МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования

«Гомельский государственный университет им. Франциска Скорины» Факультет физики и информационных технологий

Кафедра общей физики

Разработка приложения-викторины

Курсовая работа

Исполнитель:

Студент группы МС-42: _____ Марченко К.В.

Научный руководитель:

Старший преподаватель

Кафедры общей физики _____ Грищенко В.В.

Гомель 2023

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В современную цифровую эпоху мобильные приложения стали неотъемлемой частью нашей жизни, предлагая удобство, развлечения и возможности обучения под рукой. Одним из таких приложений, которое органично сочетает в себе образование и развлечение, является приложение-викторина.

Целью данной курсовой работы является создание мобильного приложения-викторины, отвечающего современным потребностям и требованиям пользователей. Проект направлен на обучение пользователя в интерактивной форме. В фокусе внимания – создание приложения, которое не только соответствует высоким технологическим стандартам, но и приносит реальную пользу пользователям.

Несколько плюсов приложения-викторины:

- Интерактивное обучение. Приложение-викторина обеспечивает интерактивный и увлекательный процесс обучения. Он побуждает пользователей активно участвовать в процессе обучения, проверяя свои знания и навыки критического мышления.

- Гибкость и удобство. Приложения викторин предлагают пользователям гибкость и удобство. Они могут получить доступ к викторинам в любое время и в любом месте, используя свои мобильные устройства.

- Персонализированное обучение. Приложения-викторины можно адаптировать к индивидуальным предпочтениям и целям обучения. Пользователи могут выбирать конкретные категории или темы викторин, соответствующие их интересам или областям деятельности. Такой персонализированный подход гарантирует, что пользователи будут взаимодействовать с контентом, который для них актуален и значим, что повышает их мотивацию и сохранение знаний.

- Разнообразие тем и уровней. Приложения-викторины обычно предлагают широкий выбор тем и уровней сложности. Такое разнообразие позволяет пользователям изучать различные предметы, расширять свои знания и бросать вызов самим себе в удобном для них темпе. Пользователи могут выбирать из таких категорий, как история, наука, литература, спорт и т. д., в зависимости от их конкретных интересов и предпочтений в обучении.

Мобильные приложения в настоящее время стали незаменимым инструментом в повседневной жизни. Их использование предоставляет уникальную возможность обеспечить максимальную мобильность и доступность. Разработка приложения, ориентированного на мобильные устройства, позволяет удовлетворить потребности современного пользователя.

1 Анализ предметной области

1.1 Актуальность приложений для учета личных финансов

Предметная область приложений-викторин остаётся актуальной из-за своей незамысловатости и интерактивности, и это обусловлено несколькими фундаментальными тенденциями.

Образование и обучение:
Приложения-викторины предлагают динамичный и интерактивный процесс обучения. Они предоставляют образовательным учреждениям, преподавателям и учащимся платформу для закрепления концепций, оценки знаний и содействия активному обучению. Приложения-викторины могут охватывать широкий спектр предметов и тем, что делает их актуальными для академических целей, подготовки к экзаменам и непрерывного обучения.

Развитие навыков:
Приложения-викторины выходят за рамки простого приобретения знаний. Они стимулируют критическое мышление, навыки решения проблем и принятия решений. Предлагая пользователям сложные вопросы и обеспечивая мгновенную обратную связь, приложения-викторины способствуют интеллектуальному развитию и улучшают когнитивные способности. Они могут быть разработаны так, чтобы сосредоточиться на конкретных навыках, таких как знание языка, логическое рассуждение или математические способности.

Геймификация и вовлеченность:
Приложения-викторины часто включают в себя элементы геймификации, такие как системы подсчета очков, достижения и таблицы лидеров. Эти игровые функции делают обучение более приятным и интересным для пользователей. Геймификация мотивирует пользователей активно участвовать, соревноваться с другими и стремиться к более высоким баллам, что приводит к лучшему сохранению знаний и положительному опыту обучения.

Развлечение и отдых:
Приложения-викторины служат источником развлечений и отдыха для пользователей всех возрастов. Они предлагают интересный способ бросить вызов самому себе, посоревноваться с друзьями и открыть для себя новые факты и мелочи. Приложения-викторины могут охватывать различные темы, в том числе общие знания, поп-культуру, спорт и многое другое, отвечая разнообразным интересам и предпочтениям.

Персонализация и адаптивность:
Приложения-викторины можно персонализировать в соответствии с индивидуальными предпочтениями и целями обучения. Они могут адаптироваться к производительности пользователей и соответствующим образом регулировать уровень сложности. Такой персонализированный подход

гарантирует, что пользователи будут заинтересованы и вовлечены соответствующим образом, способствуя эффективному обучению и сохранению знаний.

Таким образом, приложения-викторины продолжают играть важную роль в повседневной жизни, повышая их грамотность и делая жизнь разнообразнее.

1.2 Анализ рынка приложений для учета личных финансов

Рынок приложений-викторин в последние годы пережил значительный рост и популярность. Давайте проанализируем некоторые ключевые аспекты рынка:

Размер и рост рынка. На рынке приложений-викторин наблюдается значительный рост, обусловленный такими факторами, как рост проникновения смартфонов, растущий спрос на платформы интерактивного обучения и популярность геймифицированного образования. Ожидается, что рынок продолжит расширяться, поскольку все больше пользователей ищут увлекательные и образовательные возможности на своих мобильных устройствах.

Сегментация рынка. Рынок приложений-викторин можно сегментировать на основе различных факторов, включая целевую аудиторию, платформу и цель. Приложения-викторины предназначены для широкого круга пользователей, включая студентов, специалистов и обычных учащихся. Они доступны на нескольких платформах, включая Android, iOS и веб-приложения. Кроме того, приложения-викторины служат различным целям, например, образовательным викторинам, викторинам, изучению языка и подготовке к экзаменам.

Конкурентная среда. Рынок приложений-викторин отличается высокой конкуренцией, и за долю рынка борются многочисленные игроки. В это пространство вошли авторитетные компании в области образовательных технологий, независимые разработчики и даже традиционные медиакомпании. Ключевые игроки часто предлагают широкий спектр приложений-викторин, охватывающих различные темы и ориентированных на разные демографические группы пользователей. Дифференциация на основе функций приложения, качества контента, дизайна и пользовательского опыта имеет решающее значение для успеха на рынке.

Модели монетизации. В приложениях-викторинах используются различные модели монетизации для получения дохода. Общие стратегии включают покупки в приложении, премиум-версии с дополнительными функциями, рекламу и услуги на основе подписки. Выбор модели монетизации зависит от целевой аудитории приложения, контента и пользовательского

опыта. Разработчикам необходимо найти баланс между монетизацией и удовлетворенностью пользователей, чтобы обеспечить устойчивый рост.

Технологические достижения. Технологические достижения сыграли значительную роль в формировании рынка приложений для викторин. Такие функции, как многопользовательская игра в реальном времени, интеграция с платформами социальных сетей, облачное хранилище и алгоритмы машинного обучения для адаптивного обучения, повысили функциональность и привлекательность приложений для викторин. Разработчики постоянно используют новые технологии для создания инновационных и захватывающих викторин.

Вовлечение и удержание пользователей. Вовлечение и удержание пользователей являются важнейшими факторами успеха на рынке приложений-викторин. Разработчики сосредотачиваются на создании привлекательного контента, реализации элементов геймификации и обеспечении функций социального взаимодействия, чтобы поддерживать интерес пользователей. Регулярные обновления контента, задачи, достижения и таблицы лидеров помогают поддерживать интерес пользователей и способствуют долгосрочному использованию.

