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

Кафедра Прикладная математика и информатика
(наименование)

02.03.03 Математическое обеспечение и администрирование информационных систем
(код и наименование направления подготовки / специальности)

Мобильные и сетевые технологии
(направленность (профиль) / специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему «Разработка мобильного приложения обучающей игры»

Обучающийся С.А. Яцын _____
(Инициалы Фамилия) (личная подпись)

Руководитель канд. техн. наук, доцент, Н.А. Сосина
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Консультант канд. пед. наук, доцент, А.В. Егорова
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Тольятти 2025

Аннотация

Тема выпускной квалификационной работы: «Разработка мобильного приложения обучающей игры».

Целью работы является создание мобильного приложения, предназначенного для обучения с использованием игровых механик.

Во введении обоснована актуальность темы, связанная с необходимостью разработки инструмента для повышения вовлеченности молодых пользователей в образование.

В первой главе проведен анализ предметной области, изучены особенности образовательных приложений и сформулированы требования к разрабатываемому продукту.

Во второй главе описаны этапы проектирования, включая выбор технологий, разработку диаграмм классов и состояний, а также создание макета пользовательского интерфейса.

Третья глава посвящена реализации приложения. Разработаны основные экраны, настроена база данных для хранения данных пользователей и результатов. Проведено тестирование, подтвердившее корректность работы всех модулей.

В заключении представлены выводы о выполненной работе, достигнутых результатах и перспективах развития приложения. Основным результатом стало готовое приложение, обеспечивающее выполнение образовательных заданий, учет очков и формирование рейтинга.

Представленная работа включает 45 страниц, 24 рисунка, 4 таблицы и список литературы из 20 источников, включая литературу на английском языке.

Abstract

Topic of the Thesis: "Development of a Mobile Learning Game Application"

The aim of the work is to create a mobile application designed for interactive learning using gamification mechanics and competitive elements.

The introduction justifies the relevance of the topic, driven by the need to develop an educational tool to enhance user engagement.

The first chapter analyzes the subject area, examines the characteristics of educational applications, and formulates requirements for the developed product.

The second chapter describes the design stages, including the selection of technologies, development of class and state diagrams, and creation of the user interface layout.

The third chapter focuses on the implementation of the application. Key screens were developed, and a SQLite database was configured to store user data and results. Testing was conducted, confirming the correct operation of all modules.

The conclusion presents findings on the completed work, achieved results, and prospects for further development. The main outcome is a fully functional application that enables the completion of educational tasks, tracking of points, and generation of rankings.

The presented work comprises 45 pages, 24 figures, 4 tables, and a bibliography of 20 sources, including literature in English.

Оглавление

Введение

В современном мире технологии стремительно внедряются во все сферы, включая образование и цифровые сервисы. Это создаёт потребность в разработке образовательных мобильных приложений, которые делают процесс обучения более интересным и удобным. Актуальность данной работы обусловлена необходимостью разработки образовательного приложения, интегрируемого с платформой, используемой в Школе №73, для поддержки интерактивного обучения.

Объектом исследования является процесс создания мобильного приложения для учеников Школы №73 на платформе Android, в котором используются игровые механики.

Предметом исследования являются методы и технологии, используемые при создании приложения.

Цель работы состоит в создании мобильного обучающего приложения с игровыми механиками для школы, которое повысит вовлеченность учащихся в образовательный процесс.

Для достижения поставленной цели необходимо выполнить следующие задачи:

– проанализировать образовательные процессы школы, определить требования к приложению;

– обосновать выбор инструментов для реализации приложения;

– спроектировать базу данных для хранения информации о пользователях;

– спроектировать пользовательский интерфейс

– реализовать функциональность приложение;

– провести тестирование и отладку приложения.

Выпускная квалификационная работа состоит из трех основных раздела, описывающих этапы создания и тестирования приложения.

В первой главе проводится анализ предметной области, изучение образовательных процессов Школы №73, с использованием CASE-средства, выбранного на основе сравнения доступных инструментов. На основе этого формируются требования к приложению, а также создается диаграмма использования, определяющая взаимодействие пользователей с системой.

Во второй главе определяются инструменты разработки. Описывается проектирование приложения, разработка диаграмм состояний и классов, а также макета интерфейса, ориентированного на удобство.

Третья глава посвящена реализации приложения. В данной главе описывается настройка проекта, создание экранов, настройка базы данных, тестирование, направленное на проверку корректности работы всех модулей программы.

Глава 1 Описание предметной области

1.1 Краткая характеристика предприятия

Школа №73 имени Героя Советского Союза Н.Ф.Карацупы предоставляет основные общеобразовательные программы начального, основного и среднего общего образования

Школа осуществляет множество функций, такие как обеспечение образованием, организация учебной, методической работы, воспитание. Структурная схема Школы № 73 описана на рисунке 1.

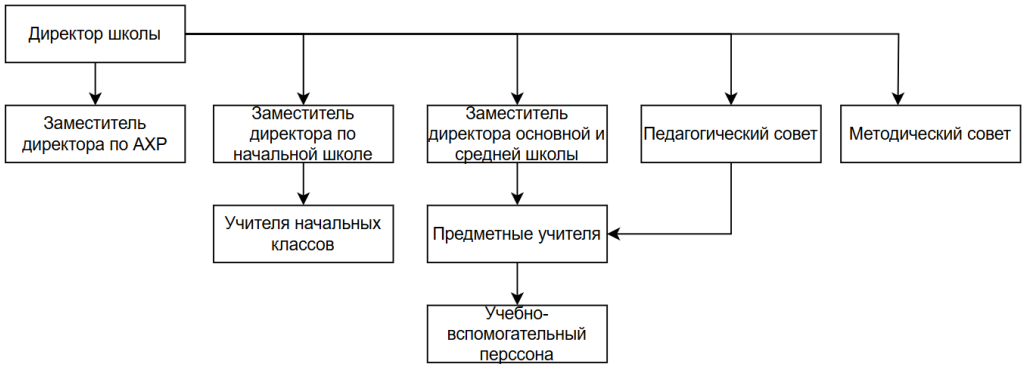


Рисунок 1 – Структурная схема Школы №73

Школа № 73 является пунктом для проведения ЕГЭ. Основными обязанностями является настройка и подключение компьютерной техники. Проверка стабильной работы локальной сети и бесперебойной работы выхода в интернет.

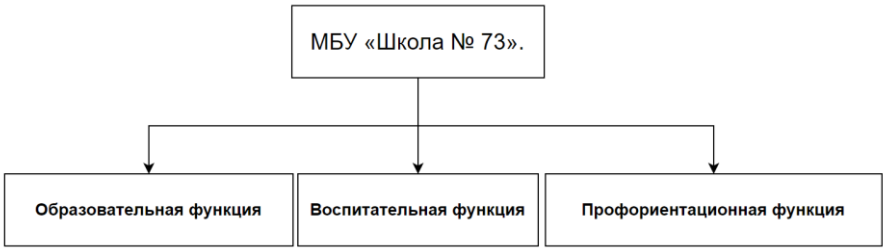


Рисунок 2 – Функции Школы № 73

Функциональные обязанности школы включают в себя:

- обеспечение образовательными программами, соответствующие государственному стандарту;
- формирование нравственных и культурных ценностей учеников;
- помощь в выборе профессионального пути, предоставление о информации о дальнейших возможностях образования и трудоустройства.

В Школе №73 строго соблюдаются правила техники безопасности, особенно при работе с компьютером. Непрерывное время за компьютером не должно превышать двух часов. В перерывах рекомендуется выполнять упражнения для глаз и тела, чтобы снизить напряжение, улучшить кровообращение и предотвратить последствия длительной работы за компьютером.

Основные требования к разработчику в Школе №73:

- владение языками программирования, применяемыми для мобильной разработки.
- умение работать с интегрированными средами разработки;
- знание инструментов и библиотек для разработки;
- понимание принципов проектирования интерфейсов и создания комфортного пользовательского опыта.

1.2 Сравнение инструментов CASE для проектирования бизнес-процессов

«Средства автоматизации разработки программ (CASE-средства) – инструменты автоматизации процессов проектирования и разработки программного обеспечения для системного аналитика, разработчика ПО и программиста.

Основной целью CASE-технологии является разграничение процесса проектирования программных продуктов от процесса кодирования и последующих этапов разработки, максимально автоматизировать процесс разработки. Для выполнения поставленной цели CASE-технологии используют два принципиально разных подхода к проектированию: структурный и объектно-ориентированный» [8].

Среди современных CASE-средств, применяемых для визуализации и моделирования информационных систем, можно выделить три программных решения, каждое из которых обладает рядом функциональных особенностей и преимуществ.

Diagrams.net – это средство для построения разнообразных диаграмм, которое можно использовать как онлайн, так и без подключения к интернету. Оно поддерживает создание разных типов схем: от обычных блок-схем и UML-диаграмм до организационных и сетевых структур. Приложение не требует установки, можно открыть его в браузере или использовать как программу, позволяя сохранять файлы в облаке.

Microsoft Visio представляет собой мощный инструмент для построения диаграмм различной сложности, предназначенный прежде всего для корпоративного использования. Одно из основных функций данного средства является двусторонняя работа с данными, пользователь может визуализировать существующие данные, а также генерировать отчеты и аналитические материалы на основе построенных диаграмм.

Ramus – это программа, разработанная для построения логических и структурных схем, подходящее для анализа образовательных процессов. С использованием Ramus можно создавать диаграммы потоков данных, что позволяет наглядно представить функционирование информационной системы.

Таблица 1 – Сравнение CASE средств.

