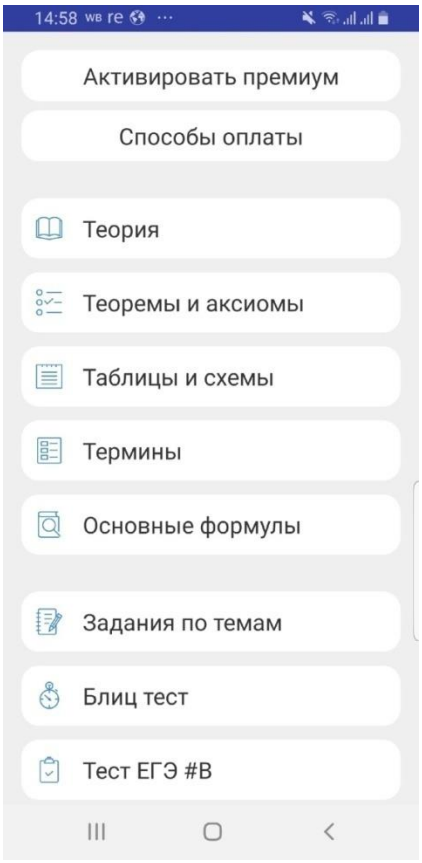


Рисунок 2 – Главное меню «ЕГЭ Математика»

В разделе «Теория» присутствуют мешающие восприятию информации знаки, которые не расшифровываются интуитивно (рисунок 3).

Рисунок 3 – Раздел «Теория»

Такие же знаки присутствуют в разделе «Задания по темам» (рисунок 4).

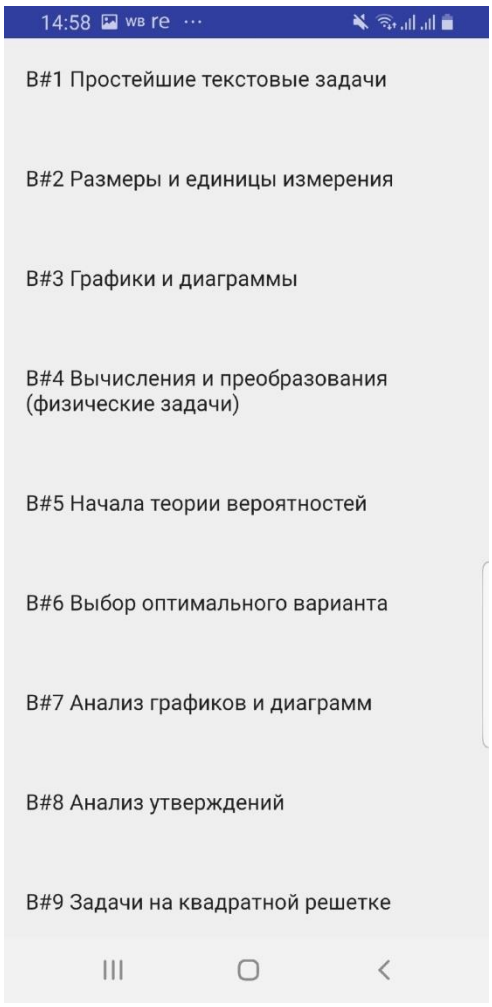


Рисунок 4 – Раздел «Задания по темам»

Раздел «Теория» представлен недостаточно подробно, и не имеет навигации (рисунок 5), что усложняет поиск и усвоение нужной информации.

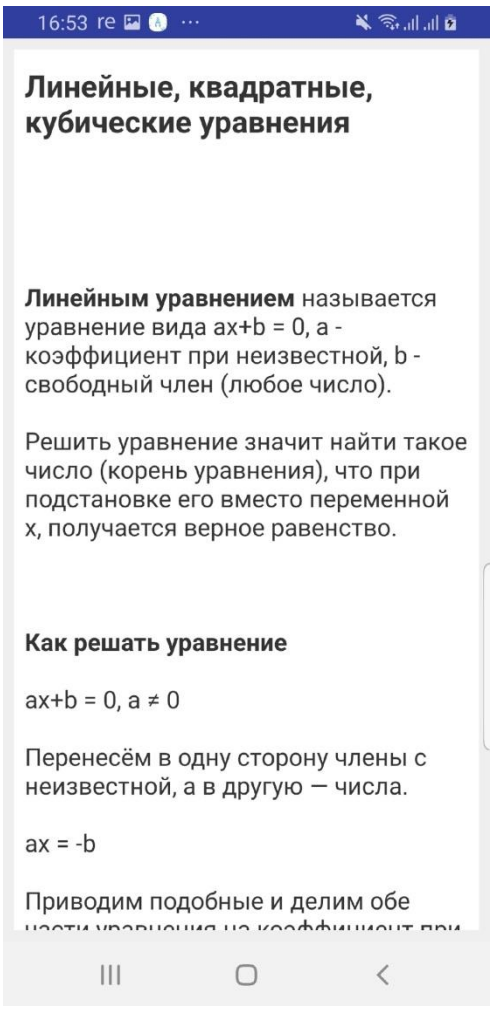


Рисунок 5 – Теоретическая информация в приложении

Приложение «Профмат – ЕГЭ Математика 2023» также имеет минималистичный дизайн (рисунок 6).

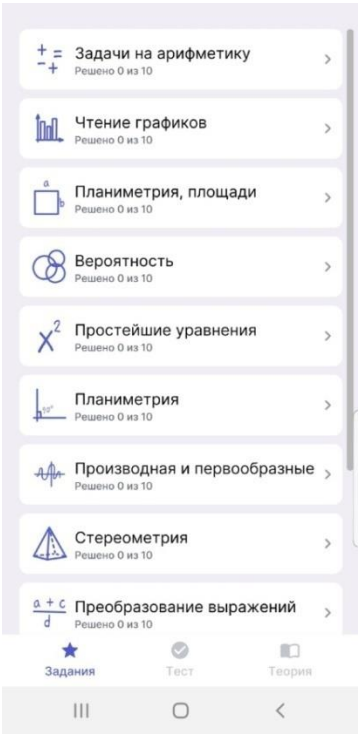


Рисунок 6 – Главное меню приложения «Профмат»

Теоретическая информация представлена в виде кратких опорных конспектов, что затрудняет ее изучение (рисунок 7).

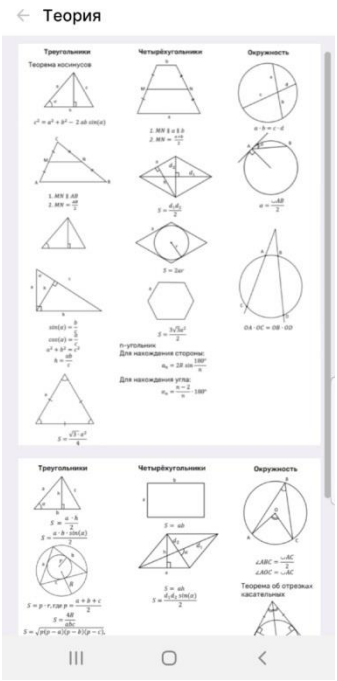


Рисунок 7 – Теоретическая информация в приложении «Профмат»

Список заданий структурирован и имеет метку о выполненном задании в виде круга слева от текста (рисунок 8).

Рисунок 8 – Список заданий

В приложении есть неоткрывающиеся разделы, и оно зависает при прокрутке экрана, что является следствием разработки на тяжёлой архитектуре игрового движка Unity, на котором сделано мобильное приложение.

Сравнив два этих мобильных приложения, можно прийти к выводам, как сделать более качественное мобильное приложение,объединив плюсы и учтя недостатки этих двух приложений.

1.4 Структурное моделирование описания бизнес-процесса, подлежащего автоматизации

Диаграмма потоков данных (DFD) позволяет отследить то, как мобильное приложение взаимодействует с внешними сущностями, а также как протекает информация внутри [2].

В данном случае внешней сущностью являются пользователь приложения, выполняющий задания вариантов ЕГЭ по математике.

От пользователей приложению поступает результат выполненных заданий. Приложение после ввода ответа пользователем сохраняет результат.

Контекстнаямодель приложения изображена на рисунке 9.

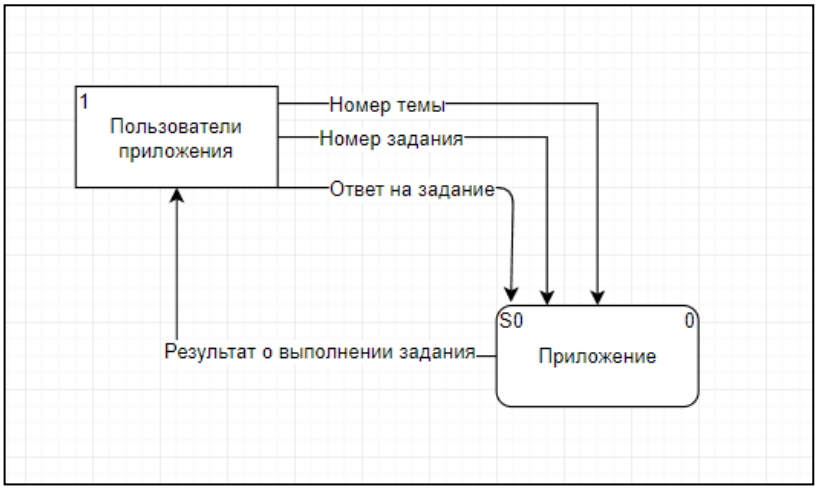


Рисунок 9 – Контекстная модель приложения

Далее стоит рассмотреть, как протекает получаемая информация внутри приложения (рисунок 10). При запуске приложения для пользователя открывается меню тем заданий. После выбора темы открывается список заданий. После выполнения задания результат выполнения приходит пользователю [3].

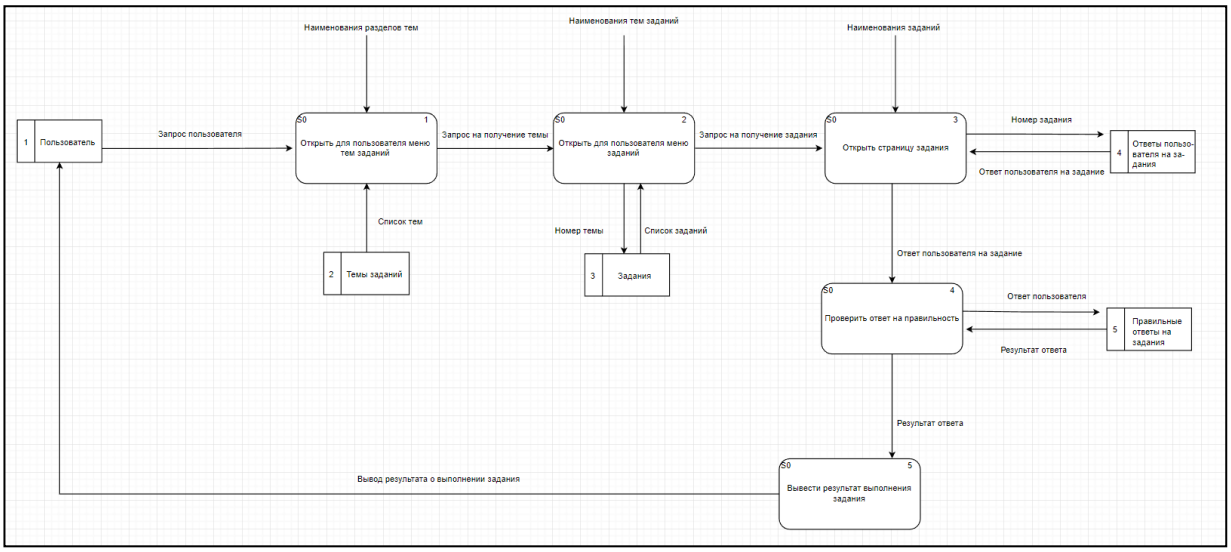


Рисунок 10 – Модель DFD

1.5 Проектирование базы данных

Для работы с большим количеством данных необходимо спроектировать базу данных.

В базе данных необходимо хранить информацию о темах заданий (таблица «Темы заданий»): содержит единственное поле, которое содержит название темы для заданий. Также база данных содержит информацию с пользовательскими данными, в данном случае это имя пользователя, которое хранится в таблице «Данные пользователя» в поле «имя пользователя». Таблица «Разделы приложения» содержит поля «название раздела» и «тема задания». Таблица «Задания» содержит поля «id задания», «название задания», «описание задания», «тема задания», «id ответа задания», «картинка задания», «состояние выполнения задания», «номер задания», «ответ пользователя на задание». Таблица «Ответы заданий» содержит поля «id ответа задания», «картинка ответа», «текст ответа». Между сущностями установлена связь «один ко многим». Спроектированная база данных представлена на рисунке 11.

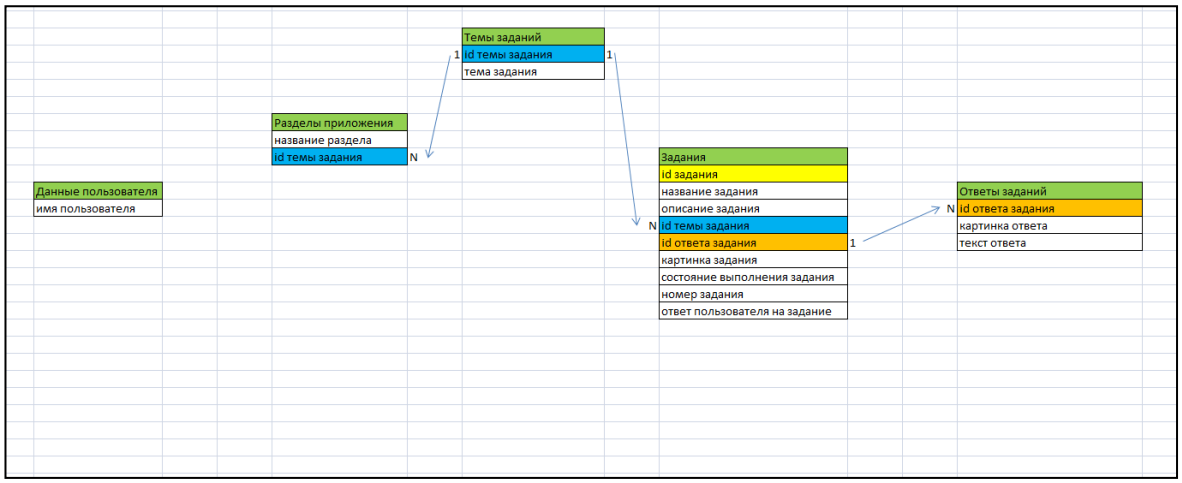


Рисунок 11 – Модель базы данных приложения

1.6 Выбор средств разработки

От выбора средств разработки зависит качество выпускаемого программногопродукта и время на его создание. Современная среда разработки способна сама ещё на стадии написания кода находить возможные ошибки и предлагать решения. Проведем сравнительную характеристику самых популярных инструментов (таблица 1) для разработки мобильного приложения под ОС «Андройд»: IntelliJIdea, Eclipse и AndroidStudio [4].

Таблица 1 – Сравнение IntelliJ Idea, Eclipse и Android Studio

Критерий	IntelliJ Idea	Eclipse	Android Studio
Поддерживаемые языки	Java Scala Groovy Kotlin JavaScript TypeScript SQL	Java C C++ C# JavaScript Python	Java C C++ Kotlin
Работает на	Windows MacOS Linux	Любая ОС, поддерживающая Java	Windows MacOS Linux
Целевая платформа	Любая ОС, поддерживающая Java	Android iOS Linux MacOS Windows	Android

Окончание таблицы 1

Лицензия	Лицензия на свободное программное обеспечение ApacheSoftwareFoundation.	EclipsePublicLicense	Свободно-распространяемое ПО
Цена	Бесплатно или до $499/год	Бесплатно	Бесплатно

Исходя из сравнения, более предпочтительным инструментом для разработки является AndroidStudio. Так как это более специализированная среда разработки для приложений на ОС «Android».