Интеграция рынка образования. Приложения-викторины нашли актуальность в образовательных учреждениях, от школ K-12 до высших учебных заведений. Учителя используют приложения-викторины, чтобы дополнить обучение в классе, оценить знания учащихся и предоставить персонализированную обратную связь. Интеграция с системами управления обучением (LMS) и образовательными платформами стала важной, обеспечивая плавную интеграцию данных приложений викторин в более широкую образовательную экосистему.

Потенциал международного рынка. Приложения-викторины имеют глобальную привлекательность и могут быть ориентированы на различные рынки и языки. Усилия по локализации, включая переводы и культурно релевантный контент, позволяют приложениям-викторинам эффективно проникать на международные рынки. Выход на развивающиеся рынки с ростом популярности смартфонов открывает значительные возможности роста для разработчиков приложений-викторин.

Конфиденциальность и безопасность данных. Как и в случае с любым приложением, которое собирает пользовательские данные, обеспечение конфиденциальности и безопасности данных имеет решающее значение для поддержания доверия пользователей. Разработчики приложений Quiz должны соблюдать правила защиты данных и применять надежные меры безопасности для защиты информации пользователей.

В заключение отметим, что рынок приложений-викторин продолжает расти благодаря растущему спросу на интерактивное обучение, геймификацию и доступность мобильных устройств. Разработчикам необходимо

сосредоточиться на стратегиях дифференциации, вовлечения пользователей и монетизации, чтобы добиться успеха на этом конкурентном рынке. По мере развития технологий на рынке приложений-викторин, скорее всего, будут появляться новые инновации и достижения, предоставляющие пользователям более захватывающий и персонализированный опыт обучения.

1.2.1 Приложение 94% - Викторины, викторины и логика

Эта игра привлекла внимание пользователей как лучшая викторина для Android. В отличие от обычных игр-викторин, эта игра, как следует из названия, задаст вам вопрос и ожидает, что вы дадите правильные ответы. За каждый правильный ответ вы получите балл, и общий балл, основанный на количестве возможных правильных ответов, составит 94%.

Вопросы будут охватывать различные темы, такие как ботаника, география, человеческое тело, история и многие другие. Чтобы помочь вам перейти к правильному ответу, вам будет предложена опция под названием «Джокер». Вариант с джокером генерирует подсказки для вас, и вы можете использовать его только один раз в начале игры, так как вам нужно будет разблокировать его по мере прохождения игра.

Рисунок 1 – Иконка приложения 94% - Викторины, викторины и логика

Важные особенности:

- Внутри опции «Джокер» вы увидите опции, например, «Показать позже», чтобы показать вам последнюю букву слово «Покажите слово», чтобы раскрыть все слово, и «Удалите буквы», чтобы в итоге получить возможные письма.

Схема вопросов будет такая, например, что поесть в кинотеатре? Вы должны угадать реальный сценарий для правильного ответа, и его легко угадать.

Разработчики часто обновляют игры и задают больше интересных вопросов, чтобы избавить вас от скуки.

Эту игру можно загрузить бесплатно, но для доступа к дополнительным функциям возможны микротранзакции.

Хоть данное приложение имеет скудный функционал, но оно достаточно понятно пользователю и им удобно пользоваться, а платная версия открывает не так уж и много возможностей по сравнению с бесплатной.

1.2.2 Приложение HQ мелочи

Эта игра для Android была в центре внимания с момента ее выпуска в 2018 году. Очень немногие викторины для Android предлагают реальные деньги, как HQ Trivia. Он предоставляет вам реальные деньги, когда вы участвуете в их регулярных и еженедельных конкурсах. Эта игра проводится в два разных времени в будние дни, а именно в 15:00. и 21:00 По восточноевропейскому времени, а один раз по выходным, то есть в 21:00. ET.

Вам будет задано 10 вопросов, и на каждый вы получите всего 10 секунд, которые пролетят очень быстро. Кроме того, вы можете настраивать ответы или получать подсказки, но угадывать ответы. Если вы не ответите на один вопрос, вы будете исключены из раунда. Однако, если вы доживете до последнего, тогда все подходящие участники смогут разделить деньги.

Рисунок 2 – Иконка приложения HQ мелочи

Важные особенности:

- Эта игра проста благодаря своим интуитивно понятным правилам и в нее легко играть, пока сохраняется время.

- Призовые деньги в виртуальном мире становятся реальными после победы в игре.

-Призовые деньги могут быть разделены, если победителей будет больше одного, и все получат равную долю денег.

-Многие известные знаменитости, такие как Рок, Роберт Де Ниро, Гордон Рамзи, приняли участие в вручении призов на миллионы долларов своим претендентам.

Хоть это приложении и ориентировано не на обучении пользователя, но в нём реализовано крайне интересная особенность – выплата денежных средств.

1.2.3 Итоги анализа конкурентов

Итак, проведем анализ двух популярных приложений-викторин и выведем список их достоинств и недостатков для определения ключевых аспектов, на которые стоит обратить внимание при разработке нашего приложения.

Достоинства, на которые нужно обратить внимание:

- обширный функционал с подробной статистикой данных

- возможность вести неограниченное количество записей

Недостатки которые нужно свести к минимуму:

-минимизировать баги приложения

- сделать бесплатную полную версию

- сделать полностью понятный функционал для пользователя

Для получения качественного приложения необходимо учесть эти аспекты, предоставив пользователю удобное и многофункциональное приложение.

1.3 Описание используемых технологий и инструментов

1.3.1 Android Studio

Android Studio — это официальная интегрированная среда разработки (IDE) для разработки приложений Android. Он предоставляет разработчикам полный набор инструментов и функций для создания, тестирования и развертывания приложений Android. Вот обзор Android Studio и ее ключевых функций:

Редактирование кода и структура проекта:

Android Studio предлагает мощный редактор кода с такими функциями, как завершение кода, подсветка синтаксиса и рефакторинг кода.

IDE поддерживает несколько языков программирования, включая Java, Kotlin и C++, что позволяет разработчикам выбирать предпочтительный язык для разработки приложений для Android.

Android Studio организует проекты в иерархическую структуру, включая модули, библиотеки и ресурсы, что упрощает управление крупномасштабными проектами.

Редактор макета:

Android Studio предоставляет визуальный редактор макетов, который позволяет разработчикам создавать пользовательский интерфейс (UI) своего приложения с помощью интерфейса перетаскивания.

Редактор макетов поддерживает широкий спектр компонентов пользовательского интерфейса, включая кнопки, текстовые представления, изображения и многое другое.

Разработчики могут предварительно просмотреть пользовательский интерфейс приложения в режиме реального времени, что упрощает повторение и доработку дизайна.

Система сборки Gradle:

Android Studio использует Gradle в качестве системы сборки проектов Android.

Gradle упрощает процесс управления зависимостями, компиляции кода и упаковки приложения для распространения.

Разработчики могут легко добавлять внешние библиотеки и зависимости в свои проекты с помощью Gradle, расширяя функциональность и возможности своих приложений.

Тестирование эмулятора и устройства:

Android Studio включает в себя эмулятор, который позволяет разработчикам тестировать свои приложения на виртуальных устройствах Android.

Эмулятор поддерживает различные конфигурации устройств, размеры экрана и версии Android, что позволяет проводить комплексное тестирование.

Разработчики также могут подключать физические устройства Android к своей машине разработки для тестирования и отладки в реальном времени.

Отладка и профилирование:

Android Studio предлагает надежные возможности отладки, позволяющие разработчикам выявлять и устранять проблемы в своем коде.

В среде IDE предусмотрены такие функции, как точки останова, пошаговое выполнение, проверка переменных и просмотрщик logcat для эффективной отладки.