Критерий	Ramus	Diagrams.net	Visio
Поддержка работы без интернета.	+	+	-
Поддержка различных типов диаграмм.	-	+	+
Бесплатная программа.	-	+	-
Поддержка облачных хранилищ.	-	+	-

Анализ критериев, представленных в таблице, демонстрирует, что среди рассмотренных CASE-средств Diagrams.net является наиболее подходящим, удовлетворяя всем основным требованиям.

1.3 Анализ бизнес процессов (IDEF0)

Бизнес-процессы представляют совокупность последовательных и взаимосвязанных действий, они охватывают все этапы работы: от планирования и организации до реализации и контроля результатов. В процессе участвуют как люди, так и технические средства, что позволяет наладить слаженную работу всех подразделений компании.

Главная задача бизнес-процессов заключается в повышении эффективности деятельности, снижении затрат и улучшении качества продукции или услуг.

«IDEF0 (Integration Definition for Function Modeling) – это семейство методологий и нотаций, разработанных для моделирования различных аспектов деятельности организаций и систем. Изначально создано в рамках программы ICAM (Integrated Computer Aided Manufacturing) для ВВС США, целью которой было улучшение производственных процессов. Сегодня методологии IDEF0 широко применяются в различных отраслях и бизнес-задачах: например, системном и бизнес-анализе, документировании процессов, управлении качеством на производстве, обучении персонала» [4].

Контекстная диаграмма процесса игрового обучения отражена на рисунке 3.

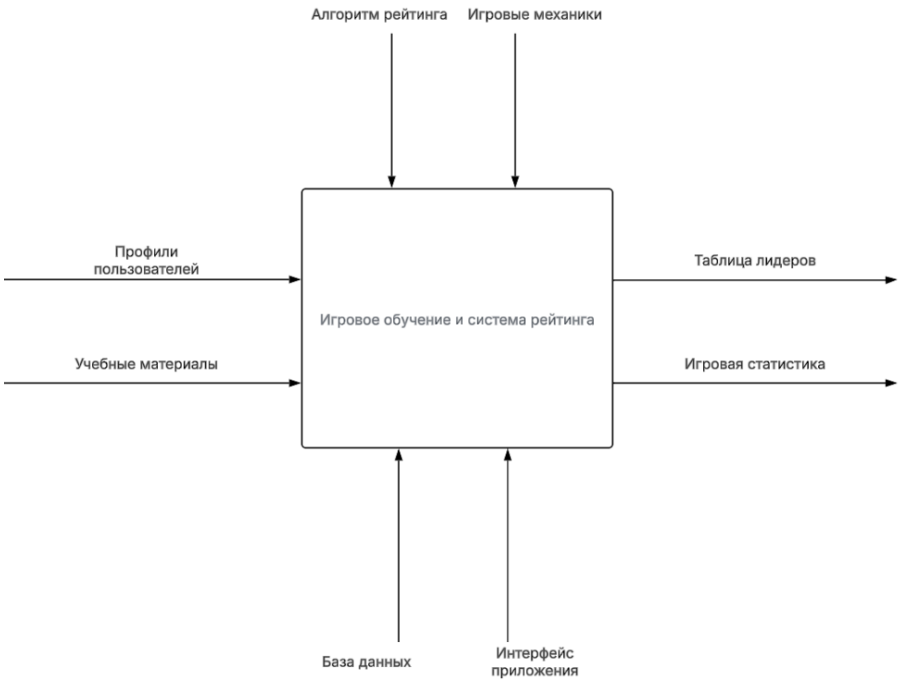


Рисунок 3 – Контекстная диаграмма процесса игрового обучения

Для более детального изучения данной диаграммы представлена декомпозиция блока А0 на рисунке 4.

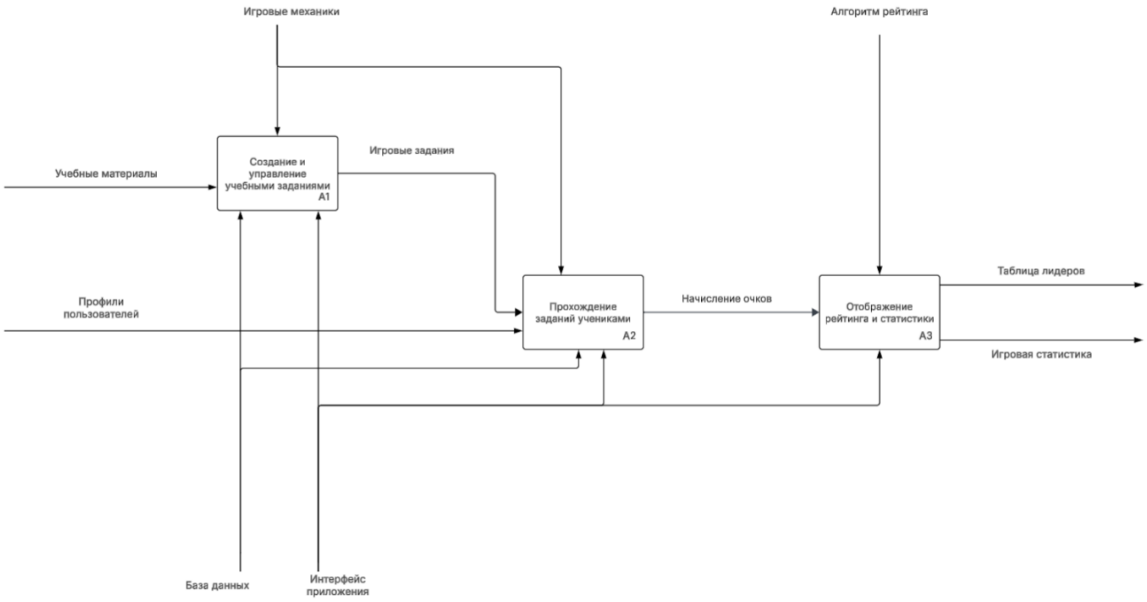


Рисунок 4 – Декомпозиция блока А0 модели «Игровое обучение»

Для наглядного отображения того, как различные пользователи взаимодействуют между собой, воспользуемся диаграммой взаимодействия, изображенной на рисунке 5. Она помогает выявить слабые места, оптимизировать процессы и четко распределить роли, что улучшает координацию и минимизирует ошибки.

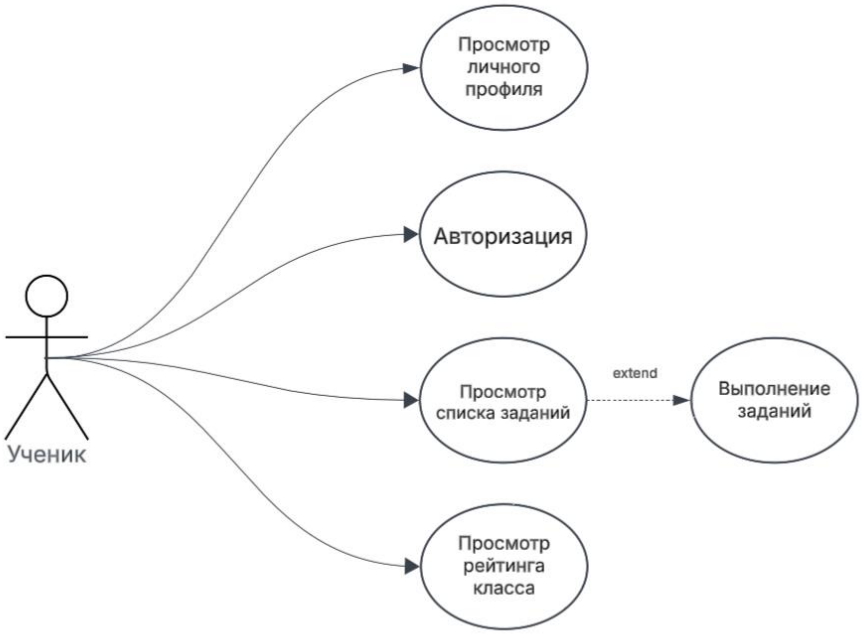


Рисунок 5 – Диаграмма взаимодействия пользователей

Данная диаграмма наглядно демонстрирует роли, которые принимает пользователи и последовательность запросов, передаваемых каждой из ролей, показывая какие функциональные возможности доступны разным категориям пользователей.

1.4 Оценка альтернативных приложений для планирования

На современном рынке мобильных приложений существует широкий выбор решений, ориентированных на образовательные цели. Эти приложения предлагают различные подходы к обучению, включая интерактивные методики, игровой формат и индивидуальный подход к каждому пользователю. Для анализа были выбраны наиболее популярные решения из крупного магазина приложений – Google Play.

«Google Play – магазин приложений, а также игр и книг от компании Google, позволяющий сторонним компаниям предлагать владельцам устройств с операционной системой Android устанавливать и приобретать различные приложения. Разработку приложений для Android можно вести на языке Java. Основной средой разработки сейчас является Android Studio» [7].

В рамках исследования мобильных приложений для обучения был проведен анализ существующих решений с учетом ключевых функций, которые могут быть полезны для эффективного процесса обучения. Были выделены следующие критерии:

- интерактивность и игровой формат;
- система рейтинга с таблицей лидеров;
- аналитика прогресса;
- удобство использование интерфейса.

Критерий использования интерактивных и игровых элементов имеет значение в обучении школьников младших и средних классов. Включение игры в задания делает процесс выполнения более похожим на развлечение, что помогает увеличить интерес к занятиям. Ученик, вовлечённый в игровой формат, воспринимает обучение как занятие, которое вызывает интерес, а не как простую обязанность. Это положительно сказывается на активности и стремлении достигать поставленных целей.

Критерий наличия рейтинга с таблицей лидеров важен для формирования у учеников желания развиваться и участвовать в соревновании.