Выводы по разделу «Анализ предметной области проекта ″Тренажер ЕГЭ по математике″»

В данном разделе проведен анализ предметной области проекта. Описана характеристика основной деятельности ХТИ – филиала СФУ и актуальность IT-проекта.Также проведен анализ программных продуктов для определения основных характеристик разработки. Выполнено структурное моделирование описания бизнес-процесса. Спроектирована база данных приложения. На основе сравнения программных средств обоснован выбор средств разработки.

2 Разработка приложения «Тренажер ЕГЭ по математике»

2.2 Разработка интерфейса приложения

Стартовая страница приложения содержит 3 кнопки: «Теория», «Практика» и «Настройки». Внешний вид стартовой страницы изображен на рисунке 12. Стиль интерфейса приложения согласован с корпоративными цветами ХТИ – филиала СФУ.

При нажатии на кнопку «Теория» открывается раздел, в котором пользователь может просмотреть теоретический материал по интересующей его

теме. При нажатии на кнопку «Практика» открывается раздел, в котором пользователь может решать задачи на интересующую его тему. Кнопка «Настройки» ведёт к открытию страницы настроек, где пользователь может выбрать подходящую для себя цветовую тему.

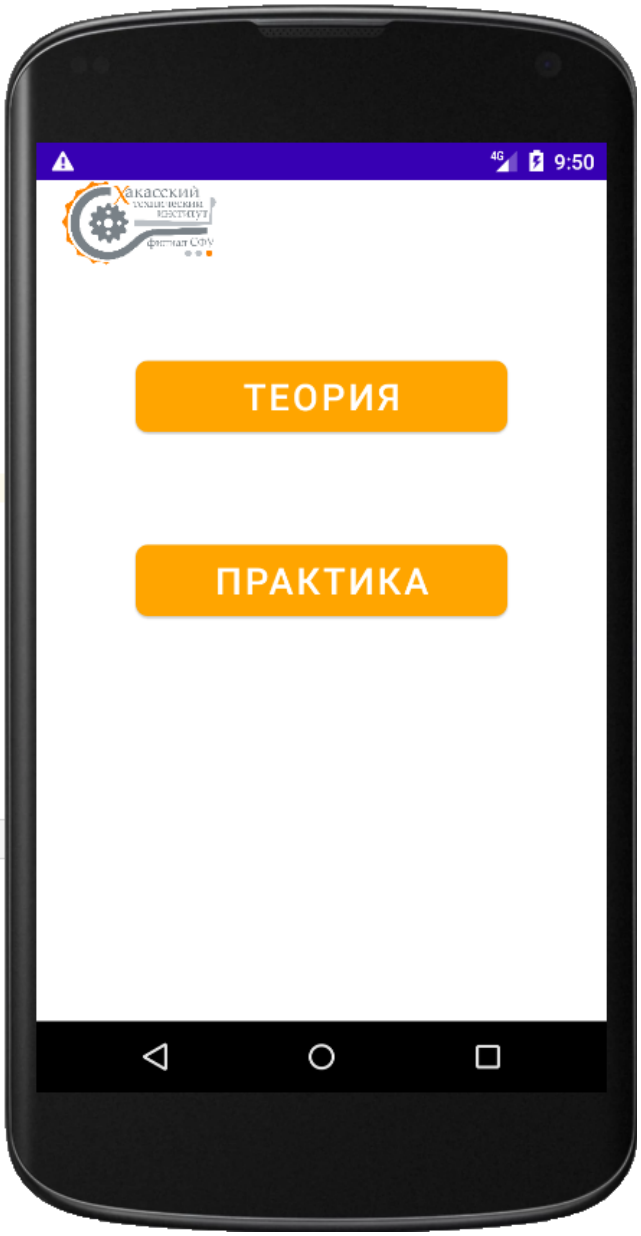


Рисунок 12 – Главная страница приложения

Программный код, задающий графические элементы, представлен на рисунке 13. Код написан на языке XML, который используется для верстки

пользовательского интерфейса в приложении. Он позволяет создавать различные элементы интерфейса, такие как кнопки, текстовые поля, изображения и многое другое. Каждый элемент имеет свои атрибуты, которые позволяют настраивать его внешний вид и поведение [5].

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="345dp"
        android:layout_height="287dp"
        android:layout_marginBottom="16dp"



        android:text=""
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent" />

    <Button
        android:id="@+id/b_Theory"
        android:layout_width="250dp"
        android:layout_height="60dp"
        android:textSize="25sp"
        android:layout_marginTop="100dp"
        android:text="Теория"

        app:cornerRadius="8dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.525"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

```

Рисунок 13 – Программный код activity_main.xml, лист 1

Рисунок 13, лист 2

В коде представлен элемент ConstraintLayout, который позволяет задавать ограничения для размещения элементов в пользовательском интерфейсе. Этот элемент является одним из основных элементов разметки в AndroidStudio и позволяет создавать гибкие и адаптивные макеты.

ConstraintLayout работает на основе концепции ограничений, которые определяют расположение и размеры элементов в макете. Ограничения могут быть заданы для каждого элемента, указывая его положение относительно других элементов или границ экрана.

В отличие от других элементов разметки, ConstraintLayout позволяет создавать более сложные макеты с различными условиями и ограничениями. Например, можно задать условия, при которых элемент должен занимать определенный процент от экрана или должен быть выровнен по центру экрана.

Для работы с ConstraintLayout в коде приложения используется API, который позволяет задавать ограничения для элементов в макете и управлять их расположением и размерами. Это делает элемент ConstraintLayout очень

мощным инструментом для создания гибких и адаптивных макетов в приложениях Android.

Также представлен элемент Button, который является одним из основных элементов пользовательского интерфейса в приложениях Android. Кнопка позволяет пользователю выполнить определенное действие при нажатии на нее.

В коде приложения на языке Java кнопки создаются программно с помощью класса Button и устанавливаются их свойства, такие как текст, цвет фона, цвет текста и т.д. Кроме того, кнопки могут быть настроены для вызова определенных методов при нажатии, что позволяет реализовать нужную функциональность в приложении.

В разметке приложения на языке XML, кнопки создаются с помощью тега Button и задаются атрибуты для установки ее свойств, например, текст кнопки, цвет фона и цвет текста. Кнопки могут быть расположены на экране в любом месте и настроены для выполнения нужных действий при нажатии.

Кнопки в Android могут иметь различные стили и виды. Например, кнопки могут быть с плоскими или выпуклыми краями, с различными иконками, текстом и т.д. Кнопки также могут быть оформлены в виде переключателей, флажков или плавающих кнопок, что позволяет создавать более интересный и функциональный пользовательский интерфейс.

Также представлен элемент TextView, который является элементом пользовательского интерфейса в приложениях Android для отображения текста на экране устройства. TextView позволяет отображать текст в различных стилях, шрифтах, размерах, цветах и форматах.

В коде приложения на языке Java, элемент TextView создается с помощью класса TextView и устанавливаются его свойства, такие как текст, цвет, шрифт и т.д. Элемент TextView также может быть настроен для выполнения различных действий, например, для перехода на другой экран или для вызова определенной функции при нажатии.

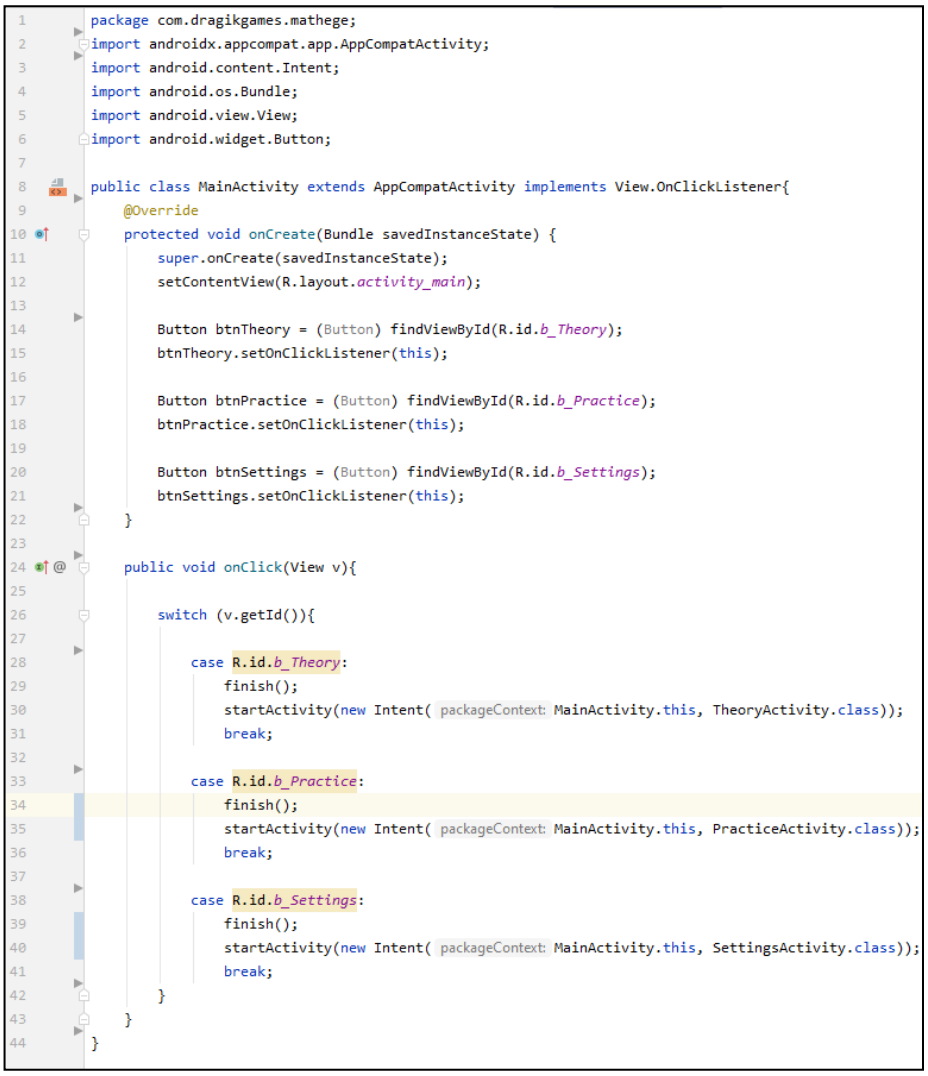
В разметке приложения на языке XML, элемент TextView создается с помощью тега TextView, и задаются атрибуты для установки ее свойств, например, текст, цвет, шрифт и т.д. Элемент TextView может быть размещен в любом месте на экране и настроен для отображения текста в нужном формате.

TextView также может быть настроен для отображения текста с использованием различных форматирований, таких как жирный, курсивный, подчеркнутый текст и т.д. Кроме того, TextView может быть настроен для отображения ссылок, изображений и других элементов внутри текста.

В целом, элемент TextView является одним из основных элементов пользовательского интерфейса в приложениях Android и позволяет отображать текст на экране устройства в нужном формате и стиле.

2.3 Реализация логики приложения

Логика приложения написана на языке Java. За стартовую страницу отвечает класс MainActivity.java, код этого класса представлен на рисунке 14.



```
package com.dragikgames.mathege;
import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

public class MainActivity extends AppCompatActivity implements View.OnClickListener{
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        Button btnTheory = (Button) findViewById(R.id.b_Theory);
        btnTheory.setOnClickListener(this);

        Button btnPractice = (Button) findViewById(R.id.b_Practice);
        btnPractice.setOnClickListener(this);

        Button btnSettings = (Button) findViewById(R.id.b_Settings);
        btnSettings.setOnClickListener(this);
    }

    public void onClick(View v){

        switch (v.getId()){

            case R.id.b_Theory:
                finish();
                startActivity(new Intent( packageContext: MainActivity.this, TheoryActivity.class));
                break;

            case R.id.b_Practice:
                finish();
                startActivity(new Intent( packageContext: MainActivity.this, PracticeActivity.class));
                break;

            case R.id.b_Settings:
                finish();
                startActivity(new Intent( packageContext: MainActivity.this, SettingsActivity.class));
                break;
        }
    }
}
```

Рисунок 14 – Программный код MainActivity.java

Этот код представляет основную активность приложения "MainActivity", которая наследуется от класса "AppCompatActivity". Она отображает пользовательский интерфейс, определенный в файле "activity_main.xml".

В методе "onCreate" происходит установка пользовательского интерфейса с помощью метода "setContentView", который загружает макет из файла "activity_main.xml". Затем, с помощью метода "findViewById", определяются три элемента Button с идентификаторами "b_Theory", "b_Practice" и "b_Settings". Эти кнопки будут использоваться для перехода на другие активности приложения.

Для каждой кнопки устанавливается обработчик нажатий с помощью метода "setOnClickListener", который связывает текущий объект MainActivity с реализацией метода onClick.

Метод onClick реализует обработчик нажатий на кнопки. При нажатии на кнопку "b_Theory", "b_Practice" или "b_Settings" вызывается соответствующая активность "TheoryActivity", "PracticeActivity" или "SettingsActivity" с помощью метода startActivity. Метод finish() завершает текущую активность MainActivity и освобождает ее ресурсы.

Таким образом, данный код отвечает за функциональность главного экрана приложения, который содержит три кнопки для перехода на другие экраны.

Кроме того, данный код реализует интерфейс "View.OnClickListener", который определяет метод "onClick", обрабатывающий нажатие на кнопки. Этот интерфейс используется для установки обработчиков нажатий на кнопки, что облегчает процесс управления интерфейсом и обработки событий в приложении.

Также, в данном коде используется интент для перехода на другие активности. Интенты - это объекты, которые используются для связи между компонентами приложения, в данном случае между MainActivity и другими активностями. Они позволяют передавать данные и инструкции между активностями, запускать новые активности и возвращаться обратно на предыдущую активность.

В итоге, данный код использует базовые концепции Android-разработки, такие как активности, макеты, обработчики нажатий, интенты и многое другое. Это позволяет создавать полноценные приложения для Android, которые могут выполнять широкий спектр задач.

Для работы с базой данных был разработан специальный класс DatabaseHelper.java, представлен на рисунке 15.