Разработчики также могут использовать инструменты профилирования для анализа производительности своего приложения, выявления узких мест и оптимизации использования ресурсов.

Контроль версий и совместная работа:

Android Studio легко интегрируется с популярными системами контроля версий, такими как Git, обеспечивая эффективную совместную работу и управление кодом.

Разработчики могут легко фиксировать изменения, переключаться между ветвями и разрешать конфликты слияния внутри IDE.

Android Studio также предоставляет инструменты для проверки кода, упрощая совместную работу команд разработчиков над проектами.

Развертывание и распространение приложений:

Android Studio упрощает процесс создания и упаковки приложений для распространения.

Разработчики могут создавать подписанные APK (пакеты приложений Android) для выпуска в Google Play Store или других каналах распространения.

IDE предлагает инструменты для управления ключами подписи приложений, оптимизации размера приложения и обработки различных вариантов сборки (например, отладки, выпуска).

Интеграция с сервисами Google:

Android Studio легко интегрируется с различными сервисами и API Google, такими как Google Maps, Firebase, Google Cloud Platform и другими.

Используя эти интеграции, разработчики могут легко добавлять в свои приложения такие функции, как службы определения местоположения, push-уведомления, аналитику и аутентификацию.

Подводя итог, можно сказать, что Android Studio — это многофункциональная интегрированная среда разработки, специально разработанная для разработки приложений для Android. Он предоставляет разработчикам полный набор инструментов и функций, включая редактирование кода, дизайн пользовательского интерфейса, тестирование, отладку, совместную работу и распространение приложений. Android Studio упрощает процесс разработки, предоставляя разработчикам возможность создавать высококачественные многофункциональные приложения для Android.

1.3.2 Язык программирования Java

Язык программирования Java — это высокоуровневый язык программирования общего назначения, разработанный компанией Sun Microsystems (ныне принадлежащей Oracle Corporation) в середине 1990-х годов. Он спроектирован так, чтобы быть независимым от платформы, надежным и безопасным, что делает его популярным выбором для разработки широкого спектра программных приложений. Вот обзор ключевых особенностей и характеристик языка программирования Java:

Объектно-ориентированный: Java — это язык объектно-ориентированного программирования (ООП), что означает, что он основан на концепции объектов и классов. Он предоставляет такие функции, как инкапсуляция, наследование и полиморфизм, позволяющие разработчикам создавать модульный и повторно используемый код.

Независимость от платформы. Одним из основных преимуществ Java является независимость от платформы. Исходный код Java компилируется в байт-код, который может быть выполнен в любой системе с виртуальной машиной Java (JVM). Эта возможность «написать один раз, запустить где угодно» позволяет развертывать приложения Java в различных операционных системах без необходимости перекомпиляции.

Сбор мусора: Java включает автоматическое управление памятью с помощью метода, называемого сборкой мусора. JVM автоматически выполняет

выделение и освобождение памяти, освобождая разработчиков от задач управления памятью вручную. Это помогает предотвратить утечки памяти и делает приложения Java более надежными и менее склонными к сбоям, вызванным проблемами, связанными с памятью.

Строгая система типов: Java имеет строгую систему статических типов, что означает, что типы переменных проверяются во время компиляции. Это помогает выявить ошибки, связанные с типом, на ранней стадии и повышает надежность и удобство сопровождения кода.

Обработка исключений: Java предоставляет надежные механизмы обработки исключений. Исключения используются для обработки и восстановления ошибок во время выполнения, гарантируя, что приложения смогут корректно обрабатывать непредвиденные ситуации и восстанавливаться после ошибок без сбоев.

Стандартная библиотека: Java поставляется с обширной стандартной библиотекой, известной как Java Development Kit (JDK), которая предоставляет широкий спектр готовых классов и API для общих задач программирования. Стандартная библиотека включает утилиты для ввода/вывода, работы в сети, подключения к базам данных, разработки графического пользовательского интерфейса и т. д., что экономит время и усилия разработчиков при создании этих функций с нуля.

Поддержка многопоточности: Java имеет встроенную поддержку многопоточности, что позволяет разработчикам создавать параллельные и параллельные программы. Это позволяет приложениям выполнять несколько задач одновременно, повышая производительность и скорость реагирования.

Безопасность. Безопасность является ключевым моментом языка Java. Он включает в себя функции безопасности, такие как изолированная среда, безопасная загрузка классов и API шифрования. Это делает Java популярным выбором для разработки безопасных приложений, особенно в таких областях, как электронная коммерция и финансовые системы.

Сообщество и экосистема. Java имеет большое и активное сообщество разработчиков, которое внесло свой вклад в богатую экосистему библиотек, фреймворков и инструментов. Эта экосистема предоставляет разработчикам широкий спектр ресурсов и поддержки для эффективного создания приложений Java.

Подводя итог, можно сказать, что язык программирования Java — это универсальный и широко используемый язык, известный своей независимостью от платформы, надежностью и надежностью. Его объектно-ориентированный характер, обширная стандартная библиотека и сильная поддержка сообщества делают его идеальным выбором для разработки различных приложений, от настольного программного обеспечения до корпоративных систем и веб-приложений.

2. Описание основных этапов разработки приложения

2.1 База данных SQLite

База данных SQLite обладает несколькими преимуществами, которые делают ее популярным выбором для различных приложений. Вот некоторые из преимуществ SQLite:

1. Простота использования: SQLite является легким в использовании и настройке. Он не требует сложной установки или конфигурации сервера баз данных, поэтому его можно легко внедрить в приложение без лишних сложностей.

2. Портативность: SQLite является самодостаточной базой данных, которая хранится в одном файле. Это означает, что можно просто скопировать этот файл и перенести его на другую платформу или компьютер, и база данных SQLite будет работать без проблем. Это делает ее отличным выбором для портативных приложений или приложений, требующих простого распространения базы данных.

3. Надежность: SQLite обеспечивает транзакционность и целостность данных, что делает его надежным выбором для приложений, где важна сохранность данных. Он использует механизмы записи на диск, которые обеспечивают долговременное хранение данных даже при сбоях системы или отключении питания.

4. Эффективность: SQLite обеспечивает высокую производительность при обработке запросов. Он имеет оптимизированный движок базы данных, который хорошо справляется с большими объемами данных. SQLite также поддерживает индексы, что ускоряет выполнение запросов к базе данных.

5. Небольшой размер: База данных SQLite имеет небольшой размер, что делает ее идеальным выбором для приложений с ограниченными ресурсами, таких как мобильные устройства или встроенные системы. База данных SQLite может быть включена в приложение без значительного увеличения его общего размера.

6. Широкая поддержка: SQLite является одной из самых популярных баз данных, и он имеет широкую поддержку в различных языках программирования. Существуют библиотеки и драйверы для множества платформ и языков, что облегчает работу с базой данных SQLite в различных окружениях.

Поэтому мною будет использована данная база данных, по-скольку мы можем её использовать локально на нашем устройстве, что упростит нашу работу над приложение.

2.2 Описание базы данных

Разрабатываемое приложение будет иметь встроенную локальную базу данных SQLite.

При проектировании базы данных для мобильного приложения-викторины следует учитывать ряд требований.

Вот некоторые из них:

1. Хранение данных о вопросах и ответах: БД хранить вопросы викторины, ответы на эти вопросы и знать правильные ответы.

2. Масштабируемость и производительность: БД должна быть спроектирована с учетом возможности масштабирования и обеспечения высокой производительности при работе с большим объемом данных и одновременным доступом нескольких пользователей.

3. Безопасность данных: БД должна обладать механизмами обеспечения безопасности данных, такими как шифрование, защита от несанкционированного доступа, резервное копирование и восстановление данных.