Когда ученик видит свои результаты в сравнении с другими, это помогает вовлечься в обучение. Позиция в общем списке лидеров побуждает стараться добиться лучшего результата, улучшать знания и быть активнее. Критерий анализа прогресса полезен в учебном процессе, так как помогает отслеживать успехи и замечать темы, которые требуют больше внимания. Критерий простоты использования интерфейса играет немаловажную роль при создании приложений для школьников. Если интерфейс перегружен или непонятен, это мешает учиться. А если дизайн простой, понятный и удобно устроен, это помогает лучше воспринимать материал и повышает эффективность.

ClassDojo в Google Play, App Store – образовательная платформа, разработанная для организации учебного процесса с применением игровых методов мотивации. Приложение позволяет учителю создавать задания для учеников, отслеживать их выполнение, а также поощрять активность с помощью системы баллов и наград. Встроенные игровые механики, такие как виртуальные аватары, а также индивидуальные и групповые достижения, способствуют вовлечению учащихся в процесс обучения.

Google Classroom в Google Play, App Store – образовательное приложение, предназначенное для эффективной организации учебного процесса и взаимодействия между учителями и учениками. Приложение позволяет преподавателю создавать курсы, выдавать задания, контролировать сроки их выполнения, а также выставлять оценки в едином цифровом пространстве.

Для перечисленных выше приложений проведем сравнение на соответствие заданным критериям, результаты которого можно увидеть в таблице 2.

Таблица 2 – Сравнение мобильных приложений

Мобильные приложения	Интерактивность и игровой формат	Система рейтинга с таблицей лидеров	Аналитика прогресса	Удобство использования интерфейса
ClassDojo	+	-	+	+
Google Classroom	-	-	+	+

В результате, ни одно из рассмотренных приложений не является полностью универсальным, показывая необходимость создания приложения, сочетающего в себе сильные стороны обоих платформ.

1.5 Определение требований к приложению

Для разработки мобильного обучающего приложения, ориентированного на учеников Школы №73, определены следующие требования, обеспечивающие функциональность и техническую устойчивость системы:

1. Использование базы данных для хранения информации об учащихся, результатах выполнения заданий и накопленных баллах.
2. Система аутентификации, обеспечивающая идентификацию ученика по логину и паролю.
3. Каждый ученик должен иметь собственный профиль, содержащий свои персональные данные, статистику выполненных заданий, количество баллов.
4. В приложении должна быть реализована таблица лидеров, отображающая результаты всех учащихся. Рейтинг должен формироваться на основе количества набранных баллов.

5. Приложение должно включать в себя обучающие игры, направленные на закрепление учебного материала. В процессе игры пользователю предлагаются случайно сгенерированные вопросы или примеры. Набранные баллы должны обновляться в базе данных.

Технические требования:

– совместимость с мобильными операционными системами Android;
– высокая производительность;
– эффективное управление памятью и хранением данных

Выводы по первой главе.

В первой главе рассмотрены теоретические и практические аспекты, необходимые для разработки мобильного приложения. Проведен анализ образовательных процессов, изучены доступные на рынке приложения, а также сформулированы основные требования к разрабатываемому приложению. Итоги анализа стали фундаментом для дальнейших этапов проектирования и реализации.

Глава 2 Проектирование мобильного приложения

2.1 Определение средств для разработки

Существуют различные платформы и инструменты для мобильной разработки на Android, каждая из которых имеет свои особенности и подходит для разных задач, среди которых можно выделить:

- Android Studio,
- Xamarin,
- React Native,
- Flutter.

Рассмотрим подробнее каждую из перечисленных платформ для разработки приложений:

«Android Studio – это официальная интегрированная среда разработки (IDE) для операционной системы Android от компании Google. Она основана на программном обеспечении IntelliJ IDEA от JetBrains и специально предназначена для разработки под Android. Программа доступна для загрузки на операционных системах Windows, macOS и Linux.

Android Studio пришла на смену инструментам разработки Eclipse Android Development Tools (E-ADT) и стала основным инструментом для создания нативных (локальных) Android-приложений. Android Studio отлично подходит для разработки на языках Java и Kotlin» [9].

Android Studio поддерживает языки Java, Kotlin и C++, оснащена мощным редактором кода, визуальным редактором интерфейсов, продвинутыми средствами тестирования и отладки, а также гибкой системой сборки Gradle. Благодаря регулярной поддержке от Google, наличию встроенных эмуляторов и инструментов аналитики, Android Studio обеспечивает комфортную и продуктивную разработку мобильных приложений на всех этапах.

«Xamarin – это платформа для разработки кроссплатформенных мобильных приложений на языке программирования C#. Она позволяет разработчикам создавать приложения для iOS, Android и Windows, используя общий код.

С помощью Xamarin разработчики могут использовать единое API для доступа к устройству, базам данных и другим ресурсам, что упрощает процесс разработки и поддержки приложений.

Xamarin позволяет использовать существующий код на C# и .NET для создания мобильных приложений, что делает его привлекательным инструментом для разработчиков, знакомых с этими технологиями» [10].

Основными достоинствами Xamarin являются возможность повторного использования значительной части кода между платформами, интеграция с Visual Studio, доступ к нативным API каждой операционной системы, а также высокое качество производительности приложений благодаря компиляции в нативный код. Платформа подходит как для разработки пользовательских интерфейсов, так и для реализации сложной бизнес-логики в одном общем проекте.

«React Native – это программный фреймворк с открытым исходным кодом для разработки пользовательских интерфейсов. Он используется для разработки приложений для Android, Android TV, iOS, macOS, tvOS, Web, Windows и UWP, позволяя разработчикам использовать фреймворк React в сочетании с возможностями нативных платформ. React Native используется для разработки приложений для Android и iOS» [11].

React Native позволяет разработчикам создавать приложения для разных платформ, используя один код. Благодаря этому, разработчики могут строить высокопроизводительные приложения, которые используют нативные функции устройств, такие как камеры, сенсоры или GPS, при этом поддерживая кроссплатформенность и ускоряя процесс разработки.

Flutter – это фреймворк с открытым исходным кодом для создания кроссплатформенных приложений, разработанный Google. Он позволяет

разрабатывать приложения для мобильных устройств. В отличие от других фреймворков, Flutter использует собственный движок для рендеринга интерфейса, что дает разработчикам полный контроль над визуальными элементами. Flutter активно используется для создания современных приложений, предлагая быстрый и удобный процесс разработки с горячей перезагрузкой, что значительно ускоряет тестирование и отладку.

Для более точной оценки проведем сравнение по критериям для средств разработки приведенных выше. Наглядное сравнение представлено в таблице 3.

Таблица 3 – Сравнение средств разработки

Критерии сравнения	Android Studio	Visual Studio & Xamarin	Flutter	React Native
Интеграция с Android	Официальная IDE от Google, полная интеграция с Android SDK.	Поддерживает Android, но требует дополнительных настроек.	Поддержка Android, но не такая глубокая, как у Android Studio.	Требуются плагины и мосты для доступа к нативным функциям.
Обучение и документация	Подробная документация, обучающие материалы от Google.	Документация от Microsoft, но меньше примеров для Android.	Документация от Google, но язык Dart менее распространен.	Много ресурсов, но документация может быть разрозненной
Инструменты разработки	Встроенный эмулятор, профилировщик, Gradle	Интеграция с Visual Studio, но меньше специфичных инструментов.	Собственный редактор и инструменты, но менее мощные, чем у Android Studio.	Зависит от внешних инструментов и редакторов кода.
Стоимость	Полностью бесплатная IDE и инструменты.	Базовая версия бесплатна, но некоторые функции платные.	Открытый исходный код, но могут потребоваться платные плагины.	Полностью открытый исходный код.

По итогам сравнения в таблице, Android Studio выигрывает по большинству критериям для Android, превосходя своих конкурентов в производительности, интеграции с Android, инструментах разработки, документации, а также предлагая полностью бесплатный доступ.

Относительно выбора языка программирования для разработки под Android Studio, актуальными языками являются Java и Kotlin, каждый обладающий своими преимуществами. Java – один из старейших и проверенных языков программирования, обладающий множество библиотек и инструментов, что делает его основой для сложных и масштабных проектов. Kotlin, в свою очередь, «более лаконичный и типобезопасный, чем Java, и более простой. Язык полностью совместим с Java, что позволяет Java-разработчикам постепенно перейти к его использованию; в частности, язык также встраивается в Android, что позволяет для существующего Android-приложения внедрять новые функции на Kotlin без переписывания приложения целиком» [11].

Таким образом, в качестве среды разработки был выбран Android Studio, а в качестве языка программирования было решено использовать Kotlin, позволяющий писать более компактный код.

2.2 Проектирование пользовательского интерфейса

Проектирование пользовательского интерфейса начинается с диаграммы состояний – это тип диаграммы моделирования UML, используемая для описания поведения объекта или системы, помогая понять, как пользователь будет взаимодействовать с приложением на разных этапах его работы, например, для моделирования поведения экранов. Данная диаграмма построена в приложении Draw.io, в модели отражаются ключевые экраны мобильного приложения, такие как, авторизация, профиль пользователя, рейтинг класса и игровые задачи. Диаграмму можно увидеть на рисунке 6.