```java
public class DatabaseHelper extends SQLiteOpenHelper {
    private static String DB_NAME = "EGEmath.db";
    private static String DB_PATH = "";
    private static final int DB_VERSION = 1;

    private SQLiteDatabase mDataBase;
    private final Context mContext;
    private boolean mNeedUpdate = false;

    public DatabaseHelper(Context context) {
        super(context, DB_NAME,  factory: null, DB_VERSION);
        if (android.os.Build.VERSION.SDK_INT >= 17)
            DB_PATH = context.getApplicationInfo().dataDir + "/databases/";
        else
            DB_PATH = "/data/data/" + context.getPackageName() + "/databases/";
        this.mContext = context;

        copyDataBase();

        this.getReadableDatabase();
    }

    public void updateDataBase() throws IOException {
        if (mNeedUpdate) {
            File dbFile = new File( pathname: DB_PATH + DB_NAME);
            if (dbFile.exists())
                dbFile.delete();

            copyDataBase();

            mNeedUpdate = false;
        }
    }

    private boolean checkDataBase() {
        File dbFile = new File( pathname: DB_PATH + DB_NAME);
        return dbFile.exists();
    }
```

Рисунок 15 – Программный код класса DatabaseHelper.java, лист 1

```
59      private void copyDataBase() {
60          if (!checkDataBase()) {
61              this.getReadableDatabase();
62              this.close();
63              try {
64                  copyDBFile();
65              } catch (IOException mIOException) {
66                  throw new Error("ErrorCopyingDataBase");
67              }
68          }
69      }
70
71      private void copyDBFile() throws IOException {
72          InputStream mInput = mContext.getAssets().open(DB_NAME);
73          OutputStream mOutput = new FileOutputStream( name: DB_PATH + DB_NAME);
74          byte[] mBuffer = new byte[1024];
75          int mLength;
76          while ((mLength = mInput.read(mBuffer)) > 0)
77              mOutput.write(mBuffer,  off: 0, mLength);
78          mOutput.flush();
79          mOutput.close();
80          mInput.close();
81      }
82
83      public boolean openDataBase() throws SQLException {
84          mDataBase = SQLiteDatabase.openDatabase( path: DB_PATH + DB_NAME,  factory: null, SQLiteDatabase.CREATE_IF_NECESSARY);
85          return mDataBase != null;
86      }
87
88      @Override
89      public synchronized void close() {
90          if (mDataBase != null)
91              mDataBase.close();
92          super.close();
93      }
94
95      @Override
96      public void onCreate(SQLiteDatabase db) {
97
98      }
99
```

Рисунок 15, лист 2

```
100     @Override
101     public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
102         if (newVersion > oldVersion)
103             mNeedUpdate = true;
104     }
105
106
107     public Bitmap getImage(String name) {
108         SQLiteDatabase db = getReadableDatabase();
109         String[] projection = {"image"};
110         String selection = "name=?";
111         String[] selectionArgs = {name};
112         Cursor cursor = db.query( table: "tasks", projection, selection, selectionArgs,  groupBy: null,  having: null,  orderBy: null);
113         Bitmap bitmap = null;
114         if (cursor != null && cursor.moveToFirst()) {
115             int imageIndex = cursor.getColumnIndexOrThrow( s: "image");
116             byte[] imageBytes = cursor.getBlob(imageIndex);
117             bitmap = BitmapFactory.decodeByteArray(imageBytes,  offset: 0, imageBytes.length);
118         }
119         if (cursor != null) {
120             cursor.close();
121         }
122         return bitmap;
123     }
124
```

Рисунок 15, лист 3

```
public List<Pair<Integer, String>> getTaskThemes() {
    SQLiteDatabase db = getReadableDatabase();
    String[] projection = {"Id_Task_theme", "Task_theme"};
    Cursor cursor = db.query( table: "TaskThemes", projection, selection: null, selectionArgs: null, groupBy: null, having: null, orderBy: null);
    List<Pair<Integer, String>> taskThemes = new ArrayList<>();
    if (cursor != null && cursor.moveToFirst()) {
        do {
            int idIndex = cursor.getColumnIndexOrThrow( s: "Id_Task_theme");
            int id = cursor.getInt(idIndex);
            int taskThemeIndex = cursor.getColumnIndexOrThrow( s: "Task_theme");
            String taskTheme = cursor.getString(taskThemeIndex);
            taskThemes.add(new Pair<>(id, taskTheme));
        } while (cursor.moveToNext());
    }
    if (cursor != null) {
        cursor.close();
    }
    return taskThemes;
}
```

Рисунок 15, лист 4

```
public List<Pair<Integer, String>> getTaskButtons(int taskThemeId) {
    SQLiteDatabase db = getReadableDatabase();
    String[] projection = {"TaskId", "TaskName"};
    String selection = "TaskThemeId=?";
    String[] selectionArgs = {String.valueOf(taskThemeId)};
    Cursor cursor = db.query( table: "Tasks", projection, selection, selectionArgs, groupBy: null, having: null, orderBy: null);
    List<Pair<Integer, String>> taskButtons = new ArrayList<>();
    if (cursor != null && cursor.moveToFirst()) {
        do {
            int idIndex = cursor.getColumnIndexOrThrow( s: "TaskId");
            int id = cursor.getInt(idIndex);
            int taskNameIndex = cursor.getColumnIndexOrThrow( s: "TaskName");
            String taskName = cursor.getString(taskNameIndex);
            taskButtons.add(new Pair<>(id, taskName));
        } while (cursor.moveToNext());
    }
    if (cursor != null) {
        cursor.close();
    }
    return taskButtons;
}
```

Рисунок 15, лист 5

2.4 Заполнение базы данных заданий

Класс DatabaseHelper наследуется от класса SQLiteOpenHelper и используется для управления базой данных SQLite. База данных используется для хранения информации, которая используется в приложении. В данном случае база данных называется EGEmath.db.

В конструкторе класса DatabaseHelper создаются переменные, которые будут использоваться позже в коде. В переменную DB_NAME записывается имя базы данных. Переменная DB_PATH используется для хранения пути к базе данных, который будет использоваться для открытия и записи данных. В переменную mContext записывается контекст приложения. Контекст используется для получения доступа к ресурсам приложения, таким как файлы, изображения и т.д.

В методе onCreate класса DatabaseHelper создается база данных, если ее еще нет. В данном случае нет необходимости создавать таблицы, так как база данных уже создана и заполнена данными.

Метод onUpgrade используется для обновления базы данных, если версия базы данных, определенная в DB_VERSION, больше, чем версия, которая уже установлена на устройстве. Если версия базы данных больше, то mNeedUpdate устанавливается в true, что означает, что базу данных нужно обновить.

Метод checkDataBase() проверяет, существует ли база данных в указанном месте. Если база данных существует, метод возвращает true, иначе false.

Метод copyDataBase() копирует базу данных из папки assets в папку /databases/ на устройстве. Если база данных уже существует, то она не будет скопирована. Если база данных не существует, то она будет скопирована из assets в /databases/ на устройстве.

Метод openDataBase() открывает базу данных для чтения и записи. Если база данных не существует, она будет создана.

Метод close() закрывает базу данных, если она была открыта.

Метод getImage(Stringname) позволяет извлекать из базы данных изображение, соответствующее заданному имени.

Сначала получается экземпляр SQLiteDatabase, который используется для выполнения запроса к базе данных. Затем определяются параметры запроса - проекция (projection), выборка (selection) и аргументы выборки (selectionArgs).

Проекция определяет, какие столбцы из таблицы будут выбраны в результате запроса. Здесь используется только один столбец "image", который содержит изображение в виде массива байтов.

Далее определяется условие выборки - "name=?", где знак "?" является аргументом, который будет заменен на соответствующий аргумент из selectionArgs. В данном случае аргументом будет имя изображения.

Затем выполняется запрос с помощью метода query() экземпляра SQLiteDatabase. В качестве аргументов передаются название таблицы ("tasks"), поле таблицы ("image"), условие выборки ("name=?"), аргументы выборки (массив с одним элементом - именем изображения), а также остальные необязательные параметры, которые не используются в данном случае (null).

Если запрос успешно выполнен и полученный курсор содержит хотя бы одну запись, то из него извлекается изображение. Сначала получается индекс столбца "image" с помощью метода getColumnIndexOrThrow(), а затем само изображение в виде массива байтов с помощью метода getBlob(). Этот массив байтов затем преобразуется в экземпляр класса Bitmap с помощью метода decodeByteArray() класса BitmapFactory.

После извлечения изображения курсор закрывается, а метод возвращает полученное изображение в виде экземпляра класса Bitmap.

Метод getTaskThemes()используется для получения списка тем заданий из базы данных.

Вначале метод создает экземпляр SQLiteDatabase для доступа к базе данных в режиме чтения. Затем определяется массив projection, который содержит имена столбцов, которые будут выбраны из таблицы "TaskThemes".

Далее выполняется запрос к базе данных с помощью метода query(), передавая имя таблицы, массив projection, аргументы для условия выборки (в данном случае не указано, поэтому все строки выбираются), а также другие необязательные параметры, такие как сортировка и группировка. Результаты запроса сохраняются в Cursor, который представляет собой набор результатов.

Если Cursor не равен null и содержит записи (то есть база данных не пустая), то происходит перебор каждой записи в цикле do-while. Внутри цикла получаются значения идентификатора и названия темы задания для текущей записи, используя методы getColumnIndexOrThrow() и getString(). Эти значения добавляются в список taskThemes с помощью класса Pair, который объединяет идентификатор и название в один объект.

После завершения цикла Cursor закрывается с помощью метода close(), чтобы освободить ресурсы. Затем список taskThemes, содержащий все полученные темы заданий, возвращается из метода.

Метод getTaskButtons(inttaskThemeId) похож на getTaskThemes(), но возвращает список кнопок заданий для заданной темы задания.

Вначале метод создает экземпляр SQLiteDatabase для доступа к базе данных в режиме чтения. Затем определяется массив projection, содержащий имена столбцов, которые будут выбраны из таблицы "Tasks".

Затем задается условие выборки с помощью строки selection и массива selectionArgs. В данном случае условие выборки основано на taskThemeId, переданном в качестве аргумента метода.

Выполняется запрос к базе данных с использованием метода query(), передавая имя таблицы, массив projection, условие выборки и другие параметры. Результаты запроса сохраняются в Cursor.

Если Cursor не равен null и содержит записи, то происходит перебор каждой записи в цикле do-while. Внутри цикла получаются значения идентификатора и названия задания для текущей записи, а затем они добавляются в список taskButtons с помощью класса Pair.

После завершения цикла Cursor закрывается с помощью метода close(), чтобы освободить ресурсы. Затем список taskButtons, содержащий все полученные кнопки заданий для заданной темы, возвращается из метода.

В итоге, класс DatabaseHelper представляет собой вспомогательный класс для работы с базой данных SQLite, который позволяет создавать, копировать,

открывать и закрывать базу данных, а также выполнять запросы и извлекать данные из базы данных. В данном случае класс используется для извлечения изображений из таблицы базы данных, которые затем используются в приложении [6].

Для проекта необходимо подготовить готовую базу данных. В качестве СУБД была выбрана SQLite (рисунок 16), как наиболее удобное средство разработки.

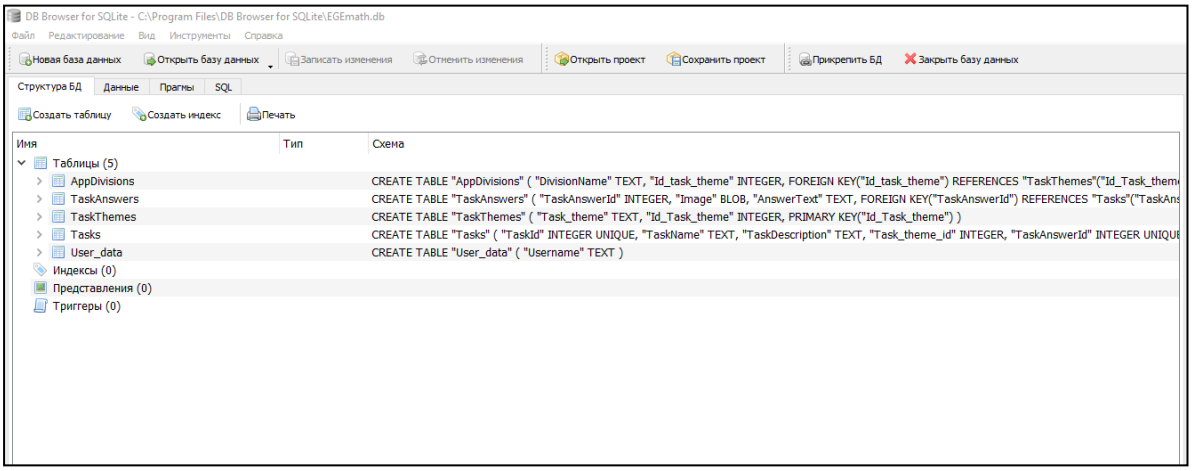


Рисунок 16 – Структура БД в SQLite

Так как записей в таблице должно быть много, то нужно автоматизировать процесс заполнения таблиц БД. Для этого нужны подготовленные данные, в данном проекте данные хранятся вфайловой системы ОС Windows (рисунок 17).

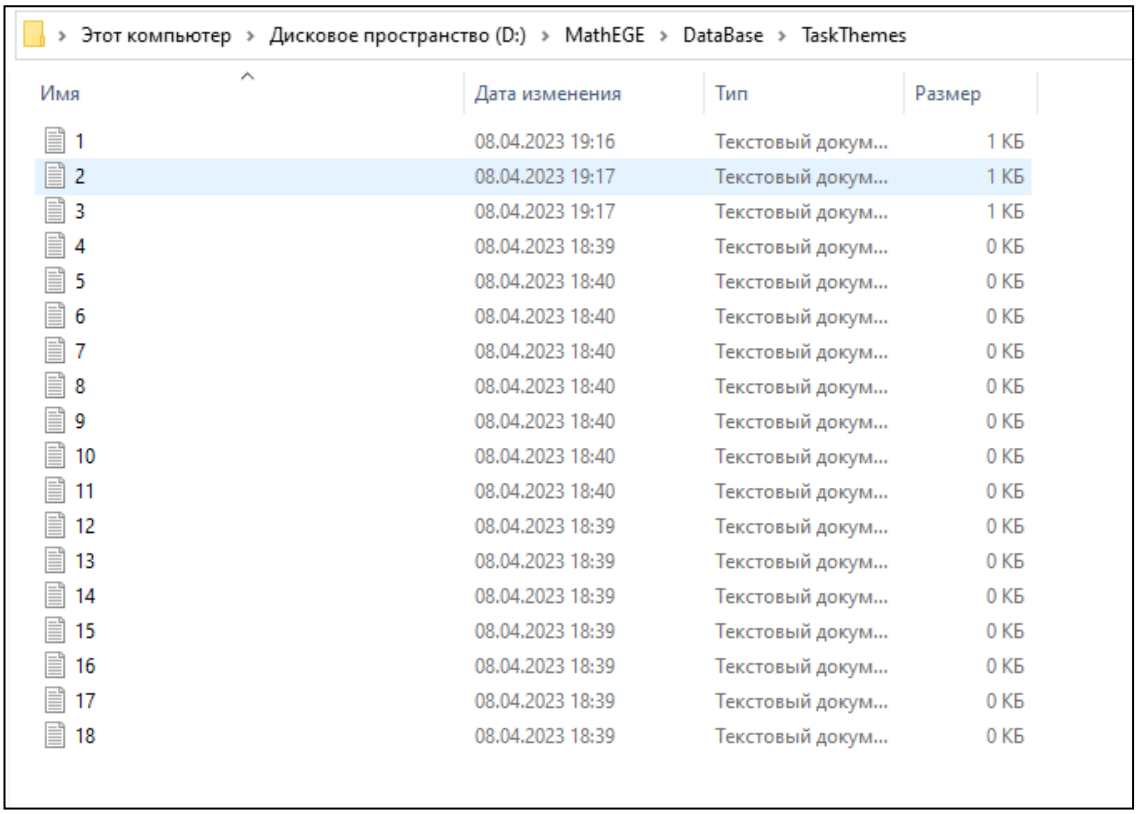


Рисунок 17 – Данные для таблицы TaskThemes

В папке Tasks(рисунок 18) находятся папки, названия которых соответствуют idтем заданий.

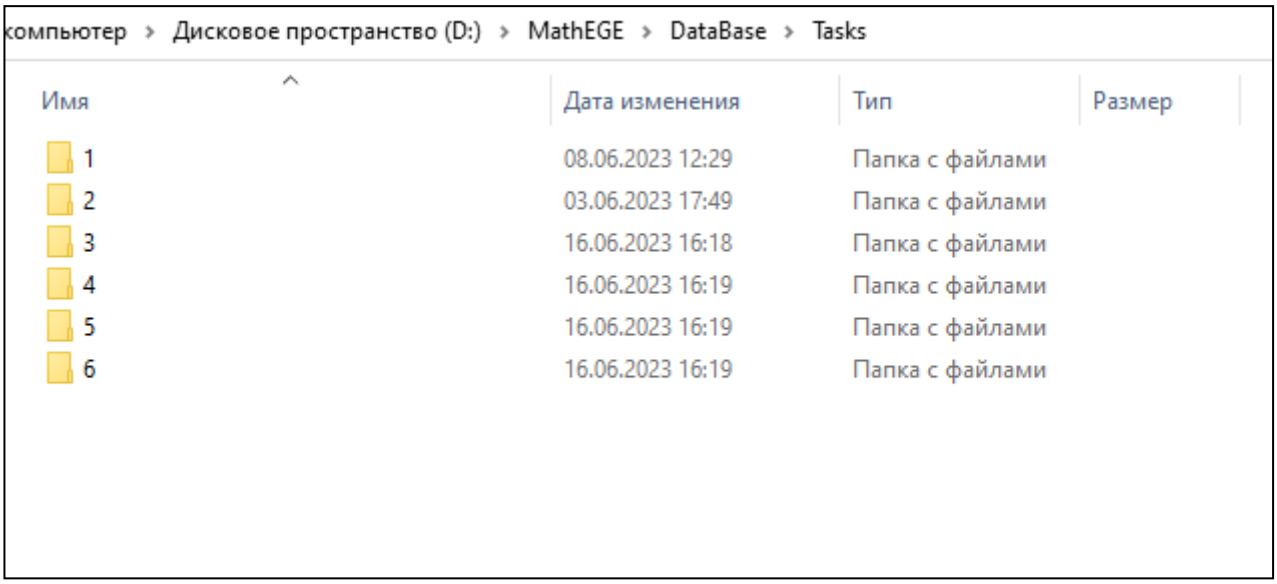


Рисунок 18 – Содержимое папки Tasks

В папке темы заданий (рисунок 19) находятся папки, названия которых соответствуют idзаданий.

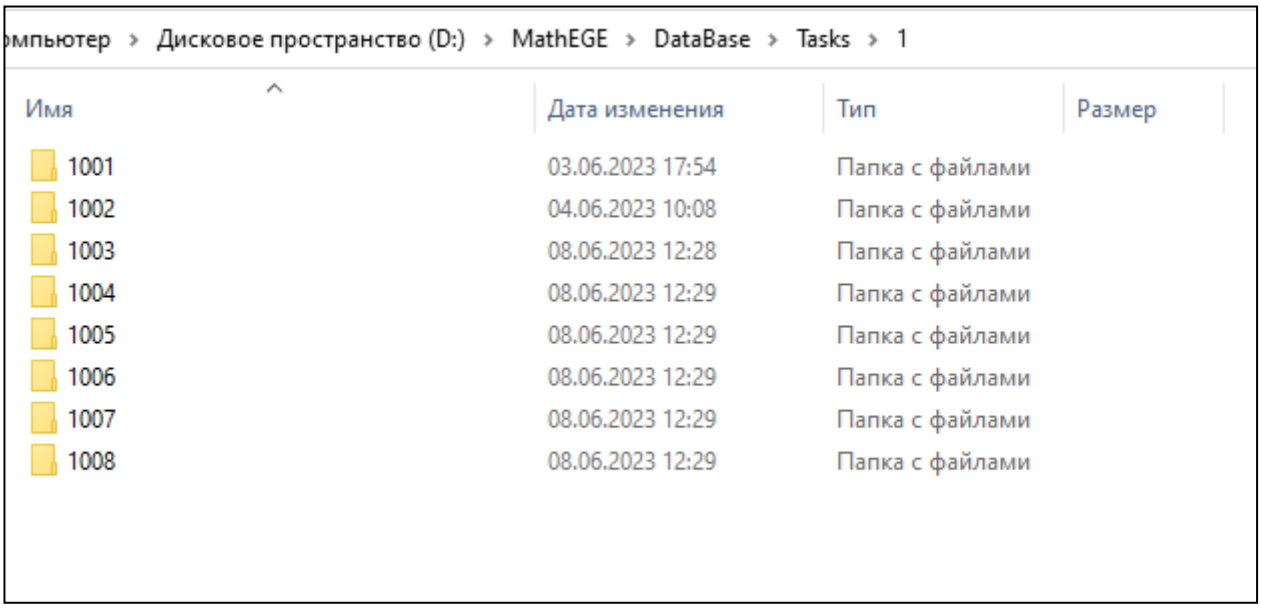


Рисунок 19 – Содержимое темы заданий

В папке idзадания (рисунок 20) находится информация по заданию: answertext–текст ответа, rightanswer–правильный ответ, taskdesc–описание задания, taskname–название задания, у этих файлов расширение txt, а taskimage–это изображение и этот файл имеет расширение png.

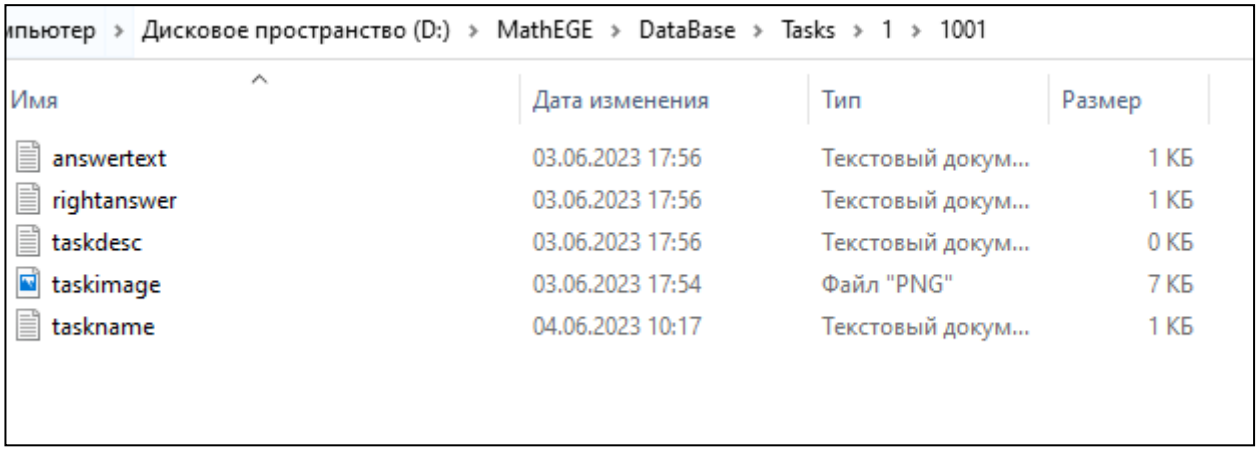


Рисунок 20 – Содержимое задания

Скрипт, осуществляющий заполнение таблицTaskThemesи Tasks представлен на рисунке 21.

Рисунок 21 – Программный код, заполняющий таблицы TaskThemes и Tasks

Рисунок 21, лист 2