На основе данных требований мною была спроектирована база данных состоящая из одной таблицы Question And Answer.В ней хранятся:

ID-номер вопроса;

Question-вопрос;

Answer1-первый ответ;

Answer2-второй ответ;

Answer3-третий ответ;

Answer4-четвертый ответ;

Correct Answer-правильный ответ;

User Selected Answer- ответ, выбранный пользователем.

Далее приведем схемы модели данных.

Физическая модель данных - включает в себя все необходимые таблицы, столбцы, связи, свойства базы данных для физической реализации баз данных. Производительность базы данных, стратегия индексации, физическое хранилище (смотреть рисунок 3).

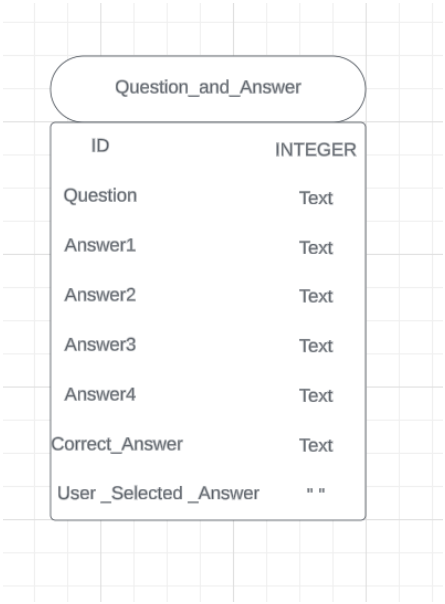


Рисунок 3 – Физическая модель данных

Логическая модель данных — это расширение концептуальной модели данных. Она включает в себя все сущности, атрибуты, ключи и взаимосвязи. (смотреть рисунок 4).

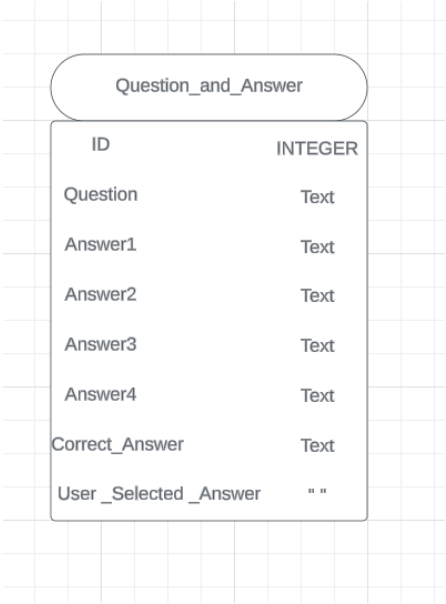


Рисунок 4 – Логическая модель данных

На этапе разработки концептуальной схемы выделяются сущности и показывается связь между ними. Концептуальная схема проектируемой базы данных имеет следующий вид: (смотреть рисунок 5).

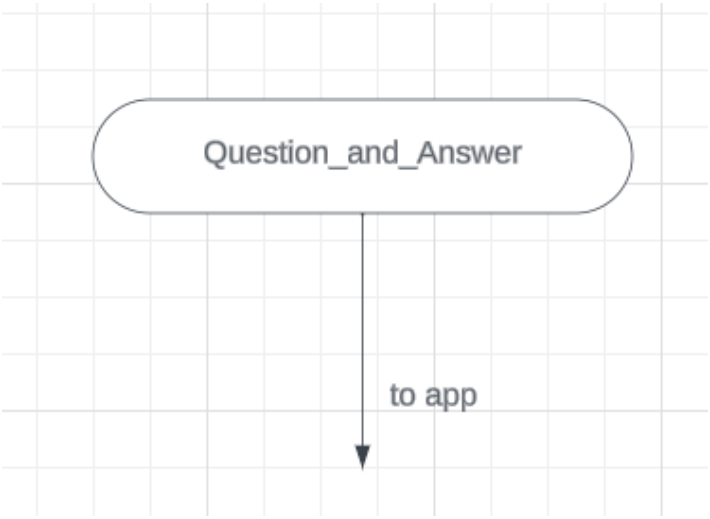


Рисунок 5 – Концептуальная модель данных

3 Схема описывающая работу приложения

Приложение-викторина - это приложение, которое предоставляет пользователю возможность участвовать в интерактивной викторине, отвечая на вопросы. Вот общее описание работы приложения-викторины:

Запуск приложения: Пользователь запускает приложение-викторину на своем устройстве.

Выбор викторины: Пользователю предоставляется список доступных викторин или категорий викторин. Он может выбрать интересующую его викторину для участия.

Отображение вопроса: Приложение отображает вопрос из выбранной викторины на экране пользователя. Вопрос может быть представлен в виде текста.

Ответ пользователя: Пользователь дает свой ответ на вопрос, выбирая один из предложенных вариантов.

Проверка ответа: Приложение проверяет ответ пользователя и определяет, правильно ли он ответил на вопрос.

Переход к следующему вопросу: После проверки ответа приложение переходит к следующему вопросу в викторине. Процесс повторяется до тех пор, пока не будут заданы все вопросы викторины.

Окончание викторины: Когда все вопросы викторины заданы, приложение отображает результаты пользователю. Может быть показан общий счет баллов.

Завершение приложения: Пользователь может завершить приложение после завершения викторины или продолжить участие в других викторинах.

Вид схемы приложения (см. рисунок 6).