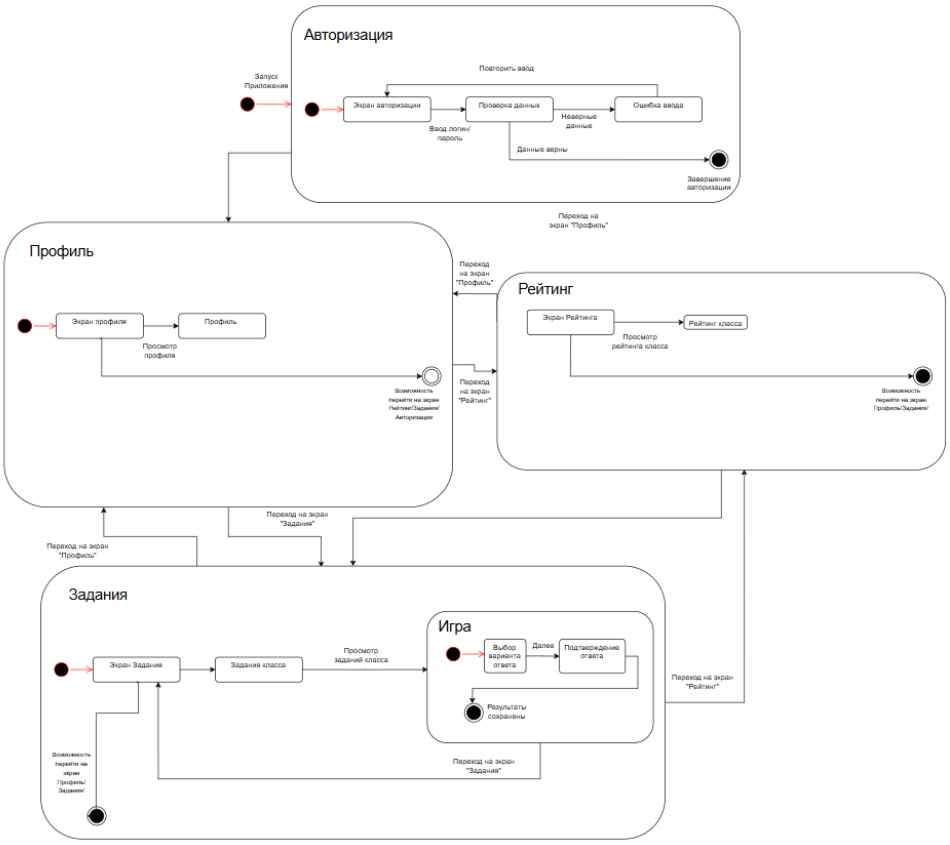


Рисунок 6 – Диаграмма состояний мобильного приложения

После запуска приложения, пользователь переходит на экран авторизации, где требуется ввести учетные данные – логин ученика и пароль. После ввода корректных данных осуществляется переход к экрану профиля, где отображается его персональные данные, а также становится доступна навигационная панель для перехода к заданиям, рейтингу.

Раздел «Рейтинг» представляет пользователю возможность ознакомиться с результатами и достижениями своих одноклассников, способствуя соревновательной мотивации, отсюда возможен возврат в профиль или переход к заданиям.

Экран с заданиями позволяет пользователю ознакомиться с перечнем доступных ему заданий в виде мини-игр, где от пользователя требуется решать задания в игровой форме. Результаты выполнения сохраняются, что в дальнейшем отражается в рейтинге и профиля пользователя.

Используя диаграмму состояний можно построить макет пользовательских экранов мобильного приложения, содержащий функциональные элементы интерфейса, отражающие логику работы системы. Данный макет приведен на рисунке 7.

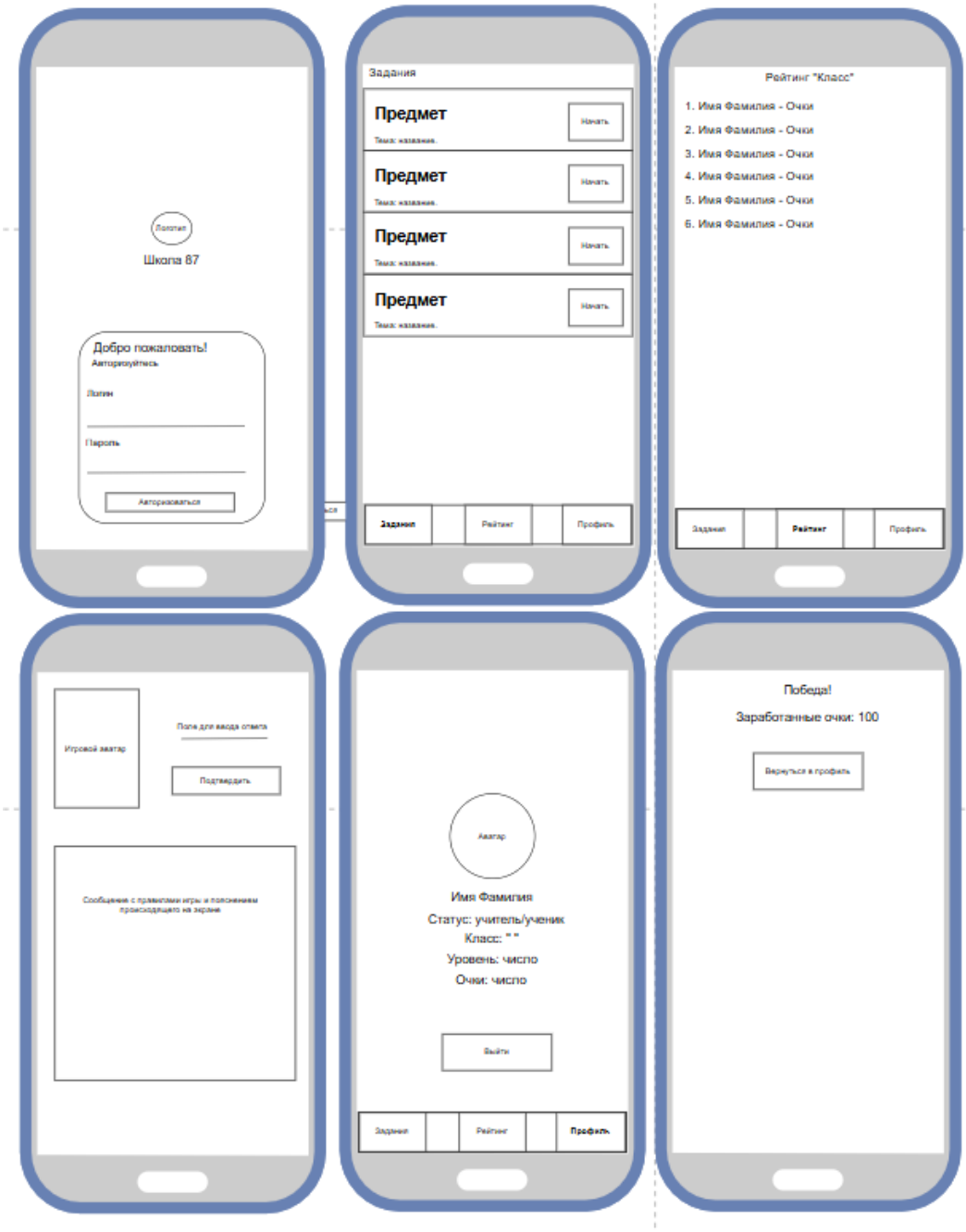


Рисунок 7 – Макет мобильного приложения

На данном изображении показаны 6 основных пользовательских экрана, демонстрирующие ключевые сценарии: авторизацию, просмотр заданий, игровой процесс, рейтинг, завершение игры и личный профиль. В нижней части макета, прикреплена панель навигации, позволяющая пользователю в любой момент перейти в нужный ему экран.

2.3 Описания средств и подходов для решения поставленных задач

Для реализации мобильного приложения, предоставляющего образовательные функции в игровой форме, выбрана среда разработки Android Studio и язык программирования Kotlin. В качестве системы хранения данных было принято решение использовать встроенную в Android базу данных SQLite.

Применение SQLite позволяет реализовать хранение информации о пользователях, их прогрессе, количестве набранных очков. Все данные структурированы в виде таблиц, и доступ к ним осуществляется с помощью SQL-запросов. Такая архитектура упрощает анализ учебной статистики и даёт возможность в будущем масштабировать систему, добавляя, например, уровни сложности или дополнительные виды заданий.

Для управления базой данных реализован вспомогательный класс, отвечающий за создание и обновление таблиц, выполнение операций вставки, чтения и удаления данных. Каждое изменение в ходе игрового процесса, например, завершение уровня, набор очков, сопровождается соответствующей записью в базу данных.

Пользовательский интерфейс приложения разработан с использованием компонентов AndroidX и Material Design, что делает его современным, удобным и визуально привлекательным. Особое внимание было уделено элементам взаимодействия – кнопкам, полям ввода, анимациям и иконкам.

Для анимации персонажей и динамики игрового процесса используется механизм Handler с таймером, который позволяет переключать спрайты с заданной периодичностью, создавая эффект движения.