```
                // Извлекаем данные из файлов
                // String taskId = readFile(taskIdFile);
                int taskNumber = taskId - 1000*taskThemeIdInt;
                String taskname = readFile(taskNameFile);
                String taskdesc = readFile(taskDescFile);
                String rightanswer =readFile(rightAnswerFile);
                String answertext = readFile(answerTextFile);
                byte[] answerimage = readBytes(answerImageFile);
                byte[] taskimage = readBytes(taskImageFile);

                // Записываем данные в базу данных
                PreparedStatement pstmt = conn.prepareStatement( sql: "INSERT INTO Tasks (TaskId, TaskName, TaskDescription, TaskAnswerText, AnswerImage, " +
                        "TaskImage, TaskThemeId, TaskNumber, RightAnswer) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?)");
                pstmt.setInt( parameterIndex: 1, taskId);
                pstmt.setString( parameterIndex: 2, taskname);
                pstmt.setString( parameterIndex: 3, taskdesc);
                pstmt.setString( parameterIndex: 4, answertext);
                pstmt.setBytes( parameterIndex: 5, answerimage);
                pstmt.setBytes( parameterIndex: 6, taskimage);
                pstmt.setString( parameterIndex: 7, taskThemeId);
                pstmt.setInt( parameterIndex: 8, taskNumber);
                pstmt.setString( parameterIndex: 9, rightanswer);
                pstmt.executeUpdate();
                pstmt.close();



            }
        }

        conn.close();
    } catch (Exception e) {
        e.printStackTrace();
    }
```

Рисунок 21, лист 3

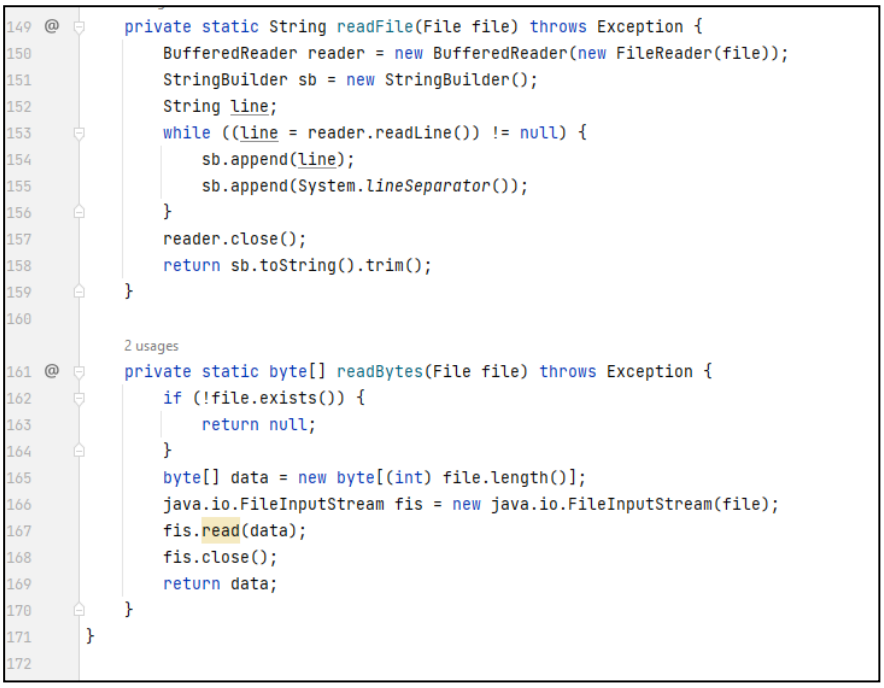
```
    try {
        Class.forName( className: "org.sqlite.JDBC");
        Connection conn = DriverManager.getConnection( url: "jdbc:sqlite:" + databasePath);

        File textDirectory = new File(textDirectoryPath);
        File[] files = textDirectory.listFiles();

        for (File file : files) {
            String name = file.getName();
            String id = name.replaceAll( regex: "[^0-9]", replacement: "");
            StringBuilder sb = new StringBuilder();
            BufferedReader reader = new BufferedReader(new FileReader(file));
            String line;
            while ((line = reader.readLine()) != null) {
                sb.append(line);
                sb.append(System.lineSeparator());
            }
            reader.close();
            String text = sb.toString();
            PreparedStatement pstmt = conn.prepareStatement( sql: "INSERT INTO TaskThemes (id_Task_theme, Task_theme) VALUES (?, ?)");
            pstmt.setString( parameterIndex: 1, id);
            pstmt.setString( parameterIndex: 2, text);
            pstmt.executeUpdate();
            pstmt.close();
        }

        conn.close();
    } catch (Exception e) {
        e.printStackTrace();
    }
```

Рисунок 21, лист 4



```
    private static String readFile(File file) throws Exception {
        BufferedReader reader = new BufferedReader(new FileReader(file));
        StringBuilder sb = new StringBuilder();
        String line;
        while ((line = reader.readLine()) != null) {
            sb.append(line);
            sb.append(System.lineSeparator());
        }
        reader.close();
        return sb.toString().trim();
    }

    2 usages
    private static byte[] readBytes(File file) throws Exception {
        if (!file.exists()) {
            return null;
        }
        byte[] data = new byte[(int) file.length()];
        java.io.FileInputStream fis = new java.io.FileInputStream(file);
        fis.read(data);
        fis.close();
        return data;
    }
}
```

Рисунок 21, лист 5

В данном коде представлен класс Main с методом main, который выполняет импорт данных из файлов в базу данных SQLite.

Первая часть кода относится к импорту данных о заданиях в таблицу "Tasks".

Сначала задаются пути к основной папке (mainDirectoryPath), базе данных (databasePath) и папке с текстами тем заданий (textDirectoryPath).

Затем происходит загрузка JDBC-драйвера SQLite с помощью Class.forName("org.sqlite.JDBC") [7].

После этого устанавливается соединение с базой данных с использованием DriverManager.getConnection("jdbc:sqlite:" + databasePath).

Создается Statement для выполнения SQL-запросов. С помощью stmt.executeUpdate() выполняются запросы для очистки таблиц "Tasks" и "TaskThemes".

После этого соединение закрывается с помощью stmt.close().

Затем создается новое соединение с базой данных.

Запускается цикл, который перебирает все папки в основной папке.

Для каждой папки проверяется, является ли она папкой (исключая файлы).

Получается идентификатор темы задания из имени папки и преобразуется в целое число.

Запускается вложенный цикл, который перебирает все папки внутри текущей папки.

Для каждой вложенной папки проверяется, является ли она папкой (исключая файлы).

Получается название вложенной папки и преобразуется в целое число, которое становится идентификатором задания.

Формируются пути к необходимым файлам (taskname.txt, taskdesc.txt, rightanswer.txt, answertext.txt, answerimage.png, taskimage.png) с использованием текущей вложенной папки.

Данные считываются из файлов с помощью метода readFile (читает текстовые файлы) и readBytes (читает бинарные файлы).

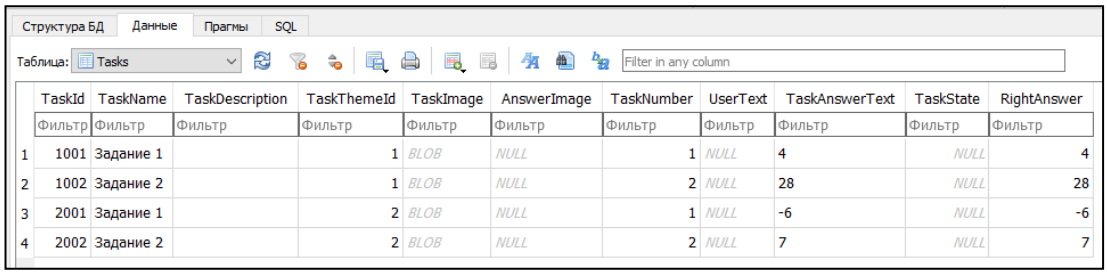
Далее создается PreparedStatement для выполнения запроса INSERT INTO и записи данных в таблицу "Tasks".

Устанавливаются параметры запроса с помощью методов setInt, setString и setBytes, а затем запрос выполняется с помощью pstmt.executeUpdate().

Внутренний цикл заканчивается, и переходим к следующей вложенной папке.

Внешний цикл заканчивается, и переходим к следующей папке.

После завершения перебора папок соединение с базой данных закрывается.А таблица Taskszаполняется (рисунок 22).



	TaskId	TaskName	TaskDescription	TaskThemeId	TaskImage	AnswerImage	TaskNumber	UserText	TaskAnswerText	TaskState	RightAnswer
	Фильтр	Фильтр	Фильтр	Фильтр	Фильтр	Фильтр	Фильтр	Фильтр	Фильтр	Фильтр	Фильтр
1	1001	Задание 1		1	BLOB	NULL	1	NULL	4	NULL	4
2	1002	Задание 2		1	BLOB	NULL	2	NULL	28	NULL	28
3	2001	Задание 1		2	BLOB	NULL	1	NULL	-6	NULL	-6
4	2002	Задание 2		2	BLOB	NULL	2	NULL	7	NULL	7

Рисунок 22 – Заполненная таблица Tasks

Вторая часть кода относится к импорту данных о темах заданий в таблицу "TaskThemes".

Снова загружается JDBC-драйвер SQLite.

Устанавливается соединение с базой данных.

Затем происходит перебор файлов в папке textDirectoryPath.

Для каждого файла извлекается его имя, из которого удаляются все символы, кроме цифр, с помощью replaceAll("[^0-9]", ""). Таким образом получается идентификатор темы задания.

Создается StringBuilder для считывания содержимого файла в память.

С помощью BufferedReader файл читается построчно, а каждая строка добавляется в StringBuilder с помощью sb.append(line).

После чтения файла он закрывается.

Содержимое файла преобразуется в строку с помощью sb.toString().

Затем создается PreparedStatement для выполнения запроса INSERT INTO и записи данных в таблицу "TaskThemes".

Устанавливаются параметры запроса с помощью методов setString, а затем запрос выполняется с помощью pstmt.executeUpdate().

После завершения перебора файлов соединение с базой данных закрывается. А таблица TaskThemesзаполняется (рисунок 23).

В конце кода обрабатываются исключения, которые могут возникнуть при выполнении операций чтения файлов или работы с базой данных. В случае

возникновения исключения они выводятся в консоль с помощью e.printStackTrace().

Код, заполняющий базу данных, представлен в приложенииА.

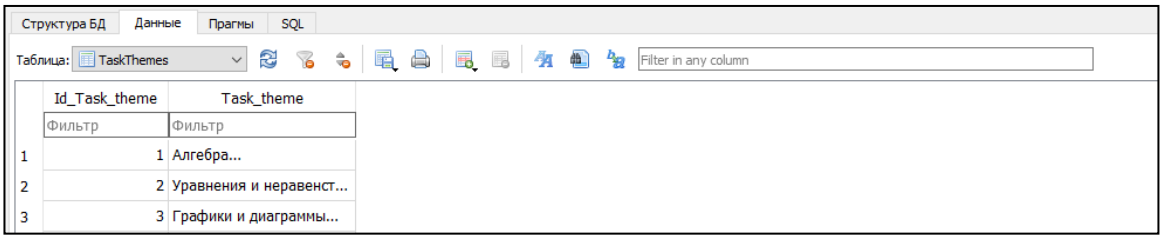


Рисунок 23 – Заполненная таблица TaskThemes

2.5 Автоматическая сборка элементов интерфейса по данным таблиц БД

Код, осуществляющий генерацию кнопок из тем заданий представлен на рисунке 24. Код класса TaskActivity.javапредставлен в приложении Б.