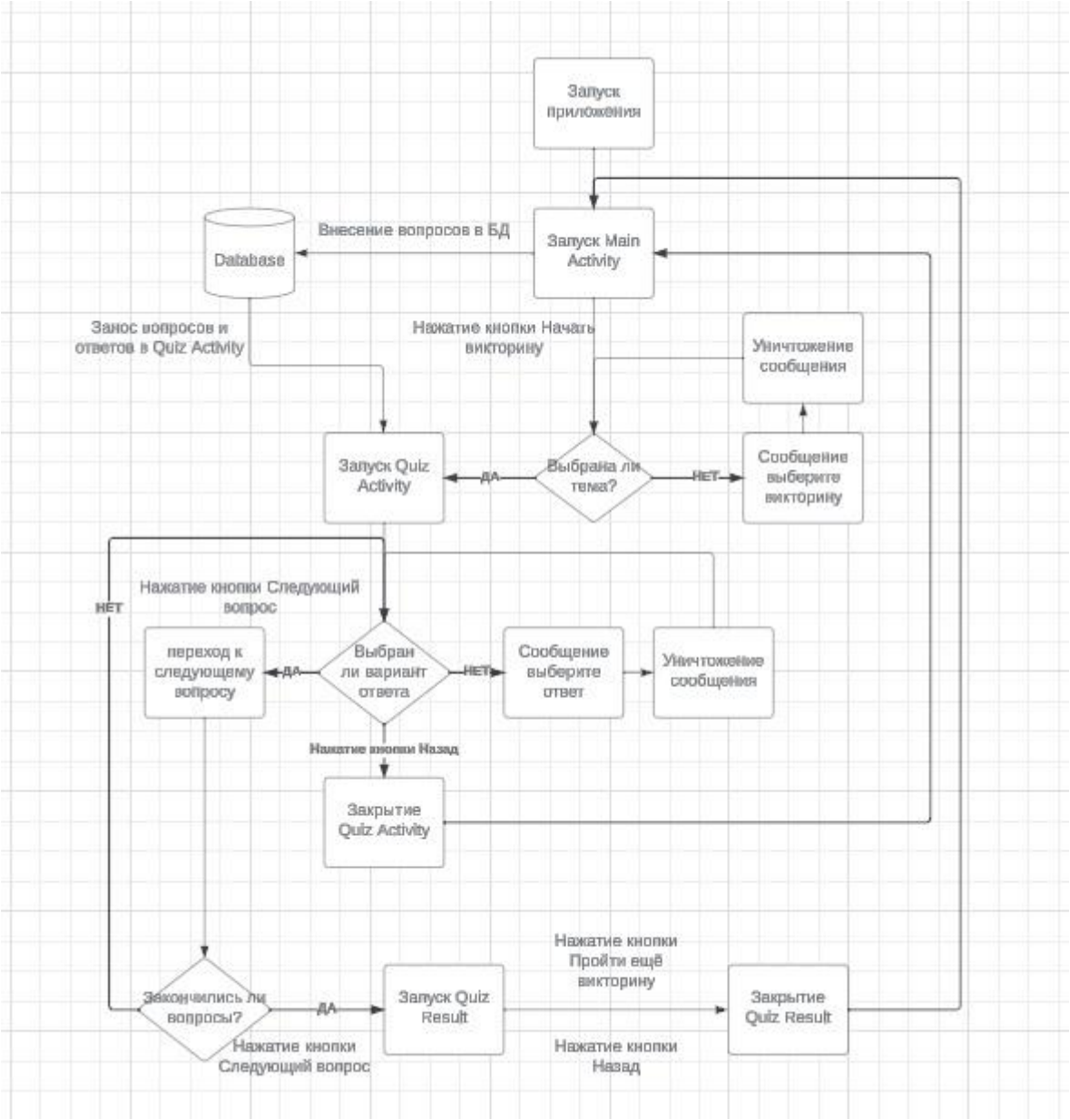


Рисунок 6 – схема взаимодействия

3.1 Обзор современных архитектурных решений приложения

Рассмотрим структуру нашего приложения-викторины.

Проект имеет следующую структуру:

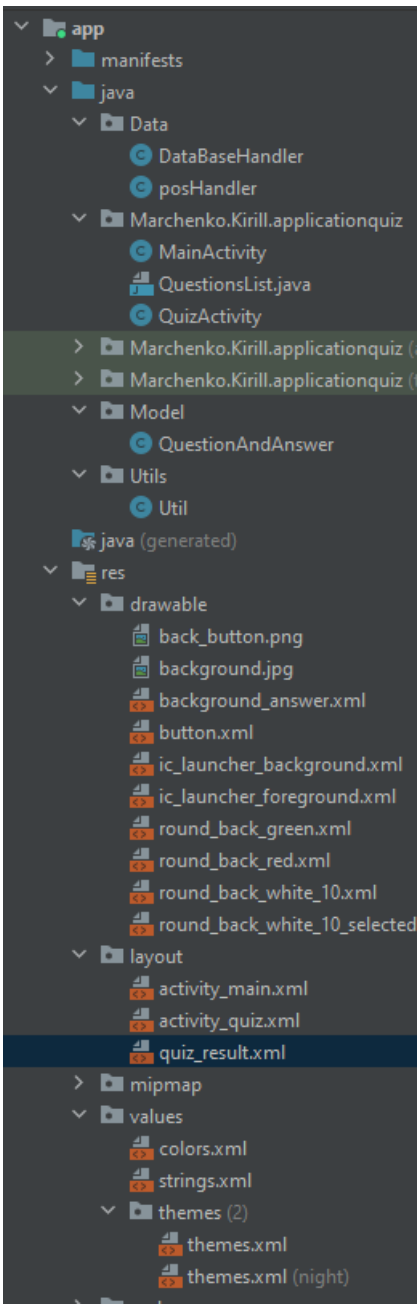


Рисунок 7 – структура проекта

В коде класса MainActivity.java, в методе onCreate(), основные моменты, которые можно выделить, это:

1)Создание экземпляра класса DataBaseHandler:

```
DataBaseHandler dataBaseHandler = new DataBaseHandler(this);
```

В этой строке создается объект dataBaseHandler, который будет использоваться для взаимодействия с базой данных.

Рисунок 8 – создание экземпляра класса

2)Добавление вопросов и ответов в базу данных:

```
dataBaseHandler.addQuest(new QuestionAndAnswer("Как древние Египтяне
называли луну?",
            "Ях", "Сарти", "Аади", "Селена", "Ях",""));
```

В этой и последующих строках вызывается метод addQuest() объекта data BaseHandler для добавления вопросов и ответов в базу данных. Каждый в опрос представлен экземпляром класса QuestionAndAnswer.

Рисунок 9 – добавление вопросов и ответов в базу данных

3)Обработка событий и взаимодействие с интерфейсом:

В коде также присутствуют обработчики событий и взаимодействие с элементами пользовательского интерфейса, такими как кнопки. Например:

```
Button startQuizButton = findViewById(R.id.startQuizButton);
startQuizButton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        // Действия при нажатии на кнопку "Начать викторину"
    }
});
```

В данном примере создается кнопка startQuizButton, устанавливается об работчик клика на эту кнопку, и определяются действия, которые будут выполняться при нажатии на кнопку.

Рисунок 10 – создание кнопки «начало викторины»

Эти моменты в коде MainActivity.java относятся к инициализации и настройке активности, добавлению данных в базу данных и обработке событий пользовательского взаимодействия. Они являются ключевыми для функционирования приложения викторины.

Код класса QuizActivity.java имеет следующую структуру и функциональность:

1)Импорт необходимых пакетов:

```
package Marchenko.Kirill.applicationquiz;

import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.AppCompatButton;

import android.annotation.SuppressLint;
import android.content.Intent;
import android.graphics.Color;
import android.os.Bundle;
import android.provider.ContactsContract;
import android.util.Log;
import android.view.View;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

import java.util.Arrays;
import java.util.Date;
import java.util.List;

import Data.DataBaseHandler;
import Data.posHandler;
import Model.QuestionAndAnswer;
```

Рисунок 11 – импорт пакетов

2)Объявление и инициализация переменных, связанных с элементами пользовательского интерфейса:

```
public class QuizActivity extends AppCompatActivity {

    public TextView numberQuestion;
    public TextView question;
    public AppCompatButton option1;
    public AppCompatButton option2;
    public AppCompatButton option3;
    public AppCompatButton option4;
    public AppCompatButton next_quest;
    public List<QuestionAndAnswer> questionAndAnswerList;
    public String selectedOptionByUser = "";
```

Рисунок 12 – объявление и инициализация переменных

3)Метод onCreate(), который выполняется при создании активности:

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_quiz);

    // Инициализация элементов пользовательского интерфейса
    numberQuestion = findViewById(R.id.numberQuestion);
    question = findViewById(R.id.question);
    option1 = findViewById(R.id.option_1);
    option2 = findViewById(R.id.option_2);
    option3 = findViewById(R.id.option_3);
    option4 = findViewById(R.id.option_4);
    next_quest = findViewById(R.id.next_quest);

    // Получение выбранной темы из предыдущей активности
    final String getSelectedTopic = getIntent().getStringExtra("SelectedTo
pic");

    // Обработка нажатия кнопки "Назад"
    final ImageView backButton = findViewById(R.id.backButton);
    backButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            startActivity(new Intent(QuizActivity.this, MainActivity.clas
s));
            finish();
        }
    });

    // Установка выбранной темы в текстовое поле
    final TextView selectedTopicName = findViewById(R.id.selectedTopicNam
e);
    selectedTopicName.setText(getSelectedTopic);
```

Рисунок 13 – метод onCreate

```
    // Инициализация базы данных и получение списка вопросов и ответов
    DataBaseHandler dataBaseHandler = new DataBaseHandler(this);
    List<QuestionAndAnswer> questionAndAnswerList = dataBaseHandler.getAll
Quest();