На рисунке 8 представлена диаграмма развертывания мобильного приложения, отражающая архитектуру хранения и обработки данных

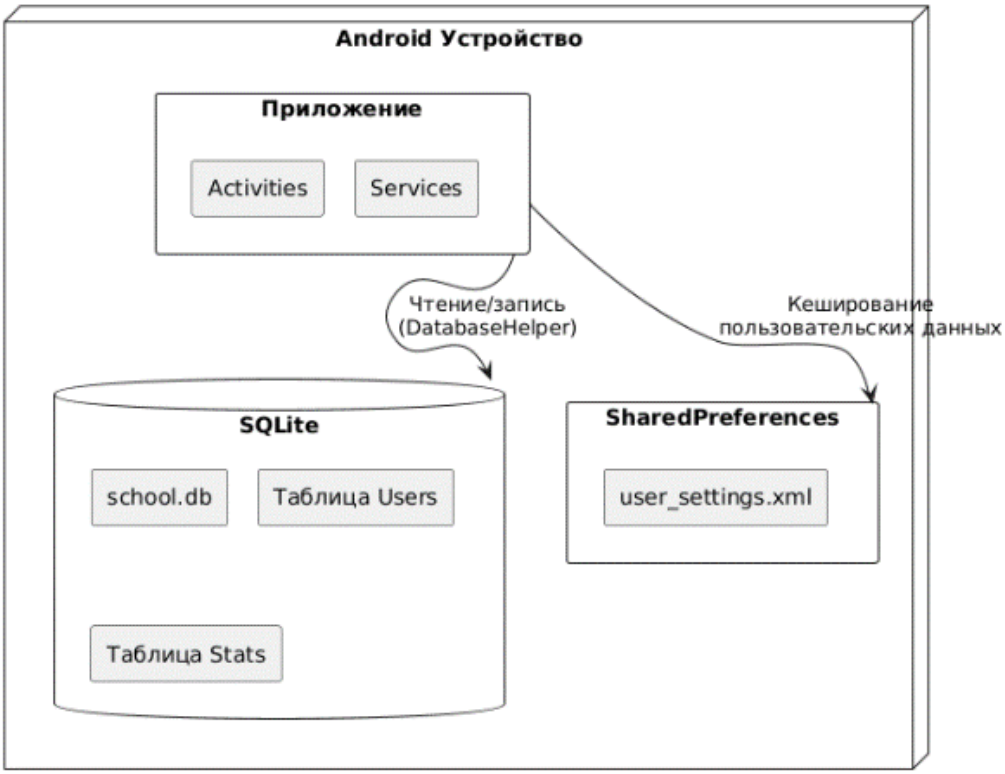


Рисунок 8 – Диаграмма развертывания мобильного приложения

После завершения проектирования архитектуры системы была разработана UML-диаграмма классов, которая наглядно отображает ключевые структурные элементы приложения и их взаимосвязи. Диаграмма классов приложения приведена на рисунке 9.

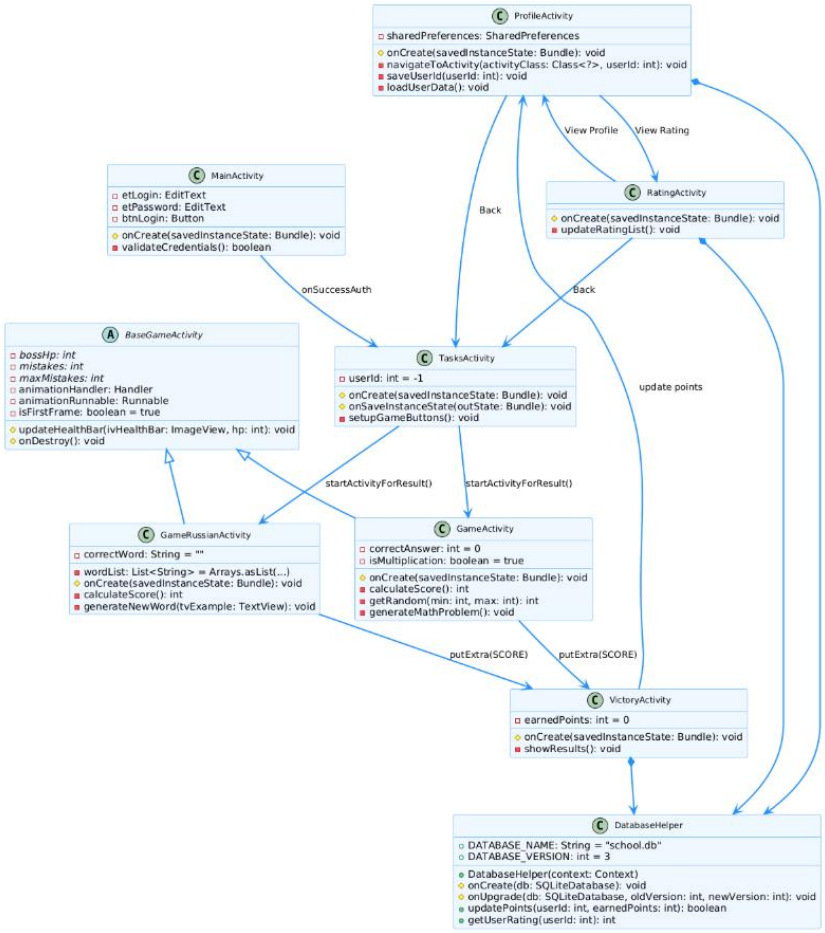


Рисунок 9 – Диаграмма классов

Выводы по главе 2

В ходе выполнения второй главы был проведен анализ инструментов, для разработки мобильных приложений. Была выбрана Android Studio как наиболее подходящая среда для проекта. В качестве языка программирования выбран Kotlin.

В рамках проектирования был создан макет интерфейса и диаграмма состояний, отражающая основные сценарии взаимодействия пользователя с приложением. Также реализована структура хранения данных с использованием встроенной базы данных SQLite. Кроме того, представлена диаграмма классов, демонстрирующая основные элементы архитектуры и связи между ними, что упрощает понимание внутренней структуры приложения.

Глава 3 Реализация мобильного приложения

3.1 Разработка мобильного приложения

В процессе проектирования и реализации обучающего мобильного приложения были разработаны семь основных классов и один вспомогательный класс для взаимодействия с базой данных, оформлены соответствующие XML-файлы макетов пользовательского интерфейса:

- VictoryActivity (Реализация отображения результатов задания);
- TasksActivity (Реализация меню выбора заданий);
- RatingActivity (Реализация таблицы рейтинга класса);
- ProfileActivity (Реализация профиля ученика);
- MainActivity (Реализация авторизации);
- GameRussianActivity (Реализация игрового режима);
- GameActivity (Реализация игрового режима);
- DatabaseHelper (Реализация взаимодействия с SQLite);
- activity_game.xml (Макет игрового режима);
- activity_main.xml (Макет авторизации ученика);
- activity_profile.xml (Макет личного кабинета ученика);
- activity_rating.xml (Макет экрана рейтинга);
- activity_rus_game.xml (Макет игрового режима);
- activity_tasks.xml (Макет списка заданий);
- activity_victory.xml (Макет итогового экрана завершения задания);
- bottom_nav_menu.xml (Реализация навигационного меню).

Рассмотрим класс DatabaseHelper создающий базу данных. Класс DatabaseHelper наследуется от SQLiteOpenHelper, что позволяет управлять созданием и обновлением базы данных. В методе onCreate создаётся таблица Users с полями для хранения логина, пароля, имени, класса и количества очков пользователя. Также на этапе инициализации добавляются тестовые записи с

предварительно заданными данными, фрагмент кода можно увидеть на рисунке 10.

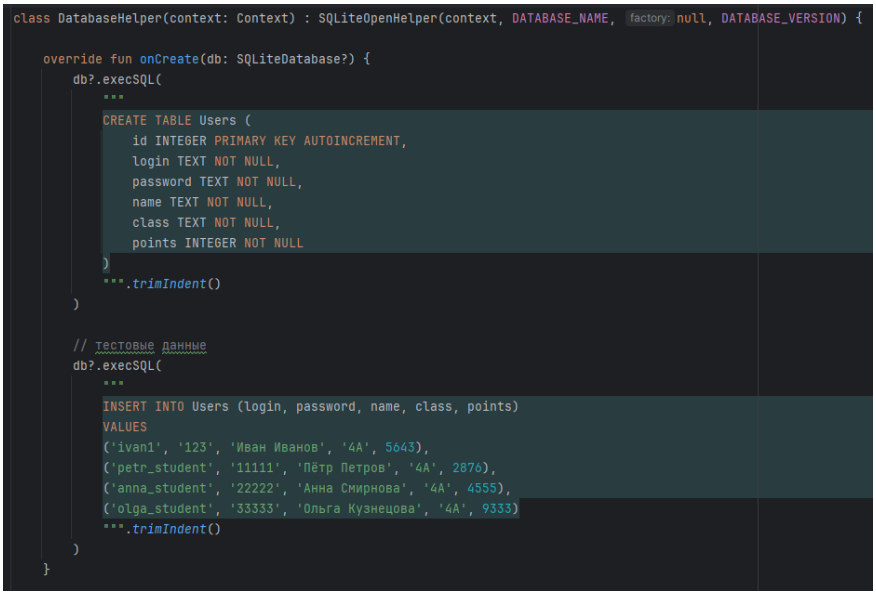


Рисунок 10 – Создание базы данных

Метод updatePoints отвечает за обновление количества очков конкретного пользователя, сначала извлекая текущее количество очков по идентификатору пользователя, затем прибавляя новое значение и обновляет соответствующую запись в таблице, фрагмент изображен на рисунке 11.

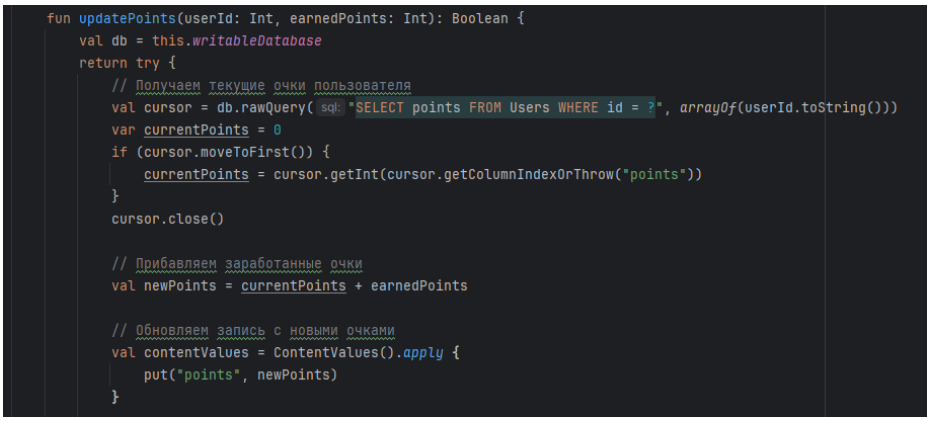


Рисунок 11 – Метод обновления очков

Класс MainActivity, отвечает за процесс авторизации пользователя, при вводе логина и пароля, введенные данные считываются и проверяются на пустоту, если одно из полей не заполнено, отображается соответствующее уведомления. В случае корректного ввода, используется метод rawQuerry для выполнения SQL-запроса, который проверяет наличие пользователя с указанными данными, если пользователь найден, из полученной строки извлекается его ID и сохраняется в SharedPreferences, что позволяет использовать его в других частях приложения. Фрагмент кода можно увидеть на рисунке 12.