```
        for (Pair<Integer, String> taskButton : taskButtons) {
            Button button = new Button( context: this);
            button.setText(taskButton.second);
//            button.setBackgroundColor(Color.rgb(255, 165, 0));


            button.setBackground(getResources().getDrawable(R.drawable.rounded_button_background));;


            int width = getResources().getDisplayMetrics().widthPixels;
            int buttonWidth = (int) (width * 0.8);
            ViewGroup.LayoutParams layoutParams = new ViewGroup.LayoutParams(
                    buttonWidth, ViewGroup.LayoutParams.WRAP_CONTENT);

            button.setLayoutParams(layoutParams);
            button.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    // Открываем новое активити при нажатии на кнопку
                    Intent intent = new Intent( packageContext: TaskActivity.this, TaskDetailsActivity.class);
                    intent.putExtra( name: "id", taskButton.first);
                    intent.putExtra( name: "name", taskButton.second);
                    startActivity(intent);
                }
            });
            layout.addView(button);
            LinearLayout.LayoutParams layoutParams1 = (LinearLayout.LayoutParams) button.getLayoutParams();
// Устанавливаем отступ снизу на 16dp
            layoutParams1.setMargins( left: 0,  top: 26,  right: 0,  bottom: 0 );
        }
    }
}
```

Рисунок 24 – Программный код класса TaskActivity.java

Этот код представляет цикл, который выполняет следующие действия для каждого элемента списка taskButtons:

1. Создается новый экземпляр класса Button с использованием текущего контекста (this).

2. Устанавливается текст кнопки из значения taskButton.second.

3. Устанавливается фон кнопки с использованием ресурса R.drawable.rounded_button_background, который определен в файле ресурсов.

4. Вычисляется ширина экрана устройства с помощью метода getResources().getDisplayMetrics().widthPixels.

5. Вычисляется ширина кнопки, которая составляет 80% от ширины экрана (width * 0.8).

6. Создается объект ViewGroup.LayoutParams с заданными шириной кнопки и высотой WRAP_CONTENT.

7. Устанавливаются параметры макета кнопки с использованием созданного объекта layoutParams.

8. Устанавливается обработчик нажатия на кнопку, который открывает новую активность TaskDetailsActivity при нажатии кнопки. В Intent передаются данные о id и name, полученные из текущего элемента taskButton.

9. Кнопка добавляется в родительскую разметку (layout).

10. Создаются параметры макета кнопки в виде LinearLayout.LayoutParams.

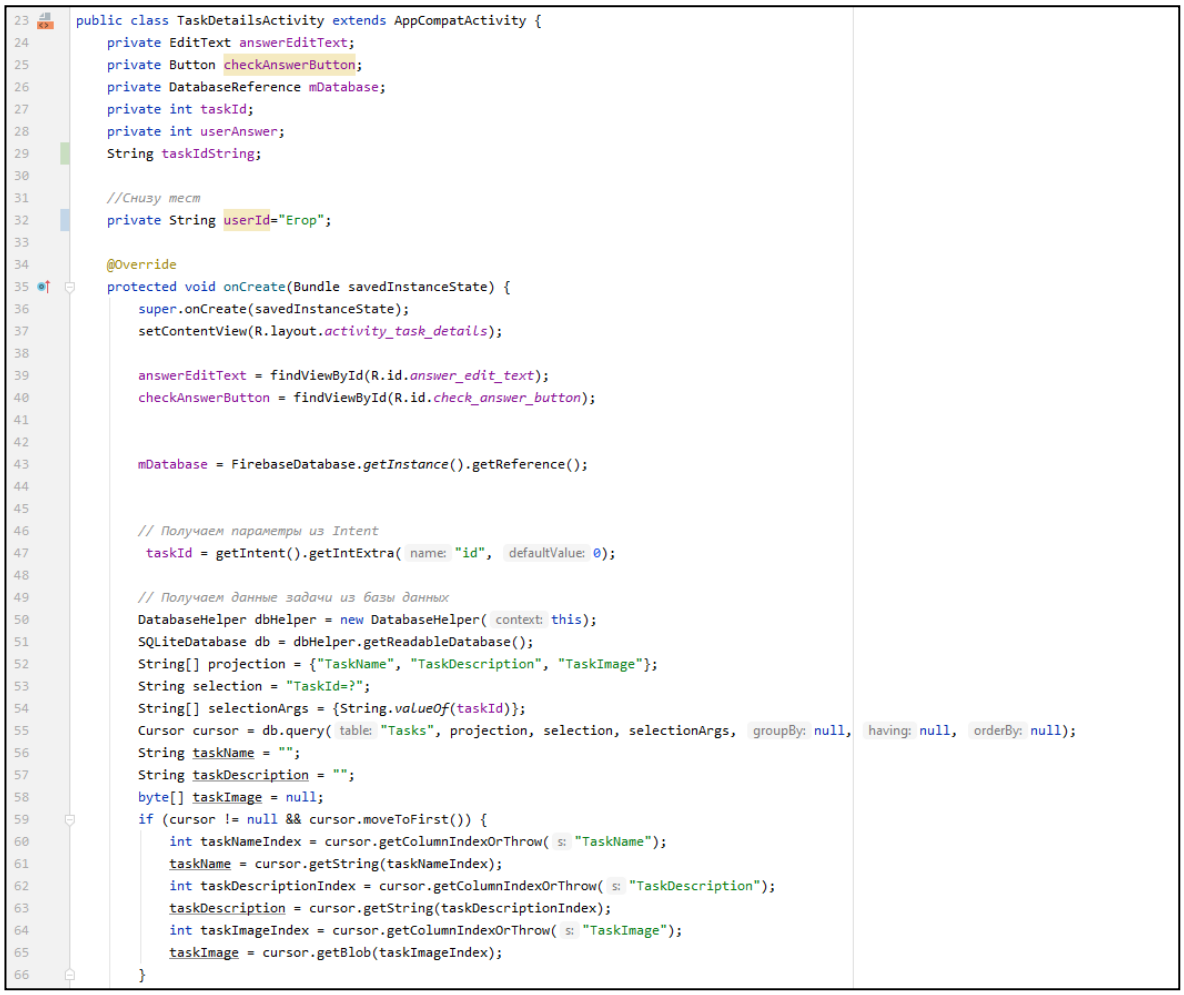
11. Устанавливается отступ снизу кнопки в размере 26 пикселей (layoutParams1.setMargins(0, 26, 0, 0)).

Таким образом, в результате выполнения цикла создается и добавляется несколько кнопок в указанную разметку, и каждая кнопка имеет уникальный текст и обработчик нажатия, который открывает новую активность и передает дополнительные данные (рисунок 25).

Рисунок 25 – Созданные кнопки заданий

2.5 Автоматическая сборка элементов задания в приложении по данным таблиц БД

Код, осуществляющий генерацию элементов задания по данным таблиц БД представлен на рисунке 26. Код класса TaskDetailsActivity.javaпредставлен в приложении В.



```
public class TaskDetailsActivity extends AppCompatActivity {
    private EditText answerEditText;
    private Button checkAnswerButton;
    private DatabaseReference mDatabase;
    private int taskId;
    private int userAnswer;
    String taskIdString;

    //Снизу тест
    private String userId="Erop";

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

        answerEditText = findViewById(R.id.answer_edit_text);
        checkAnswerButton = findViewById(R.id.check_answer_button);


        mDatabase = FirebaseDatabase.getInstance().getReference();


        // Получаем параметры из Intent
         taskId = getIntent().getIntExtra( name: "id",  defaultValue: 0);

        // Получаем данные задачи из базы данных
        DatabaseHelper dbHelper = new DatabaseHelper( context: this);
        SQLiteDatabase db = dbHelper.getReadableDatabase();
        String[] projection = {"TaskName", "TaskDescription", "TaskImage"};
        String selection = "TaskId=?";
        String[] selectionArgs = {String.valueOf(taskId)};
        Cursor cursor = db.query( table: "Tasks", projection, selection, selectionArgs,  groupBy: null,  having: null,  orderBy: null);
        String taskName = "";
        String taskDescription = "";
        byte[] taskImage = null;
        if (cursor != null && cursor.moveToFirst()) {
            int taskNameIndex = cursor.getColumnIndexOrThrow( s: "TaskName");
            taskName = cursor.getString(taskNameIndex);
            int taskDescriptionIndex = cursor.getColumnIndexOrThrow( s: "TaskDescription");
            taskDescription = cursor.getString(taskDescriptionIndex);
            int taskImageIndex = cursor.getColumnIndexOrThrow( s: "TaskImage");
            taskImage = cursor.getBlob(taskImageIndex);
        }
```

Рисунок 26–Программный код класса TaskActivity.java, лист1

```
        if (cursor != null) {
            cursor.close();
        }

        // Отображаем данные задачи
        TextView taskNameView = findViewById(R.id.task_name);
        taskNameView.setText(taskName);
        TextView taskDescriptionView = findViewById(R.id.task_description);
        taskDescriptionView.setText(taskDescription);
        ImageView taskImageView = findViewById(R.id.task_image);
        if (taskImage != null) {
            Bitmap bitmap = BitmapFactory.decodeByteArray(taskImage,  offset: 0, taskImage.length);
            taskImageView.setImageBitmap(bitmap);
        }
        else {
            taskImageView.setVisibility(View.GONE);
            Log.d( tag: "TaskDetailsActivity",  msg: "No image found for taskId=" + taskId);
        }




        checkAnswerButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                 userAnswer = 0;
                try {
                    userAnswer = Integer.parseInt(answerEditText.getText().toString());
                } catch (NumberFormatException e) {
                    Toast.makeText( context: TaskDetailsActivity.this,  text: "Введите число", Toast.LENGTH_SHORT).show();
                    return;
                }
```

Рисунок 26, лист 2

```
            // Получаем параметры из Intent
            taskId = getIntent().getIntExtra( name: "id",  defaultValue: 0);
            taskIdString = String.valueOf(taskId);

            // Получаем данные задачи из базы данных
            DatabaseHelper dbHelper = new DatabaseHelper( context: TaskDetailsActivity.this);
            SQLiteDatabase db = dbHelper.getReadableDatabase();
            String[] projection = {"RightAnswer"};
            String selection = "TaskId=?";
            String[] selectionArgs = {String.valueOf(taskId)};
            Cursor cursor = db.query( table: "Tasks", projection, selection, selectionArgs,  groupBy: null,  having: null,  orderBy: null);
            int rightAnswer=0;
            if (cursor != null && cursor.moveToFirst()) {
                rightAnswer = cursor.getInt(cursor.getColumnIndexOrThrow( s: "RightAnswer"));
            }
            if (cursor != null) {
                cursor.close();
            }

            Log.d( tag: "TaskDetailsActivity",  msg: "userAnswer = " + userAnswer);
            Log.d( tag: "TaskDetailsActivity",  msg: "rightAnswer = " + rightAnswer);
            // Сравниваем ответ пользователя и правильный ответ
            if (userAnswer == rightAnswer) {
                Toast.makeText( context: TaskDetailsActivity.this,  text: "Верно!", Toast.LENGTH_SHORT).show();
            } else {
                Toast.makeText( context: TaskDetailsActivity.this,  text: "Неверно!", Toast.LENGTH_SHORT).show();
            }