    // Установка вопроса и вариантов ответов на экране
    question.setText(questionAndAnswerList.get(posHandler.currentPos).getQ
uestion());
    option1.setText(questionAndAnswerList.get(posHandler.currentPos).getAn
swer1());
    option2.setText(questionAndAnswerList.get(posHandler.currentPos).getAn
swer2());
    option3.setText(questionAndAnswerList.get(posHandler.currentPos).getAn
swer3());
    option4.setText(questionAndAnswerList.get(posHandler.currentPos).getAn
swer4());

    // Обработка нажатия на варианты ответов
    option1.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            if (selectedOptionByUser.isEmpty()) {
                selectedOptionByUser = option1.getText().toString();
                option1.setBackgroundResource(R.drawable.round_back_red);
                option1.setTextColor(Color.WHITE);
            }
        }
    });

    // Аналогично обрабатываем остальные варианты ответов (option2, option
3, option4)

    // Обработка нажатия кнопки "Следующий вопрос"
    next_quest.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            if (selectedOptionByUser.isEmpty()) {
```

Рисунок 14 – метод onCreate

Это основная структура и функциональность класса QuizActivity.java.

Класс DataBaseHandler.java представляет собой обработчик базы данных SQLite. Он содержит методы для создания, обновления и взаимодействия с таблицей, содержащей вопросы и ответы для викторины.

```
package Data;

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;

import java.util.ArrayList;
import java.util.List;

import Model.QuestionAndAnswer;
import Utils.Util;

public class DataBaseHandler extends SQLiteOpenHelper {

    public DataBaseHandler(Context context) {
        super(context, Util.DATABASE_NAME, null, Util.DATABASE_VERSION);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        String CREATE_QUESTIONANDANSWER_TABLE = "CREATE TABLE " + Util.TAB
LE_NAME + " ("
                + Util.KEY_ID + " INTEGER PRIMARY KEY,"
                + Util.KEY_QUESTION + " TEXT,"
                + Util.KEY_ANSWER1 + " TEXT,"
                + Util.KEY_ANSWER2 + " TEXT,"
                + Util.KEY_ANSWER3 + " TEXT,"
                + Util.KEY_ANSWER4 + " TEXT,"
                + Util.KEY_CORRECTANSWER + " TEXT,"
                + Util.KEY_USERSELECTEDANSWER + " TEXT)";

        db.execSQL(CREATE_QUESTIONANDANSWER_TABLE);
    }
```

Рисунок 15– метод создания таблицы

```
        return questionAndAnswer;
    }

    public List<QuestionAndAnswer> getAllQuest() {
        List<QuestionAndAnswer> questionAndAnswerList = new ArrayList<>();
        SQLiteDatabase db = this.getReadableDatabase();
        String selectAllQuest = "SELECT * FROM " + Util.TABLE_NAME;
        Cursor cursor = db.rawQuery(selectAllQuest, null);

        if (cursor.moveToFirst()) {
            do {
                QuestionAndAnswer questionAndAnswer = new QuestionAndAnswe
r();
                questionAndAnswer.setId(Integer.parseInt(cursor.getString
(0)));
                questionAndAnswer.setQuestion(cursor.getString(1));
                questionAndAnswer.setAnswer1(cursor.getString(2));
                questionAndAnswer.setAnswer2(cursor.getString(3));
                questionAndAnswer.setAnswer3(cursor.getString(4));
                questionAndAnswer.setAnswer4(cursor.getString(5));
                questionAndAnswer.setCorrectAnswer(cursor.getString(6));
                questionAndAnswer.setUserSelectedAnswer(cursor.getString
(7));

                questionAndAnswerList.add(questionAndAnswer);
            } while (cursor.moveToNext());
        }

        return questionAndAnswerList;
    }
}
```

Рисунок 16 – метод вызова всех вопросов и ответов из таблицы

Это основной код класса DataBaseHandler, который отвечает за создание и управление базой данных SQLite для хранения вопросов и ответов в приложении викторины.

Класс Util.java содержит некоторые утилитарные значения, используемые в приложении, связанном с базой данных и вопросами для викторины.

```
package Utils;

public class Util {

    public static final int DATABASE_VERSION = 11;
    public static final String DATABASE_NAME = "QuestionsDB";
    public static final String TABLE_NAME = "Questions";


    public static final String KEY_ID = "id";
    public static final String KEY_QUESTION = "question";
    public static final String KEY_ANSWER1 = "answer1";
    public static final String KEY_ANSWER2 = "answer2";
    public static final String KEY_ANSWER3 = "answer3";
    public static final String KEY_ANSWER4 = "answer4";
    public static final String KEY_CORRECTANSWER = "correctanswer";
    public static final String KEY_USERSELECTEDANSWER = "userselectedanswe
r";
}
```

Рисунок 17 – класс Util.java

В классе Util определены следующие значения:
DATABASE_VERSION: версия базы данных.
DATABASE_NAME: имя базы данных.
TABLE_NAME: имя таблицы в базе данных.
Также определены константы для столбцов таблицы:
KEY_ID: идентификатор вопроса.
KEY_QUESTION: текст вопроса.
KEY_ANSWER1: первый вариант ответа.
KEY_ANSWER2: второй вариант ответа.
KEY_ANSWER3: третий вариант ответа.
KEY_ANSWER4: четвертый вариант ответа.
KEY_CORRECTANSWER: правильный ответ.
KEY_USERSELECTEDANSWER: ответ, выбранный пользователем.

Класс QuestionAndAnswer.java представляет модель для вопросов и ответов в приложении викторины. Он содержит поля для идентификатора, текста вопроса, вариантов ответа, правильного ответа и ответа, выбранного пользователем.

```java
package Model;

import java.util.logging.Logger;

public class QuestionAndAnswer {

    public static Logger log = Logger.getLogger(QuestionAndAnswer.class.ge
tName());

    private int id;
    private String Question;
    private String Answer1;
    private String Answer2;
    private String Answer3;
    private String Answer4;
    private String CorrectAnswer;
    private String UserSelectedAnswer;

    public QuestionAndAnswer() {
    }

    public QuestionAndAnswer(int id, String question, String answer1, Stri
ng answer2, String answer3, String answer4, String correctAnswer, String u
serSelectedAnswer) {
        this.id = id;
        Question = question;
        Answer1 = answer1;
        Answer2 = answer2;
        Answer3 = answer3;
        Answer4 = answer4;
        CorrectAnswer = correctAnswer;
        UserSelectedAnswer = userSelectedAnswer;
    }
```

Рисунок 18 – указание полей идентификатора