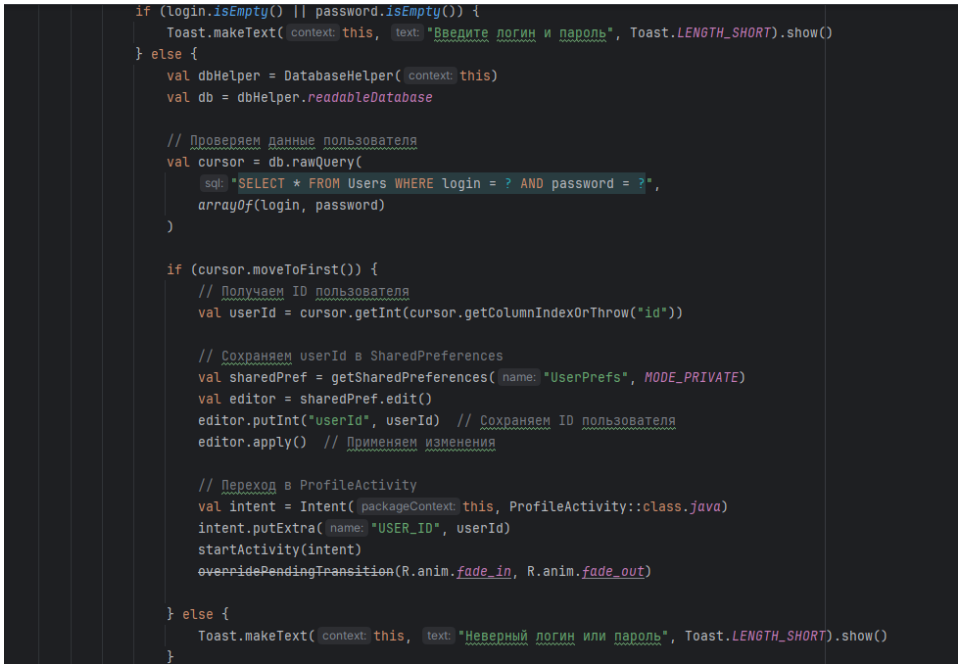


Рисунок 12 – Проверка логина и пароля.

После этого происходит переход к экрану профиля. Класс ProfileActivity показывает данные пользователя, отображает его уровень, обрабатывает выход из аккаунта и позволяет переходить между экранами с помощью нижнего меню навигации. После запуска интерфейса через setContentView настраиваются элементы управления: кнопка выхода, текстовые поля для

имени, класса, очков и уровня, а также изображения аватара и полоски опыта. Для выхода из аккаунта очищаются сохранённые данные SharedPreferences, после чего запускается MainActivity с нужными флагами, чтобы закрыть остальные экраны.

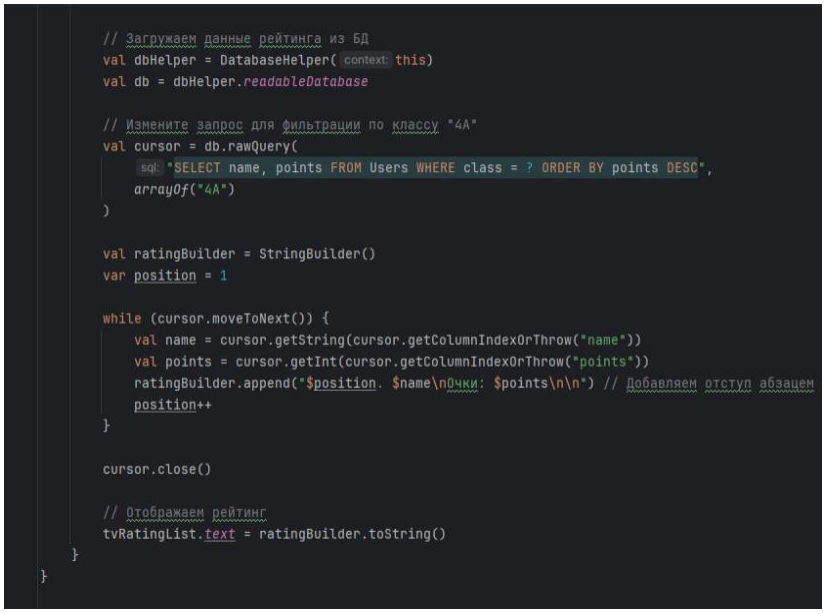
Уровень определяется по очкам: каждые 1000 очков – это новый уровень. Остаток от деления очков на 1000 указывает на прогресс внутри уровня и помогает выбрать подходящее изображение полоски опыта. При ошибках в загрузке информации на экране показывается сообщение об этом.

Нижнее меню BottomNavigationView используется для переключения между экранами. Обработчик нажатий setOnItemSelectedListener вызывает метод navigateToActivity, который запускает нужную активность и передаёт туда ID пользователя, добавляя анимацию перехода. Класс реализации профиля можно увидеть на рисунке 13.

```
// Проверяем корректность ID
if (userId != -1) {
    val dbHelper = DatabaseHelper( context: this)
    val db = dbHelper.readableDatabase

    // Загружаем данные пользователя из базы данных
    val cursor = db.rawQuery(
         sql: "SELECT name, class, points FROM Users WHERE id = ?",
        arrayOf(userId.toString())
    )

    if (cursor.moveToFirst()) {
        val name = cursor.getString(cursor.getColumnIndexOrThrow("name"))
        val userClass = cursor.getString(cursor.getColumnIndexOrThrow("class"))
        val points = cursor.getInt(cursor.getColumnIndexOrThrow("points"))

        // Устанавливаем данные в виджеты
        tvName.text = name
        tvClass.text = "Класс: \"$userClass\""
        tvPoints.text = "Очки: $points"

        // Вычисляем уровень мастерства
        val skillLevel = points / 1000 + 1 // Уровень (1000 очков = следующий уровень)
        tvSkillLevel.text = "Уровень мастерства: $skillLevel"

        // Устанавливаем аватар (ресурс по умолчанию)
        ivAvatar.setImageResource(R.drawable.ic_avatar_placeholder)
    } else {
        // Если пользователь не найден в БД
        tvName.text = "Ошибка: пользователь не найден"
        tvClass.text = ""
        tvPoints.text = ""
        tvSkillLevel.text = ""
    }
    cursor.close()
    db.close()
```

Рисунок 13 – Класс реализации профиля

Класс RatingActivity отвечает за отображение списка учеников по количеству заработанных очков за выполненные задания, подключаясь к базе данных через ранее упомянутый DatabaseHelper. Выполняется SQL-запрос, который выбирает имена и очки всех учеников из текущего класса пользователя, отсортированных по убыванию очков. Результаты формируются в виде строки, где каждому ученику присваивается позиция в списке. Метод для реализации рейтинга можно наблюдать на рисунке 14.



```
        // Загружаем данные рейтинга из БД
        val dbHelper = DatabaseHelper( context: this)
        val db = dbHelper.readableDatabase

        // Измените запрос для фильтрации по классу "4А"
        val cursor = db.rawQuery(
            sql: "SELECT name, points FROM Users WHERE class = ? ORDER BY points DESC",
            arrayOf("4А")
        )

        val ratingBuilder = StringBuilder()
        var position = 1

        while (cursor.moveToNext()) {
            val name = cursor.getString(cursor.getColumnIndexOrThrow("name"))
            val points = cursor.getInt(cursor.getColumnIndexOrThrow("points"))
            ratingBuilder.append("$position. $name\nОчки: $points\n\n") // Добавляем отступ абзацем
            position++
        }

        cursor.close()

        // Отображаем рейтинг
        tvRatingList.text = ratingBuilder.toString()
    }
}
```

Рисунок 14 – Класс для реализации рейтинга

Класс TasksActivity отображает список заданий с краткой информацией и очками за выполнение. Пользователь может выбрать задание и перейти к соответствующей игре. Идентификатор пользователя передаётся через Intent.

Класс GameActivity представляет из себя игру, основанную на решении примеров на умножение и деление. Цель игры заключается в том, что пользователь должен решить сгенерированные примеры, тем самым уменьшая количество очков здоровья противника. У противника есть 10 очков здоровья.

Каждый раз, когда игрок правильно решил пример, здоровье противника уменьшается на 2, а если достигает нуля, то игрок побеждает.

Изображение полоски здоровья обновляется в зависимости от текущего количества здоровья у противника. Для разных значений HP показываются разные изображения полоски здоровья.