            ContentValues values = new ContentValues();
            values.put("UserText", userAnswer);
            int count = db.update( table: "Tasks", values, selection, selectionArgs);
        /*  if (count > 0) {
                Toast.makeText(TaskDetailsActivity.this, "Ответ сохранен!", Toast.LENGTH_SHORT).show();
            } else {
                Toast.makeText(TaskDetailsActivity.this, "Ошибка сохранения ответа!", Toast.LENGTH_SHORT).show();
            }*/

            writeNewUser();
        }
    });

```

Рисунок 26, лист 3

```

        TextView myTextView = findViewById(R.id.bottom_text_view);
        myTextView.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                // переход к новой активности
                Intent intent = new Intent( packageContext: TaskDetailsActivity.this, TaskAnswer.class);
                intent.putExtra( name: "id", taskId);
                startActivity(intent);
            }
        });
    }

    public void writeNewUser() {
        User user = new User( userId: 1,taskId,userAnswer);


        mDatabase.child("users").child(userId).child(taskIdString).setValue(user);
    }
}

```

Рисунок 26, лист 4

В данном коде представлен класс TaskDetailsActivity, который отображает подробности задачи и позволяет пользователю проверить свой ответ на задачу.

В методе onCreate происходит инициализация различных элементов пользовательского интерфейса, таких как текстовые поля и кнопки.

Затем устанавливается соединение с базой данных Firebase и получается идентификатор задачи (taskId) из параметров Intent.

Далее создается объект DatabaseHelper, который используется для работы с базой данных SQLite.

Выполняется запрос к базе данных, чтобы получить данные задачи (название, описание, изображение). Результат запроса сохраняется в переменные taskName, taskDescription и taskImage.

Данные задачи отображаются на экране в соответствующих элементах пользовательского интерфейса (рисунок 27).

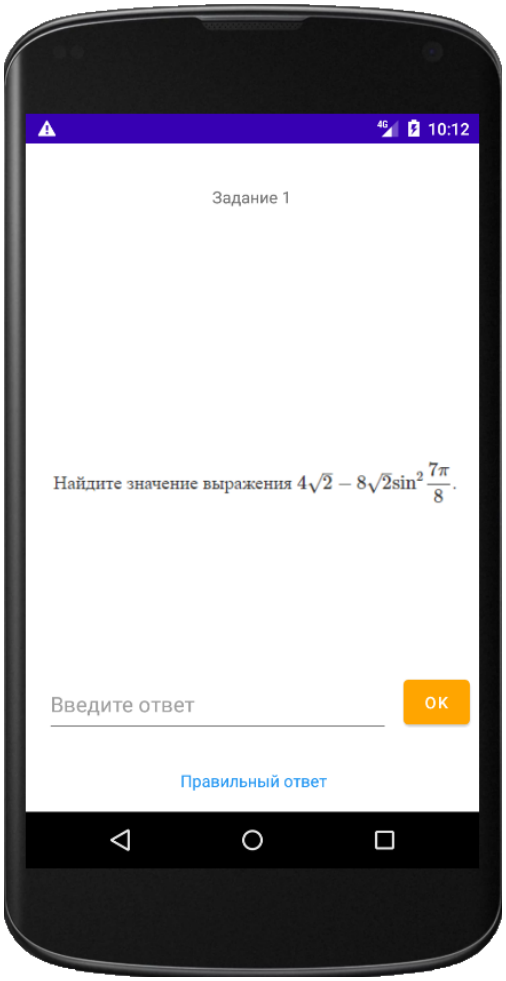


Рисунок 27 – Данные задачи

Затем устанавливается обработчик нажатия на кнопку "Проверить ответ".

При нажатии на кнопку происходит считывание ответа пользователя из текстового поля. Если введено некорректное значение (не число), выводится сообщение об ошибке.

Далее выполняется запрос к базе данных, чтобы получить правильный ответ на задачу. Результат запроса сохраняется в переменную rightAnswer.

Ответ пользователя и правильный ответ выводятся в консоль для отладки.

Проверяется, совпадает ли ответ пользователя с правильным ответом. В зависимости от результата выводится соответствующее сообщение.

Затем обновляется запись в базе данных, чтобы сохранить ответ пользователя (рисунок 28 и рисунок 29).

Рисунок 28 – Уведомление о неправильном ответе

Рисунок 29 – Уведомление о правильном ответе

Также вызывается метод writeNewUser(), который записывает информацию о пользователе и его ответе в базу данных Firebase.

В методе writeNewUser() создается объект User с информацией о пользователе (идентификатор пользователя, идентификатор задачи, ответ пользователя). Затем информация записывается в базу данных Firebase (рисунок 30) [8].

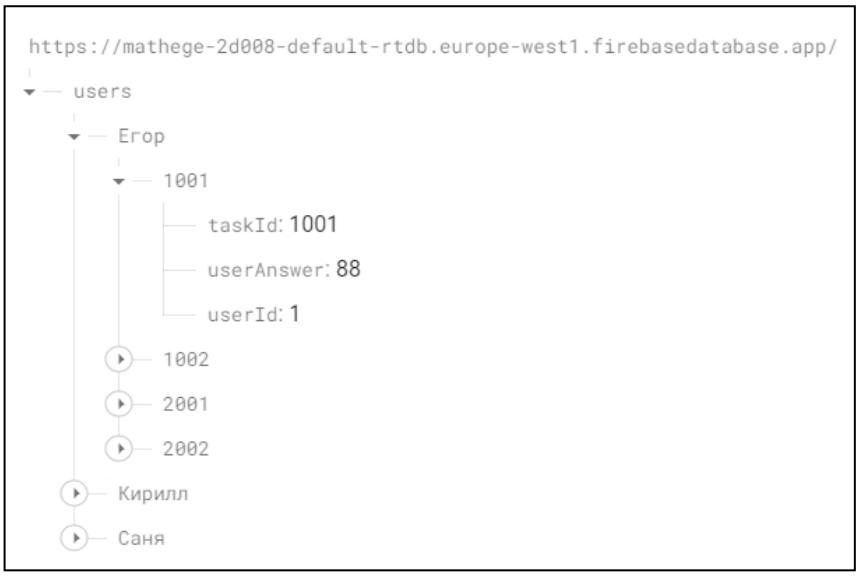


Рисунок 30 – Структура базы данных Firebase

2.6 Отправка данных FireBase

Код, осуществляющий работу с базой данной Firebase в реальном времени, представлен на рисунке 31.

```
package com.dragikgames.mathege;

public class User {

    int userId, taskId, userAnswer;

    public User(int userId, int taskId, int userAnswer) {
        this.userId = userId;
        this.taskId = taskId;
        this.userAnswer = userAnswer;
    }