```java
    public QuestionAndAnswer(String question, String answer1, String answe
r2, String answer3, String answer4, String correctAnswer, String userSelec
tedAnswer) {
        this.Question = question;
        this.Answer1 = answer1;
        this.Answer2 = answer2;
        this.Answer3 = answer3;
        this.Answer4 = answer4;
        this.CorrectAnswer = correctAnswer;
        this.UserSelectedAnswer = userSelectedAnswer;
    }

    public int getId() {
        return id;
    }

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

    public String getQuestion() {
        return Question;
    }

    public void setQuestion(String question) {
        Question = question;
    }

    public String getAnswer1() {
        return Answer1;
    }

    public void setAnswer1(String answer1) {
        Answer1 = answer1;
    }

    public String getAnswer2() {
```

Рисунок 19 – указание полей идентификатора

```
@Override
public String toString() {
    return "QuestionAndAnswer{" +
            "id=" + id +
            ", Question='" + Question + '\'' +
            ", Answer1='" + Answer1 + '\'' +
            ", Answer2='" + Answer2 + '\'' +
            ", Answer3='" + Answer3 + '\'' +
            ", Answer4='" + Answer4 + '\'' +
            ", CorrectAnswer='" + CorrectAnswer + '\'' +
            ", UserSelectedAnswer='" + UserSelectedAnswer + '\'' +
            '}';
}
```

Рисунок 20 – переопределение метода toString

В классе QuestionAndAnswer определены поля и методы для доступа к данным вопросов и ответов. Он также содержит переопределение метода toString(), чтобы предоставить строковое представление объекта QuestionAndAnswer.

3.2 Маршрутизация в приложении

В Android Studio на Java для маршрутизации (навигации) между различными экранами (фрагментами или активностями) в приложении можно использовать различные подходы. Ниже приведены два из них:

1.Использование Intent:

- Создайте новый объект Intent, указав текущий контекст (Activity) и целевую активность или фрагмент.
- Добавьте дополнительные данные в Intent, если необходимо.
- Вызовите метод startActivity(Intent) для запуска новой активности или метод startActivityForResult(Intent, int) для запуска активности с ожиданием результата.
- В целевой активности или фрагменте вы можете получить переданные данные из Intent и выполнить соответствующие операции.

2.Использование Navigation Component:

Включите зависимость на Navigation Component в файле build.gradle (Module-level).

- Создайте файл ресурсов навигации (например, navigation.xml), в котором определите различные навигационные графы и связи между ними.
- В XML-разметке текущего фрагмента или активности добавьте элемент NavController для связи с навигационным графом.
- Используйте метод navigate() для перехода на другой фрагмент или активность, указав соответствующий идентификатор действия.

Мы выберем использование Intent.

3.3 Экраны приложения

Экраны играют важную роль в разработке приложений для Android и имеют следующую важность:

Предоставление пользовательского интерфейса: Экраны служат для отображения пользовательского интерфейса (UI) и взаимодействия с пользователем. Они позволяют представить информацию, функциональность и элементы управления таким образом, чтобы пользователи могли легко понять и взаимодействовать с приложением.