При победе в игре подсчитываются очки. Игрок получает базовое количество в размере 100 очков, при отсутствии ошибок, пользователь получит бонус в размере 50 очков. Каждая ошибка уменьшит базовую награду на 10 очков, вплоть до 30. Когда здоровье босса опускается до нуля, игра завершена, и игрок переходит на экран победы, где отображается итоговый счет. На рисунке 14 мы можем увидеть реализацию, перечисленных выше механик.

```
69      private fun calculateScore(): Int {
70          var score = 100 // Базовые очки за победу
71
72          if (mistakes == 0) {
73              score += 50 // Бонус за отсутствие ошибок
74          } else {
75              score -= mistakes * 10 // Штраф за ошибки (максимум -30)
76          }
77
78          return score
79      }
80
81      private fun updateHealthBar(ivHealthBar: ImageView, hp: Int) {
82          val healthBarRes = when (hp) {
83              10 -> R.drawable.ic_healthbar10
84              8 -> R.drawable.ic_healthbar8
85              6 -> R.drawable.ic_healthbar6
86              4 -> R.drawable.ic_healthbar4
87              2 -> R.drawable.ic_healthbar2
88              else -> R.drawable.ic_healthbar0
89          }
90          ivHealthBar.setImageResource(healthBarRes)
91      }
92
93      private fun generateNewTask(tvExample: TextView) {
94          val num1: Int
95          val num2: Int
96
97          if (isMultiplication) {
98              num1 = Random.nextInt( from: 2,  until: 50)
99              num2 = Random.nextInt( from: 2,  until: 15)
100             correctAnswer = num1 * num2
101             tvExample.text = "$num1 × $num2 = ?"
102         } else {
103             correctAnswer = Random.nextInt( from: 2,  until: 11)
104             num2 = Random.nextInt( from: 2,  until: 11)
105             num1 = correctAnswer * num2
106             tvExample.text = "$num1 ÷ $num2 = ?"
107         }
```

Рисунок 14 – Класс GameActivity

Класс VictoryActivity отображает количество набранных очков пользователем и сохраняет результат в базу данных после успешного выполнения задания. На экране также размещена кнопка, по нажатию на которую пользователь может вернуться к профилю. Затем запускается новая активность ProfileActivity, а текущий экран закрывается.

```
class VictoryActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_victory)

        val tvScore = findViewById<TextView>(R.id.tvScore)
        val btnReturn = findViewById<Button>(R.id.btnReturn)

        // Получаем данные из Intent
        val score = intent.getIntExtra( name: "score",  defaultValue: 0)

        // Получаем userId из SharedPreferences
        val sharedPref = getSharedPreferences( name: "UserPrefs", MODE_PRIVATE)
        val userId = sharedPref.getInt("userId", -1) // Если не найден, возвращается -1

        // Устанавливаем очки
        tvScore.text = "Ваши очки: $score"

        // Сохраняем очки в базу данных
        if (userId != -1) {
            val dbHelper = DatabaseHelper( context: this)
            val isUpdated = dbHelper.updatePoints(userId, score)
            if (isUpdated) {
                tvScore.text = "Ваши очки: $score"
            } else {
                tvScore.text = "Ошибка: не удалось обновить очки пользователя."
            }
        } else {
            tvScore.text = "Ошибка: пользователь не найден."
        }
```

Рисунок 15 – Класс VictoryActivity

Для реализации пользовательского интерфейса используются XML-файлы макетов, которые определяют структуру и внешний вид экранов приложения. Каждому экрану приложения прилагается отдельный XML-файл, содержащий компоненты интерфейса в виде: текстовых полей, кнопок, изображений и других элементов. Данные файлы хранятся в папке проекта layout.

– activity_main.xml описывает экран авторизации пользователя, с полями ввода для логина и пароля;

– activity_game.xml описывает экран математической игры, где отображается пример, состояние здоровья босса, а также предусмотрено поле для ввода ответа и кнопка отправки ответа;

– activity_rus_game.xml описывает экран игры правописания русского языка, содержит задание в виде слова с пропущенной буквой, поле для ввода и кнопку подтверждение ответа;

– activity_victory.xml описывает экран результатов пройденного задания, демонстрирующий его заработанные очки с возможностью вернуться в профиль;

– activity_profile.xml описывает экран профиля пользователя, где отображаются его личные данные и кнопка выхода;

– activity_rating.xml описывает экран рейтинга, отображающий список пользователей класса с их именами и очками, отсортированных по убыванию;

– activity_tasks.xml описывает экран с возможностью выбора заданий, содержит список и кнопки позволяющие запустить игру.

На рисунке 16 можно увидеть 7 созданных макетов интерфейса.

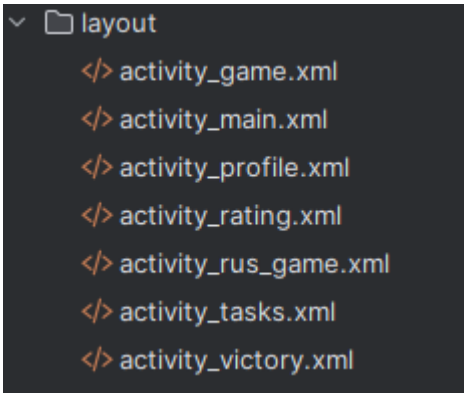


Рисунок 16 – Папка layout с созданными интерфейсами

В папке drawable находятся графические ресурсы, используемые в интерфейсах приложения. Например, изображения и кадры анимации противника boss_frame1, boss_frame2. Также были использованы изображения шкалы здоровья противника ic_healthbar10, ic_healthbar8, и т.д. Кроме того, в данной папке хранятся иконки для навигационной панели ic_tasks.xml, ic_profile.xml и др. Папку drawable со всеми файлами можно увидеть на рисунке 17.

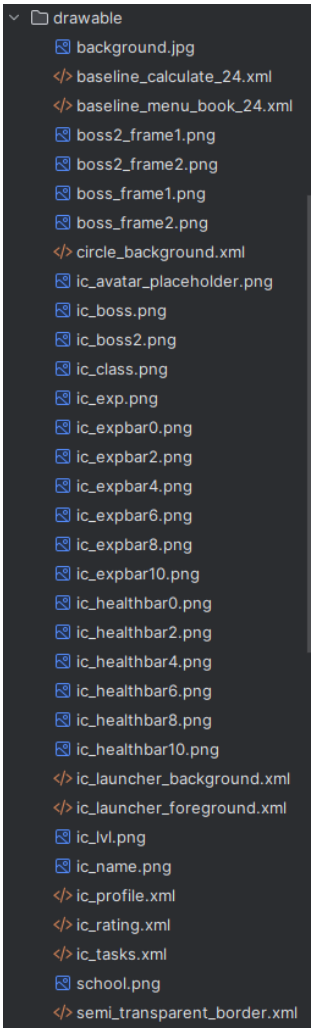


Рисунок 17 – Папка drawable с графическими ресурсами

В результате получилось разработать визуальное оформление интерфейса, улучшающее восприятие приложения пользователем.

3.2 Тестирование мобильного обучающего приложения

Планируется проведение тестирования функциональности мобильного обучающего приложения. Особое внимание будет уделено работе с данными пользователей, что имеет ключевое значение для функционирования приложения. Точность обработки информации о выполненных заданиях и начисленных баллах напрямую влияет на удобство использования системы. В частности, при выполнении пользователем задания система должна корректно обрабатывать введенные данные (ответы, баллы, время выполнения), сохранять их в локальной базе данных SQLite в правильном формате и оперативно обновлять.

Необходимо проверить корректность всех полей ввода для обеспечения их обязательного заполнения. При введении некорректных данных, например, неверных ответов на вопросы, приложение должно выдавать понятные сообщения об ошибке. Важно убедиться, что данные о результатах заданий корректно передаются и отображаются в профиле пользователя.

Необходимо протестировать навигацию между разделами приложения, чтобы обеспечить плавные и стабильные переходы между экранами. Пользователь должен иметь возможность легко перемещаться между четырьмя основными разделами: профиль, таблица лидеров, игровой модуль и помощь.

Следует также проверить функциональность игрового модуля и системы начисления баллов. Приложение должно корректно генерировать случайные вопросы или задания, начислять баллы за правильные ответы и автоматически обновлять данные в базе данных и таблице лидеров. При достижении пользователем определенного количества баллов приложение должно корректно отображать его прогресс и обновлять позицию в рейтинге.

Результаты тестирования приложения можно увидеть в таблице 4.

Таблица 4 – Результаты тестирования приложения

Действие	Ожидаемый результат	Результат
Выполнение задания в игровом модуле	Начисление баллов за выполненное задание, сохранение результатов в БД, обновление профиля пользователя, обновление таблицы лидеров	Соответствует
Аутентификация пользователя	Корректная проверка логина и пароля, переход в профиль пользователя, сохранение текущей сессии пользователя	Соответствует
Отображение и обновления профиля пользователя	Корректное отображение персональных данных пользователя, корректное обновление баллов и уровня пользователя	Соответствует
Обновление таблицы лидеров	Корректное отображение места в рейтинге, обновление в реальном времени	Соответствует
Навигация между разделами	Плавные переходы между экранами и сохранение состояния при возврате	Соответствует

На основе результатов тестирования приложения можно сделать вывод, что все реализованные функции работают корректно. Это свидетельствует о том, что приложение выполняет все предусмотренные задачи.

На рисунках 18 – 24 продемонстрировано приложение на реальном устройстве.