    public int getuserId() { return userId; }

    public void setuserId(int id) { this.userId = userId; }

    public int getTaskId() { return taskId; }

    public void setTaskId(int taskId) { this.taskId = taskId; }

    public int getUserAnswer() { return userAnswer; }

    public void setUserAnswer(int userAnswer) { this.userAnswer = userAnswer; }
}
```

Рисунок 31 – Программный код класса User.java

Этот код представляет класс "User" (пользователь), который имеет три переменные экземпляра: userId (идентификатор пользователя), taskId (идентификатор задания) и userAnswer (ответ пользователя).

Конструктор класса "User" принимает три аргумента: userId, taskId и userAnswer. Он используется для инициализации переменных экземпляра класса при создании объекта типа "User".

У класса "User" также есть геттеры и сеттеры для каждой переменной экземпляра. Геттеры используются для получения значений переменных, а сеттеры - для установки новых значений.

Например, метод "getuserId()" возвращает значение переменной "userId", метод "setuserId()" устанавливает новое значение переменной "userId", метод "getTaskId()" возвращает значение переменной "taskId", а метод "setTaskId()" устанавливает новое значение переменной "taskId". Аналогично для переменной "userAnswer".

Этот класс может быть использован для создания объектов, представляющих пользователей и хранения их данных, таких как идентификатор пользователя, идентификатор задания и ответ пользователя.

Выводы по разделу «Разработка приложения ″Тренажер ЕГЭ по математике″»

В разделе 2 описана разработка мобильного приложения «Тренажер ЕГЭ по математике», включая разработку интерфейса приложения, реализацию логики приложения, заполнение базы данных заданий. Описан процесс автоматической сборки элементов интерфейса по данным таблиц базы данных и автоматическая сборка элементов задания в приложении. Рассмотрена отправка данных в FireBase.

3 Оценка затрат реализации проекта

3.1 Анализ состава и стоимости ресурсов необходимых для реализации проекта

Разработчику приложения придется выступить в нескольких ролях:

1) роль разработчика – программиста, который отвечает за backend-составляющую, написание кода и прочее. Эта работа будет вестись на протяжении всей разработки, так как после тестирования потребуется исправить выявленные крупные баги;

2) роль дизайнера который отвечает за творческую составляющую приложения, за создание пользовательского интерфейса;

3) роль тестировщика, который будет проверять работоспособность, оцениватьпредрелизную версию со стороны рядового пользователя и выявлять пробелы, на учтенные на первых этапах.

Для разработкиbackend-составляющей не потребуется дорогой компьютер, а в частности монитор, так как он по большей части работает с кодом и интегрированной средой разработки.

В то же время при создании интерфейсапонадобится хорошее оборудование для визуализации результата. Важно иметь монитор с большим разрешением и матрицей, для полной передачи картинки и создания дизайна.

Перечень необходимого оборудования программиста-разработчика представлен в таблице 2.

Таблица 2 – Перечень оборудования программиста-разработчика

Наименование	Стоимость, руб	Дополнительные сведения	Сроки использования
Процессор AMD Ryzen 3 1200 BOX	7 799	AM4 3.1 ГГц x 4	Все время разработки
Материнская плата GIGABYTE B450M	4 199	AM4	Все время разработки

Окончание таблицы 2

Корпус DEXP DC-201M	1 699	Черный	Все время разработки
MSI GeForce GTX 1650	29 999	4 G	Все время разработки
Кулер для процессора ID-COOLING DK-03 RAINBOW	799	1800 об/ мин	Все время разработки
Оперативнаяпамять AMD Radeon R7 Performance Series	2 999	8 ГБ	Все время разработки
Жесткийдиск Toshiba P300	2 999	500 ГБ	Все время разработки
Блок питания AeroCoolVXPLUS	2 399	550 W	Все время разработки
Монитор LG 24MK430H черный	12 199	23.8"	Все время разработки
Клавиатура проводная Oklick 90M	299	USB	Все время разработки
Мышь беспроводная Ritmix RMW-555 серый	150	1600 Dpi	Все время разработки

Общая стоимость: 65 540 руб.

Из программных средств в первую очередь необходимо иметь ОС для компьютера. Так же необходимо иметь AndroidStudio– интегрированную среду разработки приложений на устройствах с ОС Android. Её пользование бесплатное. То есть программа является бесплатной до тех пор, пока созданное в ней приложение не начнет приносить указанный доход. Подписка Creativecloud для возможности пользования Photoshop обойдется в 1622,40 Р / месяц. Git бесплатен и необходим для совместной работы над кодом проекта. Также не менее важно антивирусное ПО.

Таблица 3 – Затраты на ПО

Наименование	**Стоимость, руб**	**Срок использования, месяц**
Операционная система MicrosoftWindows 11 Pro	12 999	Все время разработки
Unity 3Dcommunity	0	1
VisualStudio	0	1
Blender	0	1

Окончание таблицы 3

CreativeCloud (Photoshop), вмесяц	1622,40 / месяц	1
Git	0	1
Kaspersky antivirus, в год	1799/год	1

3.2 Расчет проектных затрат

Затраты на проектирование рассчитываются по следующей формуле

$$К_{пр} = К_{зп} + К_{ипс} + К_{свт} + К_{проч}, \quad (1)$$

где $К_{зп}$ – затраты на заработную плату проектировщика;

$К_{ипс}$ – затраты на инструментальные программные средства;

$К_{свт}$ – затраты на средства вычислительной техники;

$К_{проч}$ – прочие затраты на проектирование.

Оклад сотрудникасоставляет 16242 рубля (МРОТ с 1 января 2023 г.) без учета районного коэффициента (30%) и северной надбавки (30%).

Расчет зарплаты с учетом региональной надбавки и северного коэффициента будет: 16242 р. * 1.6 = 25987р.

Разработка проекта будет длиться 1 месяц.

НДФЛ составит: 25987руб * 0.13 = 3379 рублей

По закону имеется обязанность уплаты страховых взносов (п. 2 ст. 6 Закона от 16.07.1999 № 165-ФЗ). Отчисления во внебюджетные фонды составляют 30.2%. Затраты составят: 25987 * 0.302 = 7848 руб.

Итого $К_{зп}$= 25987руб + 7848руб = 33835 руб.

Затраты на оборудование составляют 65 540 руб.

Амортизация будет рассчитываться по линейному методу. Годовой процент амортизации составляет 25%. Тогда ежегодная сумма 16 385 руб. Тогда сумма амортизации за месяц:16 385 / 12 = 1 366 Р.

Итого $К_{свт}$ = 1 366 руб.

Лицензия программы Photoshopна 1 месяц использования будет стоить1622руб. Антивирусное ПО берется только на год.

Аналогично амортизация будет рассчитываться по линейному методу. Для лицензий Windows 11 годовой процент амортизации 25%. Ежегодная сумма = 12 999 руб * 0.25 = 3 250 руб. Ежемесячная сумма составит: 3 250руб / 12 = 271 руб. Использоваться в данном проекте лицензия будет 1 месяц. Для лицензии антивирусного ПО необходимо рассчитать месячную сумму: 1799руб / 12 мес = 150 руб.

Итого $К_{ипс}$ = 271 руб + 150руб = 421 руб.

Прочие затраты составляют 3% от расходов наПО, заработную плату и вычислительную технику.

$К_{проч}$ = (33 835 руб + 1 366 руб +421 руб) *0.03 = 1 069 руб.

Проектные затраты приведены на рисунке 32.

В таблице 4 приводится полный перечень проектных затрат.

Таблица 4 – Структура проектных затрат

Затраты	**Сумма, руб.**
$К_{зп}$	33 835
$К_{свт}$	1 366
$К_{ипс}$	421
$К_{проч}$	1 069
Итого	36 691

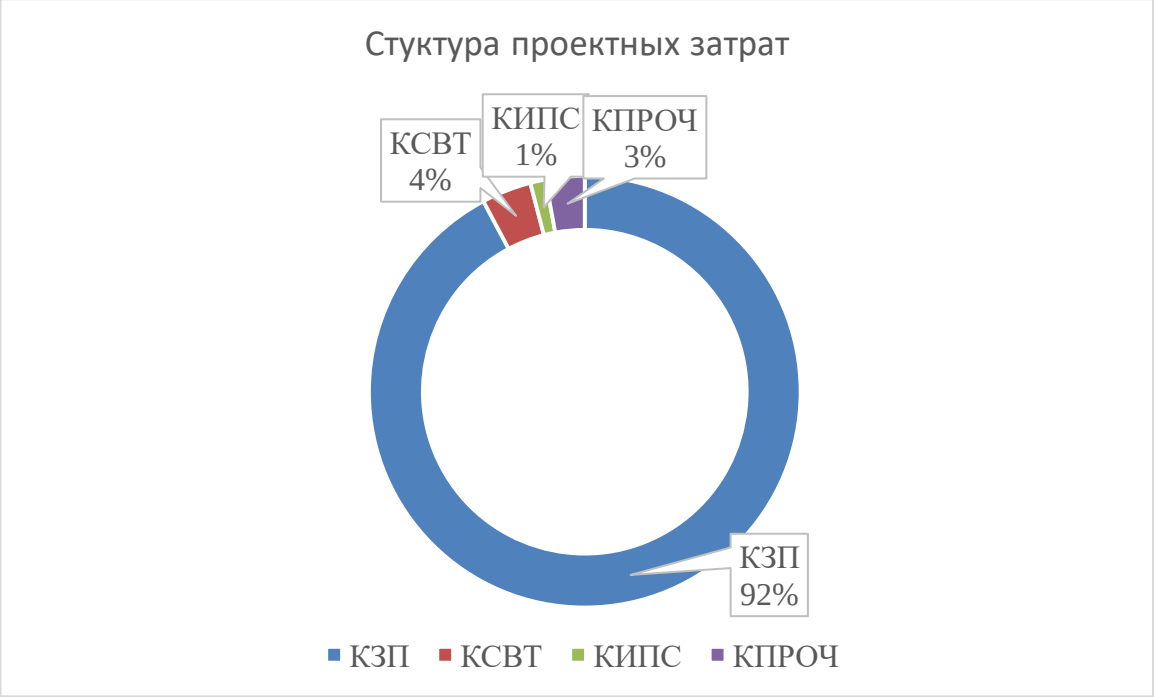


Рисунок 32 – Струкутра проектных затрат

3.3 Расчет капитальных затрат

Капитальные затраты на разработку информационной системы вычисляются по формуле

$$К = К_{тс} + К_{лс} + К_{по} + К_{ио} + К_{оэ}, \quad (2)$$

где $К_{тс}$ – затраты на технические средства;

$К_{лс}$ – затраты на создание линий связи локальных сетей;

$К_{по}$ – затраты на программные средства;

$К_{ио}$ – затраты на формирование информационной базы;

$К_{оэ}$ – затраты на опытную эксплуатацию.

На этапе внедрения необходимо будет использовать имеющийся компьютер. Поскольку этап внедрения будет длиться 10 рабочих дней, расчитаем аммортизацию используемого ПК на этот период:1 366 руб. / 21 * 10

дней = 651 руб. Дальнейшая работа создаваемой ИС не учитывается, так как это зависит от оборудования пользователя.

$К_{тс}$ = 651 руб.

Поскольку приложение будет внедряться на онлайн-сервисы цифрового распространения мобильных программ, для нее не потребуются линии связи локальных сетей. Но на этапе внедрения будет необходим доступ в интернет. Месячный тариф составит 650 руб.

$К_{лс}$ = 650 руб.

Размещаться приложение будет на площадке Playmarket. Для Playmarket нужно внесение разовой оплаты за соглашение разработчика в размере 25$. Это примерно2000 руб.

$К_{по}$ = 2000 руб.

Внедрение информационной базы SQlite является бесплатным

$К_{ио}$ = 0 руб.

На этапе внедрения будет добавлена тестовая ранняя версия приложения, которую любой пользователь сможет бесплатно попробовать и оценить. На исправление найденных ошибок необходимо будет выделить около 10 часов рабочего времени программиста. 155 руб * 10 часов * 1,302 = 2018 рублей.

$К_{оэ}$= 2018 руб.

Капитальные затраты приведены на рисунке 33.

Полный перечень капитальных затрат приведен в таблице 5.

Таблица 5 – Перечень капитальных затрат

Затраты	**Сумма, руб.**
$К_{пр}$	36 691
$К_{тс}$	651
$К_{лс}$	650
$К_{по}$	2 000
$К_{ио}$	0
$К_{оэ}$	2 018
Итого	42 010

Рисунок 33 – Капитальные затраты

3.4 Расчет эксплуатационных затрат

Формула подсчета эксплуатационных затрат проводится по формуле

$$C = C_{зп} + C_{ао} + C_{то} + C_{лс} + C_{ни} + C_{проч}, \quad (3)$$

где $C_{зп}$ – зарплата персонала, работающего с информационной системой;

$C_{ао}$ – амортизационные отчисления;

$C_{то}$ – затраты на техническое обслуживание;

$C_{лс}$ – затраты на использование глобальных сетей;

$C_{ни}$ – затраты на носители информации;

$C_{проч}$ – прочие затраты.

Зарплата персонала, работающего с информационной системой.

Приложение будет поддерживаться различными обновлениями, патчами и исправлениями, потому будет необходим программист. В месяц потребуется 5 часов работы, соответственно за год он отработает 60 часов.

155 руб * 60 часов * 1,302 = 12109 руб. – затраты на зарплату программиста за год.

$С_{зп}$ = 12109 руб.

Амортизационные отчисления.

Программистом будет использоваться компьютер на протяжении его рабочего времени. С учетом расчетов амортизации эта величина составит 488 руб.

$С_{ао}$ = 488 руб.

Затраты на техническое обслуживание

Техническое обслуживание не будет нужно, поскольку компьютер будет использоваться очень малое количество времени. Эти незначительные затраты учтены в прочих.

$С_{то}$ = 0 руб.

Затраты на использование глобальных сетей

Для работы программиста над обновлениями проекта будет необходим кратковременный доступ в Интернет. Незначительные затраты будут учтены в прочих.

$С_{лс}$= 0 руб.

Затраты на носители информации

Все данные и само приложение будут храниться на площадках, где опубликовано приложение. Затраты учтены на этапе внедрения.

$С_{ни}$ = 0 руб.

Прочие затраты

Затраты на электроэнергию и прочее составят 3% от предыдущих эксплуатационных затрат.

$С_{проч}$= (12 109руб + 488руб) * 0.03 = 378 руб.

Итого эксплуатационные зарплаты составят: С = 12 109руб + 488руб + 378 руб = 12 975 руб.

Эксплуатационные затраты приведены на рисунке 34.

Структура эксплуатационных затрат приведена в таблице 6.

Таблица 6по центру– Структура эксплуатационных затрат

Затраты	Сумма,руб.
$С_{зп}$	12 109
$С_{ао}$	488
$С_{то}$	0
$С_{лс}$	0
$С_{ни}$	0
$С_{проч}$	378
Итого	12 975

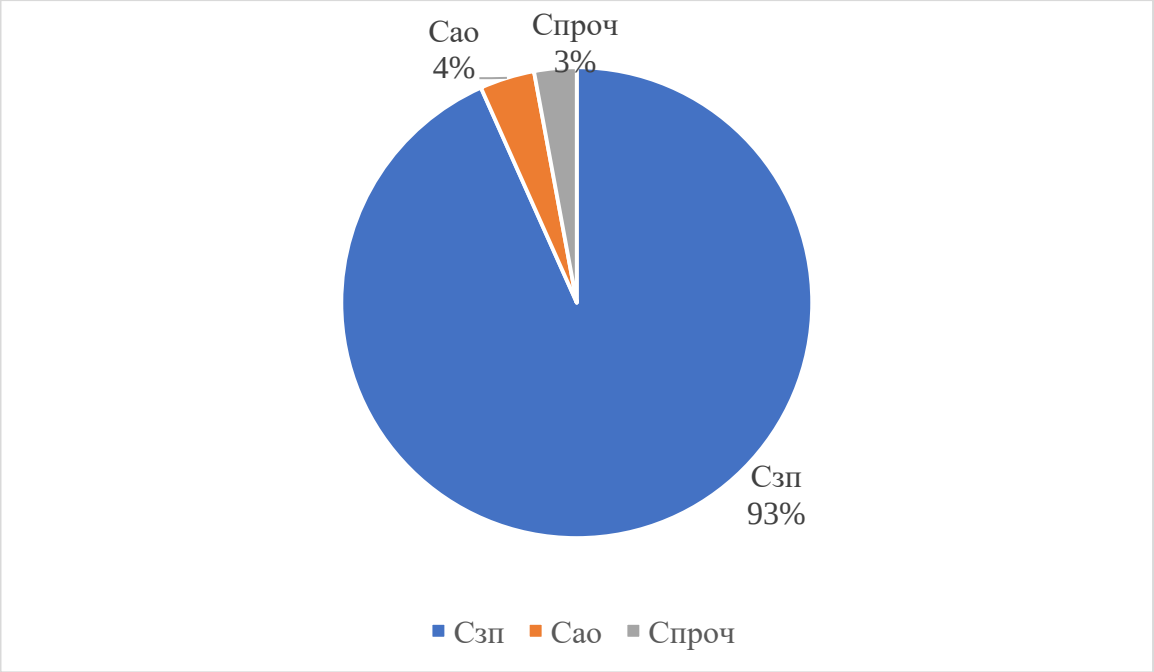


Рисунок 34 – Структура эксплуатационных затрат

Больше всего составили затраты на зарплату для программиста, который будет заниматься обновлениями.

3.5 Расчет совокупной стоимости владения системой

Показатель совокупной стоимости владения ИС рассчитывается по формуле

$$TCO = DE + IC, \quad (4)$$

гдеDE (directexpenses) – прямыерасходы

IC - (indirectcosts) – косвенные расходы

Прямые расходырассчитываются по формуле

$$DE = DE1 + DE2 + DE3 + DE4 + DE5 + DE6 + DE7 + DE8, \quad (5)$$

где DE1 - капитальные затраты, которые берутся из предыдущих расчетов, DE1 = 42 010руб;

DE2 - расходы на управление ИТ. Проект размещен на площадках и является автономным, но требуются расходы на актуализацию обновлений. DE2 = 12 109руб;

DE3 - расходы на техническую поддержку АО и ПО, то есть Сао + Сто = 488 руб. в год;

DE4 - расходы на разработку прикладного ПО внутренними силами. Для данного приложения использовалось только существующее ПО и разработка нового не была необходима. DE4 = 0 руб;

DE5 - расходы на аутсорсинг. Данному проекту не нужен ни один из видов аутсорсинга. DE5 = 0 руб;

DE6 - командировочные расходы. Данный вид расходов не предусмотрен, поскольку командировки отсутствуют, потому DE6 = 0 руб;

DE7 - расходы на услуги связи. Будет необходим кратковременный доступ к сети Интернет, потому данный тип расходов учтён в DE8;

DE8 - другие группы расходов, обычно составляет около 1% от остальных. DE8 = 546 руб.

Косвенные затраты.Оценка рисков.

Реализация такого проекта как мобильное приложение сопровождается некоторыми рисками, которые могут повлиять на дальнейшую судьбу проекта.

Можно выделить следующий перечень рисков для данного проекта:

1. Невостребованность приложения среди целевой аудитории.
2. Риск технической реализации.
3. Увеличение планируемых расходов.
4. Нарушение авторских прав.

В таблице 7 представлены возможные риски созданного проекта.

Таблица 7 – Риски проекта

Риск	**Уровень влияния**	**Вероятность**	**Возможность предотвращения**
Невостребованность приложения среди целевой аудитории	Высокий	Средняя	Проработка хорошего дизайна и скорости работы приложения
Риск технической реализации	Средний	Низкая	Тщательное тестирование на разных конфигурациях, создание системы отчетов об ошибках
Увеличение планируемых расходов	Средний	Средняя	Оформление дополнительного договора
Нарушение авторских прав	Средний	Средняя	Отказ от использования готовых разработок (модели, рисунки)

Возможность невостребованности приложения среди целевой аудитории имеет высокий уровень влияния, потому как в случае его возникновения, проект не будет полноценно реализован. Вероятность такого события определить трудно, но это и не столь важно, если целью является продемонстрировать целевой аудитории (абитуриентам) возможности того, чем бы они могли научиться в ХТИ – филиале СФУ.

Финансовые риски. Во избежание подобного риска, нужно оформлять дополнительный денежный договор, в котором будут учитываться все

непредвиденные расходы, которые могут часто возникнуть при разработке приложения.

Нарушение авторских прав – очень редкий риск и имеет шанс произойти только при большой популярности продукта. Суд за нарушение авторских прав это трудоемкий процесс. К тому же, взыскиваемая компенсация может составлять 5 миллионов и меньше, но поскольку приложением в таком случае могут заинтересоваться только если оно станет популярным, то к этому времени заработок будет превышать эту сумму. Тем не менее, следует учитывать данный риск и стараться не использовать наработки, модели, картинки, музыку с авторскими правами.

Структура ТСО приведена в таблице 8.

Таблица 8 – Структура ТСО

Вид расхода	**Цена, руб.**
DE1	42 010
DE2	12 109
DE3	488
DE4	0
DE5	0
DE6	0
DE7	0
DE8	546
IC	1 655
Итого	56808

Итого ТСО = (42 010 + 12 109+ 488 + 546) + 1 655 = 56 808 руб. Капитальные затраты имеют большую долю, это обуславливается тематикой проекта – мобильное приложение.

Прямые расходы приведены на рисунке 35.

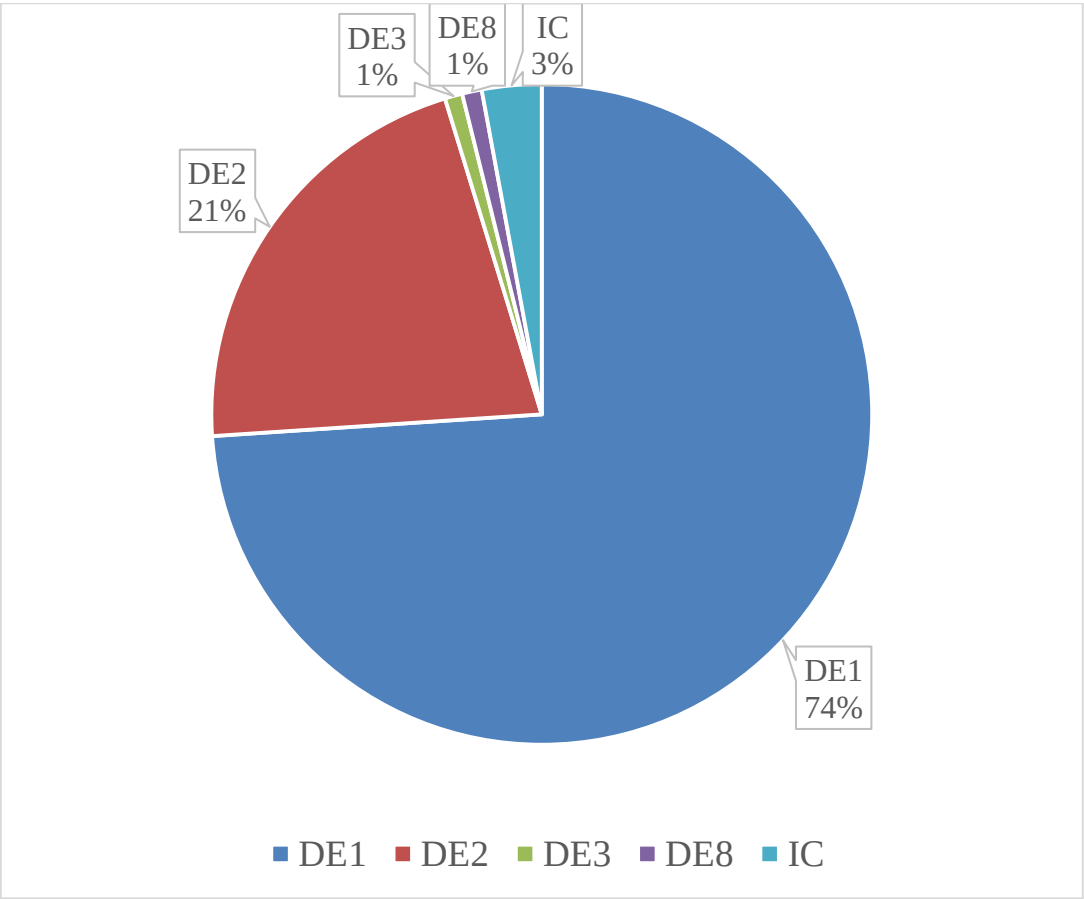


Рисунок 35 – Прямые расходы

Выводы по разделу «Оценка затрат реализации проекта»

В разделе 3выполнена оценка затрат реализации проекта. Данная оценка включает анализ состава и стоимости ресурсов, необходимых для реализации проекта. Произведены расчеты проектных, капитальных и эксплуатационных затрат.

Рассмотрена совокупная стоимость владения системой, учитывающая как проектные, так и эксплуатационные затраты. Капитальные затраты составили 42010 рублей, эксплуатационные – 12975 рублей, совокупная стоимость владения продуктом – 56808 рублей.

ЗАКЛЮЧЕНИЕ

В результате выполнения работы по разработке мобильного приложения для подготовки к единому государственному экзамену по математике был проведен анализ предметной области деятельности ХТИ – филиала СФУ.

Была разработана концепция IT-проекта, проанализированы похожие программныепродукты и обоснована необходимость собственной разработки, поскольку продукты конкурентов не подходили под особенности процесса.

Для определения всех особенностей проекта было выполнено структурное моделирование бизнес-процесса.

Модель потоков данных позволяет проследить, как движется информация внутри системы, и взаимодействие системы с внешними сущностями. Определяются основные процессы внутри системы и внешние сущности, потоки между ними, а также хранилища данных.

Для хранения информации была спроектирована база данных. Поскольку процесс выполнения заданий требует только ответов от пользователя, то база данных получилась небольшой, но эффективной и достаточно гибкой для добавления новых критериев или объектов для модификаций приложения.

Перед разработкой был обоснован выбор средств разработки AndroidStudio. Были оценены самые популярные решения и выбраны самые эффективные для данного проекта.

В период преддипломной практики создано мобильное приложение «Тренажер ЕГЭ по математике». Разработан программный код для заполнения базы данных заданий.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Хакасский технический институт - филиал федерального государственного автономного образовательного учреждения высшего образования «Сибирский федеральный университет» : официальный сайт. – Абакан, 2023. – URL: https://khti.ru (дата обращения: 26.02.2023)

2. Правила построения диаграмм потоков данных DFD // harrix.dev : [сайт]. – URL: https://harrix.dev/blog/2017/exist-sqlite-android-studio/ (дата обращения: 22.02.23).

3. Методология UML-проектирования // lektsii.org : [сайт]. – URL: https://lektsii.org/18-50974.html (дата обращения 21.02.23).

4. Сравнительная характеристика Android IDE // infourok.ru : [сайт]. – URL: https://infourok.ru/vibor-sredi-razrabotki-dlya-sozdaniya-android-prilozheniy-3028586.html (дата обращения: 23.02.23).

5. РуководствоAndroidStudionaJava // metanit.com : [сайт]. – 2022 – URL: https://metanit.com/java/android/2.2.php (датаобращения: 22.02.2023)

6. Подключение существующей БД SQLite в AndroidStudio // harrix.dev : [сайт]. – URL: https://harrix.dev/blog/2017/exist-sqlite-android-studio/ (дата обращения: 22.02.23).

7. Основы JDBC // oracle.com: [сайт]. – URL: https://docs.oracle.com/javase/tutorial/jdbc/basics/index.html (дата обращения: 25.02.23).

8. Чтение и запись данных в Firebase // google.com: [сайт]. – URL: https://firebase.google.com/docs/database/android/read-and-write#java (дата обращения: 26.02.23).

ПРИЛОЖЕНИЕА

Код, заполняющий базу данных. Документjava