Навигация и маршрутизация: Экраны определяют структуру и навигацию в приложении. Они позволяют пользователям перемещаться между различными разделами, функциями и задачами в приложении. Правильная организация экранов обеспечивает понятную и интуитивную навигацию, что улучшает пользовательский опыт.

Разделение функциональности: Приложения могут состоять из нескольких экранов, каждый из которых выполняет определенную функцию или отображает определенную информацию. Разделение функциональности между экранами помогает управлять сложностью приложения, обеспечивает модульность и повышает повторное использование кода.

Адаптивность и реактивность: Экраны позволяют приложению адаптироваться к различным размерам экранов и ориентациям устройств. Разработчики могут создавать разные макеты экранов для поддержки разных типов устройств и обеспечивать отзывчивость приложения при изменении размеров экрана или ориентации.

Визуальное представление и брендирование: Экраны являются основным средством представления визуального стиля и бренда приложения. Они позволяют разработчикам создавать привлекательный и согласованный дизайн, который соответствует целям и ценностям приложения.

Используя правильное проектирование и разработку экранов, разработчики могут создавать эффективные и привлекательные приложения, которые обеспечивают удобство использования и удовлетворение пользователей.

Рисунок 21 – экран Main Activity

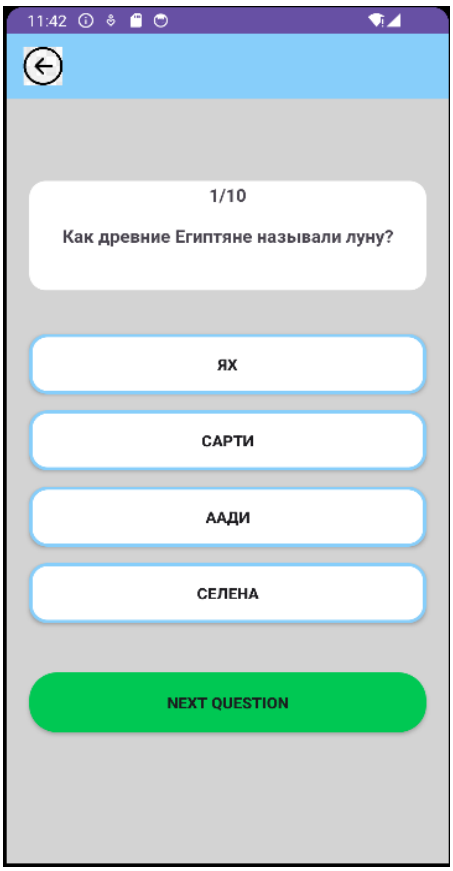


Рисунок 22 – экран Quiz Activity

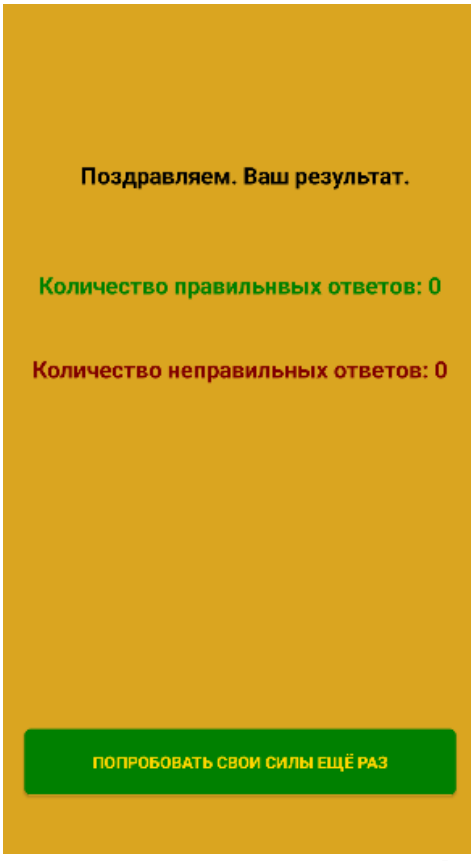


Рисунок 23 – экран Quiz Result

Заключение

В ходе выполнения данной курсовой работы была достигнута её цель, которая заключалась в разработке мобильного приложения-викторины.

Так в ходе работы был проведён анализ предметной области, анализ рынка предложений конкурентов, выявлены их слабые и сильные стороны.

Проведен анализ инструментов и технологий разработки мобильных приложений.

Таким образом, задачи курсовой работы были выполнены, мы разработали приложение-викторину, которое можно установить на мобильные устройства, установив APK файл. Код разработки находится на Github в репозитории.

Список использованных источников

SQLite — самая простая база данных, которая работает везде [Электронный ресурс]. – Режим доступа: https://habr.com/ru/articles/596183/ Дата доступа: 09.12.2023.

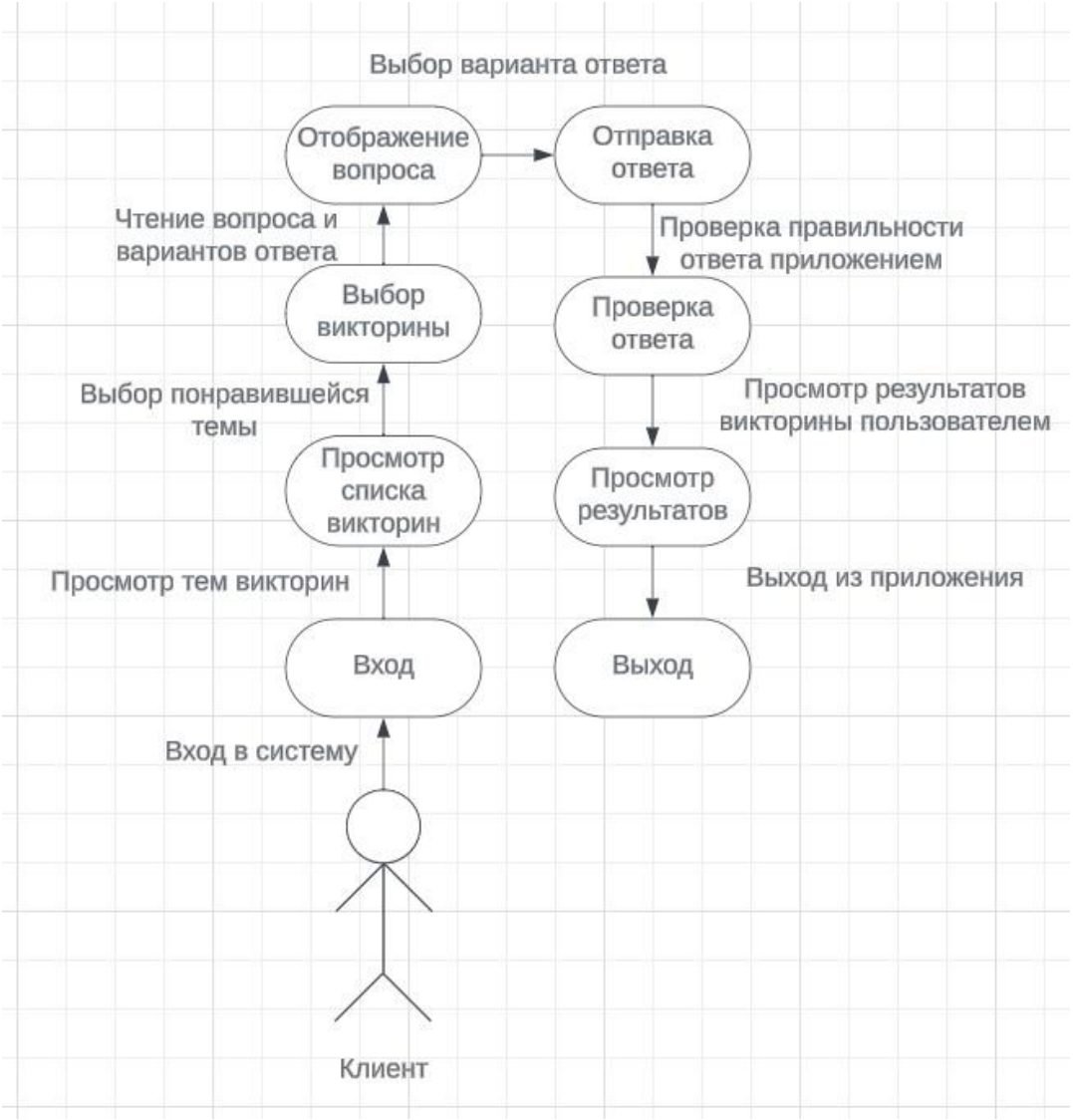
Android Studio — официальная интегрированная среда разработки (IDE) для разработки приложений Android [Электронный ресурс]. – Режим доступа: https://developer.android.com/studio. Дата доступа: 07.12.2023.

Язык программирования Java — это высокоуровневый язык программирования общего назначения [Электронный ресурс]. – Режим доступа: https://www.java.com/ru/ Дата доступа: 10.12.2023.

Github — это веб-платформа для хостинга и совместной работы над проектами разработки программного обеспечения. [Электронный ресурс]. – Режим доступа: https://github.com/KiRiLMarchenko/kursovaia_android Дата доступа: 20.12.2023.

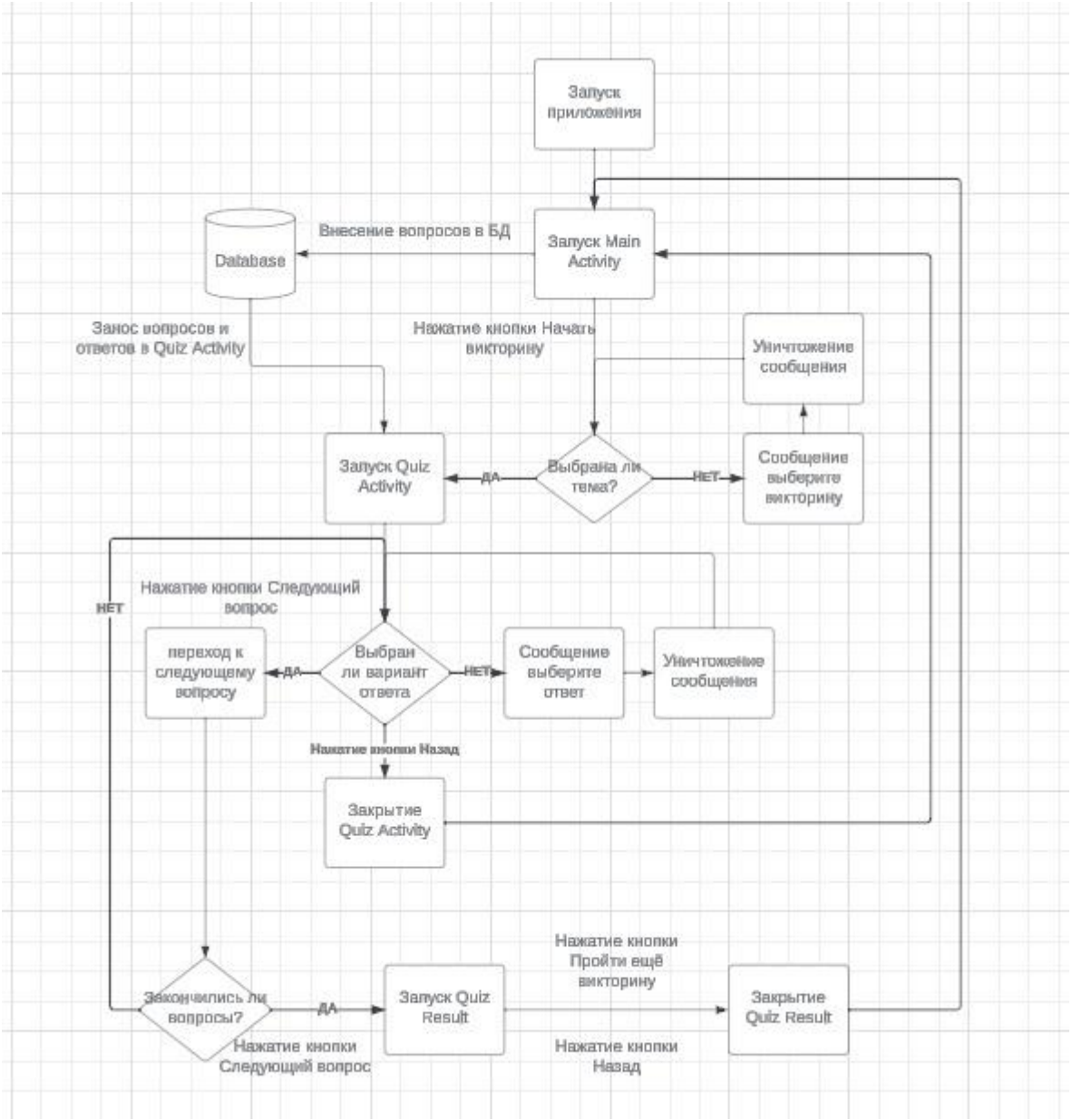
Приложение А

Диаграмма прецедентов



Приложение Б

Схема взаимодействия



Приложение В

Заключение о результатах плагиат-проверки текста курсовой работы