```
import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.Statement;
public class Main {
public static void main(String[] args) {
String mainDirectoryPath= "D:\\MathEGE\\DataBase\\Tasks"; // путькосновнойпапке
String databasePath= "C:\\Program Files\\DB Browser for SQLite\\EGE_math.db"; //
путькбазеданных
String textDirectoryPath= "D:\\MathEGE\\DataBase\\TaskThemes";
try {
Class.forName("org.sqlite.JDBC");
Connection connn= DriverManager.getConnection("jdbc:sqlite:" + databasePath);
Statement stmt= connn.createStatement();
stmt.executeUpdate("DELETE FROM Tasks");
stmt.executeUpdate("DELETE FROM TaskThemes");
stmt.close();
Connection conn = DriverManager.getConnection("jdbc:sqlite:" + databasePath);
// Перебираемвсепапкивосновнойпапке
File mainDirectory= new File(mainDirectoryPath);
File[] taskDirectories= mainDirectory.listFiles();
for (File taskDirectory: taskDirectories) {
if (!taskDirectory.isDirectory()) {
continue;
                }
// Получаемназваниепапки
String taskThemeId= taskDirectory.getName();
inttaskThemeIdInt= Integer.parseInt(taskThemeId);

// Перебираемвсепапкивнутритекущейпапки
File[] subDirectories= taskDirectory.listFiles();
for (File subDirectory: subDirectories) {
if (!subDirectory.isDirectory()) {
continue;
```

```
                    }

// Получаемназваниевложеннойпапки
String subDirectoryName= subDirectory.getName();
inttaskId= Integer.parseInt(subDirectoryName);
// Ищемнужныефайлы



//                    File[] tfiles = subDirectory.listFiles();


File taskNameFile= new File(subDirectory, "taskname.txt");
File taskDescFile= new File(subDirectory, "taskdesc.txt");
File rightAnswerFile= new File(subDirectory, "rightanswer.txt");
File answerTextFile= new File(subDirectory, "answertext.txt");
File answerImageFile= new File(subDirectory, "answerimage.png");
File taskImageFile= new File(subDirectory, "taskimage.png");

// Извлекаемданныеизфайлов
                    // String taskId = readFile(taskIdFile);
inttaskNumber= taskId- 1000*taskThemeIdInt;
String taskname= readFile(taskNameFile);
String taskdesc= readFile(taskDescFile);
String rightanswer=readFile(rightAnswerFile);
String answertext= readFile(answerTextFile);
byte[] answerimage= readBytes(answerImageFile);
byte[] taskimage= readBytes(taskImageFile);

// Записываемданныевбазуданных
PreparedStatementpstmt= conn.prepareStatement("INSERT INTO Tasks (TaskId,
TaskName, TaskDescription, TaskAnswerText, AnswerImage, TaskImage, TaskThemeId,
TaskNumber, RightAnswer) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?)");
pstmt.setInt(1, taskId);
pstmt.setString(2, taskname);
pstmt.setString(3, taskdesc);
pstmt.setString(4, answertext);
pstmt.setBytes(5, answerimage);
pstmt.setBytes(6, taskimage);
pstmt.setString(7, taskThemeId);
pstmt.setInt(8, taskNumber);
```

```
pstmt.setString(9, rightanswer);
pstmt.executeUpdate();
pstmt.close();
                }
            }
conn.close();
        } catch (Exception e) {
e.printStackTrace();
        }

try {
Class.forName("org.sqlite.JDBC");
Connection conn = DriverManager.getConnection("jdbc:sqlite:" + databasePath);

File textDirectory= new File(textDirectoryPath);
File[] files = textDirectory.listFiles();

for (File file: files) {
String name = file.getName();
String id = name.replaceAll("[^0-9]", "");
StringBuildersb= new StringBuilder();
BufferedReader reader = new BufferedReader(new FileReader(file));
String line;
while ((line = reader.readLine()) != null) {
sb.append(line);
sb.append(System.lineSeparator());
                }
reader.close();
String text = sb.toString();
PreparedStatementpstmt= conn.prepareStatement("INSERT INTO TaskThemes
(id_Task_theme, Task_theme) VALUES (?, ?)");
pstmt.setString(1, id);
pstmt.setString(2, text);
pstmt.executeUpdate();
pstmt.close();
            }

conn.close();
        } catch (Exception e) {
e.printStackTrace();
        }
```

```
    }

private static String readFile(File file) throws Exception {
BufferedReader reader = new BufferedReader(new FileReader(file));
StringBuildersb= new StringBuilder();
String line;
while ((line = reader.readLine()) != null) {
sb.append(line);
sb.append(System.lineSeparator());
        }
reader.close();
return sb.toString().trim();
    }

private static byte[] readBytes(File file) throws Exception {
if (!file.exists()) {
return null;
        }
byte[] data = new byte[(int) file.length()];
java.io.FileInputStreamfis= new java.io.FileInputStream(file);
fis.read(data);
fis.close();
return data;
    }
}
```

ПРИЛОЖЕНИЕБ

Код генерации кнопок. Документ java

```
packagecom.dragikgames.mathege;

importandroid.content.Intent;
importandroid.database.Cursor;
importandroid.database.sqlite.SQLiteDatabase;
importandroid.graphics.Color;
importandroid.graphics.drawable.Drawable;
importandroid.graphics.drawable.GradientDrawable;
importandroid.graphics.drawable.LayerDrawable;
importandroid.os.Bundle;

importcom.google.android.material.floatingactionbutton.FloatingActionButton;
importcom.google.android.material.snackbar.Snackbar;

importandroidx.appcompat.app.AppCompatActivity;
importandroidx.appcompat.widget.Toolbar;

importandroid.util.Pair;
importandroid.view.Gravity;
importandroid.view.View;
importandroid.view.ViewGroup;
importandroid.widget.Button;
importandroid.widget.LinearLayout;
importandroid.widget.TextView;

importjava.util.List;

publicclassTaskActivityextendsAppCompatActivity{
privateList<Pair<Integer, String>>taskButtons;


@Override
protectedvoidonCreate(BundlesavedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_task);

// ПолучаемпараметрыизIntent
inttaskThemeId= getIntent().getIntExtra("id", 0);

// Получаемсписокзадачдляданнойтемы
DatabaseHelperdbHelper= newDatabaseHelper(this);
taskButtons= dbHelper.getTaskButtons(taskThemeId);

// Создаемкнопкидлякаждойзадачи
LinearLayoutlayout= findViewById(R.id.layout_task_buttons);



for(Pair<Integer, String>taskButton: taskButtons) {
Buttonbutton= newButton(this);
button.setText(taskButton.second);
```

```
button.setBackground(getResources().getDrawable(R.drawable.rounded_button_background));
;


intwidth= getResources().getDisplayMetrics().widthPixels;
intbuttonWidth= (int) (width* 0.8);
ViewGroup.LayoutParamslayoutParams= newViewGroup.LayoutParams(
buttonWidth, ViewGroup.LayoutParams.WRAP_CONTENT);

button.setLayoutParams(layoutParams);
button.setOnClickListener(newView.OnClickListener() {
@Override
publicvoidonClick(Viewv) {
// Открываемновоеактивитипринажатиинакнопку
Intentintent= newIntent(TaskActivity.this, TaskDetailsActivity.class);
intent.putExtra("id", taskButton.first);
intent.putExtra("name", taskButton.second);
startActivity(intent);
                }
            });
layout.addView(button);
LinearLayout.LayoutParamslayoutParams1 = (LinearLayout.LayoutParams)
button.getLayoutParams();
// Устанавливаемотступснизуна 16dp
layoutParams1.setMargins(0, 26, 0, 0 );
        }
    }
}
```

ПРИЛОЖЕНИЕВ

Код генерации заданий. Документ java

```
package com.dragikgames.mathege;

import androidx.appcompat.app.AppCompatActivity;

import android.content.ContentValues;
import android.content.Intent;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;

public class TaskDetailsActivityextendsAppCompatActivity{
private EditTextanswerEditText;
private Button checkAnswerButton;
private DatabaseReferencemDatabase;
private inttaskId;
private intuserAnswer;
String taskIdString;

//Снизутест
private String userId="Erop";

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

answerEditText= findViewById(R.id.answer_edit_text);
checkAnswerButton= findViewById(R.id.check_answer_button);


mDatabase= FirebaseDatabase.getInstance().getReference();


// Получаемпараметрыиз Intent
taskId= getIntent().getIntExtra("id", 0);

// Получаемданныезадачиизбазыданных
DatabaseHelperdbHelper= new DatabaseHelper(this);
SQLiteDatabasedb= dbHelper.getReadableDatabase();
String[] projection = {"TaskName", "TaskDescription", "TaskImage"};
String selection = "TaskId=?";
```

```
String[] selectionArgs= {String.valueOf(taskId)};
Cursor cursor= db.query("Tasks", projection, selection, selectionArgs, null, null,
null);
String taskName = "";
String taskDescription = "";
byte[] taskImage = null;
if (cursor != null &&cursor.moveToFirst()) {
inttaskNameIndex= cursor.getColumnIndexOrThrow("TaskName");
taskName = cursor.getString(taskNameIndex);
inttaskDescriptionIndex= cursor.getColumnIndexOrThrow("TaskDescription");
taskDescription = cursor.getString(taskDescriptionIndex);
inttaskImageIndex= cursor.getColumnIndexOrThrow("TaskImage");
taskImage = cursor.getBlob(taskImageIndex);
        }
if (cursor != null) {
cursor.close();
        }

// Отображаемданныезадачи
TextViewtaskNameView= findViewById(R.id.task_name);
taskNameView.setText(taskName);
TextViewtaskDescriptionView= findViewById(R.id.task_description);
taskDescriptionView.setText(taskDescription);
ImageViewtaskImageView= findViewById(R.id.task_image);
if (taskImage != null) {
Bitmap bitmap= BitmapFactory.decodeByteArray(taskImage, 0, taskImage.length);
taskImageView.setImageBitmap(bitmap);
        }
else {
taskImageView.setVisibility(View.GONE);
Log.d("TaskDetailsActivity", "No image found for taskId=" + taskId);
        }
```

```
checkAnswerButton.setOnClickListener(new View.OnClickListener() {
@Override
public void onClick(View v) {
userAnswer= 0;
try {
userAnswer= Integer.parseInt(answerEditText.getText().toString());
                } catch (NumberFormatExceptione) {
Toast.makeText(TaskDetailsActivity.this, "Введитечисло", Toast.LENGTH_SHORT).show();
return;
                }

// Получаемпараметрыиз Intent
taskId= getIntent().getIntExtra("id", 0);
taskIdString= String.valueOf(taskId);

// Получаемданныезадачиизбазыданных
DatabaseHelperdbHelper= new DatabaseHelper(TaskDetailsActivity.this);
SQLiteDatabasedb= dbHelper.getReadableDatabase();
String[] projection = {"RightAnswer"};
String selection = "TaskId=?";
String[] selectionArgs= {String.valueOf(taskId)};
Cursor cursor= db.query("Tasks", projection, selection, selectionArgs, null, null,
```

```
null);
intrightAnswer=0;
if (cursor != null &&cursor.moveToFirst()) {
rightAnswer = cursor.getInt(cursor.getColumnIndexOrThrow("RightAnswer"));
                }
if (cursor != null) {
cursor.close();
                }

Log.d("TaskDetailsActivity", "userAnswer = " + userAnswer);
Log.d("TaskDetailsActivity", "rightAnswer = " + rightAnswer);
// Сравниваемответпользователяиправильныйответ
if (userAnswer== rightAnswer) {
Toast.makeText(TaskDetailsActivity.this, "Верно!", Toast.LENGTH_SHORT).show();
                } else {
Toast.makeText(TaskDetailsActivity.this, "Неверно!", Toast.LENGTH_SHORT).show();
                }

ContentValues values = new ContentValues();
values.put("UserText", userAnswer);
intcount= db.update("Tasks", values, selection, selectionArgs);
/*  if (count > 0) {
Toast.makeText(TaskDetailsActivity.this, "Ответсохранен!", Toast.LENGTH_SHORT).show();
                } else {
Toast.makeText(TaskDetailsActivity.this, "Ошибкасохраненияответа!",
Toast.LENGTH_SHORT).show();
                }*/

writeNewUser();
            }
        });



TextViewmyTextView= findViewById(R.id.bottom_text_view);
myTextView.setOnClickListener(new View.OnClickListener() {
@Override
public void onClick(View view) {
// переходкновойактивности
Intent intent= new Intent(TaskDetailsActivity.this, TaskAnswer.class);
intent.putExtra("id", taskId);
startActivity(intent);
            }
        });
    }

public void writeNewUser() {
User user= new User(1,taskId,userAnswer);


mDatabase.child("users").child(userId).child(taskIdString).setValue(user);
    }
}
```

Выпускная квалификационная работа выполнена мной самостоятельно. Использованные в работе материалы и концепции из опубликованной научной литературы и других источников имеют ссылки на них.

Отпечатано в одном экземпляре.

Библиография 8 наименований.

Один экземпляр сдан на кафедру.

«____» _____ 2023 г.

_____ Драганов Ярослав Русланович
подпись

Министерство науки и высшего образования РФ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Хакасский технический институт – филиал ФГАОУ ВО
«Сибирский федеральный университет»

Кафедра прикладной информатики, естественно-научных
и гуманитарных дисциплин

УТВЕРЖДАЮ
Заведующий кафедрой
_____ О. В. Папина
подпись
«19» июня 2023 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.03 Прикладная информатика

Разработка мобильного приложения «Тренажер ЕГЭ по математике»

Руководитель _____ 19.06.23г. доцент, канд. физ.–мат. наук М. А. Буреева
подпись, дата

Выпускник _____ 19.06.23 Я. Р. Драганов
подпись, дата

Консультанты
по разделам:

Экономический _____ 19.06.2023г. М. А.Буреева
подпись, дата

Нормоконтролер _____ 19.06.2023 А. Н. Кадычегова
подпись, дата

Абакан 2023