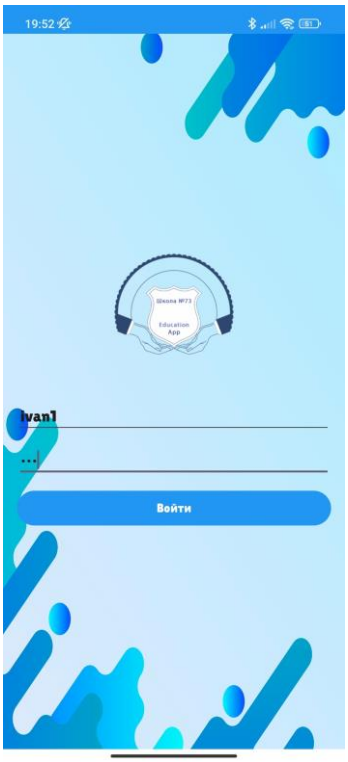


Рисунок 18 – Экран авторизации

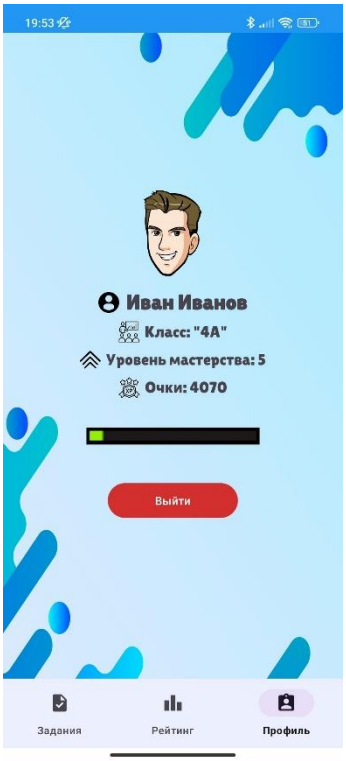


Рисунок 19 – Экран профиля

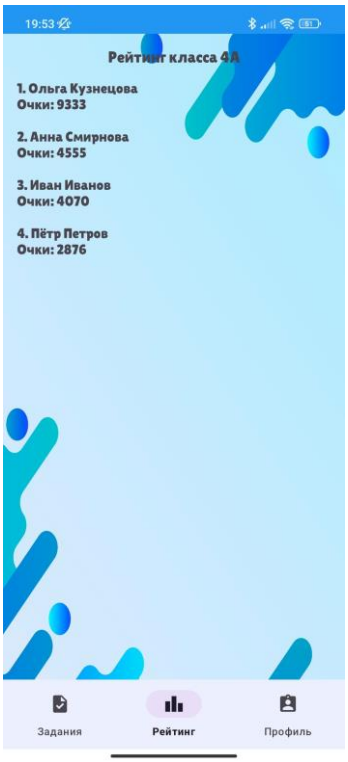


Рисунок 20 – Экран рейтинга

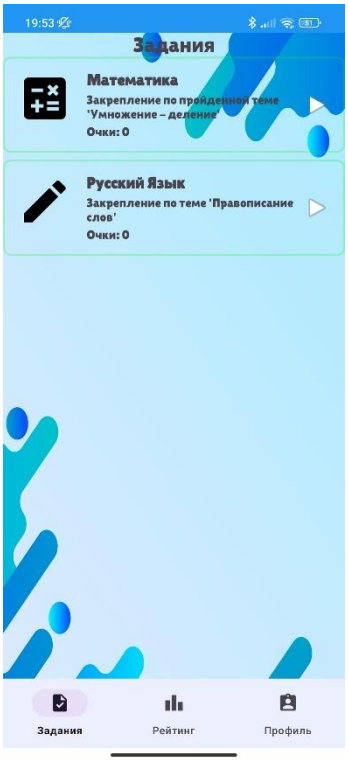


Рисунок 21 – Экран заданий

Рисунок 22 – Экран игры «Математика»

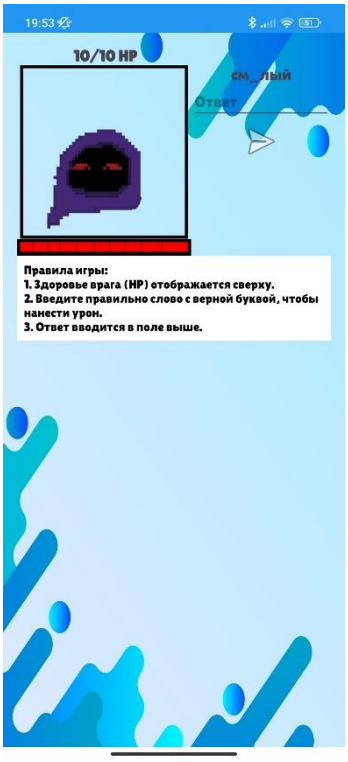


Рисунок 23 – Экран игры «Русский язык»

Рисунок 24 – Экран результата игры

После завершения разработки мобильного обучающего приложения был создан APK-файл для тестирования его работоспособности на реальных устройствах. APK представляет собой формат исполняемых файлов для операционной системы Android, содержащий весь программный код приложения и его ресурсы.

Вывод по главе 3

В результате работы были реализованы семь основных классов и вспомогательный класс DatabaseHelper. Созданы XML-файлы макетов

интерфейса, обеспечивающие интуитивно понятное отображение экранов, включая авторизацию, профиль, рейтинг, выбор заданий и игровой модуль. Графические ресурсы, хранящиеся в папке drawable, обеспечили визуальное оформление, включая анимации и шкалы прогресса. Класс DatabaseHelper реализует создание и обновление базы данных с таблицей Users для хранения пользовательских данных. Функциональность приложения включает авторизацию, отображение профиля с данными и статистикой, формирование рейтинга на основе баллов, а также игровой модуль с генерацией заданий и начислением очков. Тестирование подтвердило корректность работы всех функций: обработка данных заданий; обновление профиля и рейтинга в реальном времени, стабильная навигация. Созданный APK-файл позволил успешно проверить приложение на реальных устройствах, подтвердив его соответствие заданным требованиям.

Заключение

В ходе выполнения бакалаврской работы было создано мобильное обучающее приложение, интегрирующее игровые механики и соревновательные элементы для повышения эффективности обучения. Целью разработки стало обеспечение пользователей удобным инструментом для выполнения образовательных заданий, отслеживания прогресса и участия в рейтинговой системе.

На подготовительном этапе были проанализированы особенности образовательных приложений и определены функциональные требования. Для проектирования структуры и логики работы приложения использовались CASE-средства, позволившие создать UML-диаграммы.

Разработка приложения включала выбор технологий, проектирование архитектуры и создание интерфейса. В качестве среды разработки применялась Android Studio с использованием языка Kotlin.

Реализация функционала потребовала решения технических задач, таких как создание локальной базы данных SQLite для хранения информации о пользователях, результатах заданий и баллах. Были разработаны модули авторизации, профиля пользователя, рейтинговой системы и игрового режима с генерацией заданий и учетом баллов.

Тестирование подтвердило корректность функционирования всех компонентов. Проверялись обработка данных заданий, корректность ввода, обновление профиля и рейтинга, а также плавность переходов между разделами. Результаты показали, что приложение полностью соответствует поставленным задачам и готово к использованию на реальных устройствах.

Созданное приложение является полноценным решением для интерактивного обучения с использованием игровых и соревновательных элементов. В дальнейшем возможно расширение функциональности, например, добавление новых типов заданий. На текущем этапе приложение отвечает всем требованиям и может эффективно использоваться для обучения.

Список используемой литературы и используемых источников

1. Дубейковский В. И. Практика функционального моделирования с AllFusion Process Modeler / В. И. Дубейковский. – Москва: ДИАЛОГ-МИФИ,2021. – 464 с.
2. Новиков Б. А., Горшкова Е. А., Графеева Н. Г. Основы технологий баз данных: учебное пособие / Б. А. Новиков, Е. А. Горшкова, Н. Г. Графеева; под ред. Е. В. Рогова. – 2-е изд. – Москва: ДМК Пресс, 2020. – 582 с.
3. Нотация IDEF0: как устроена и для чего нужна. [Электронный ресурс]. URL: https://avtograf.tech/blog/notatsiya-idef0-kak-ustroena-i-dlya-chego-nuzhna/ (дата обращения 21.03.2025).
4. Мобильные ОС в России. [Электронный ресурс] / Режим доступа: URL: https://radar.yandex.ru/mobile (дата обращения 01.05.2024).
5. Обзор iOS [Электронный ресурс]. URL: https://blog.skillfactory.ru/ glossary/ios/ (дата обращения: 05.12.2024).
6. Google play магазин приложений [Электронный ресурс]. URL: https://ru.m.wikipedia.org/wiki/Google_Play/ (дата обращения: 15.04.2025).
7. Средства автоматизации разработки программ [Электронный ресурс]. URL: https://ru.m.wikipedia.org /wiki/CASE (дата обращения: 07.04.2025).
8. Android Studio интегрированная среда разработки [Электронный ресурс]. URL:https://en.wikipedia.org/wiki/Android_Studio (дата обращения: 12.04.2025).
9. Xamarin платформа для разработки C# [Электронный ресурс]. URL: https://dotnet.microsoft.com/en-us/apps/xamarin (дата обращения: 10.04.2025).
10. Kotlin объектно-ориентированный язык [Электронный ресурс]. URL: https://ru.wikipedia.org/wiki/Kotlin (дата обращения: 20.04.2025).
11. Diagrams.Net. [Электронный ресурс] / Режим доступа: URL: https://en.wikipedia.org/wiki/Diagrams.net (дата обращения 01.05.2025).

12. Гарнаев, А. Web-программирование на Java и JavaScript / А. Гарнаев. - СПб.:BHV, 2005. 1040 с.

13. Ананьев И.В., Серова Е.Г. Области эффективного применения нотации IDEF0 для задач описания бизнес-процессов // Вестник СанктПетербургского университета. Менеджмент. 2008. №2. URL: https://cyberleninka.ru/article/n/oblasti-effektivnogo-primeneniya-notatsii-idef0-dlya-zadach-opisaniya-biznes-protsessov-1 (дата обращения: 03.03.2025).

14. Антал М.А. Особенности планирования в современном производстве / М.А. Антал // В сборнике: Приоритетные направления развития экономики и менеджмента: теоретические и практические аспекты. Сборник научных статей. Уфа, 2021. С. 98-101.

15. Фрайман, З. Создание приложений для смартфонов и планшетов под ОС Android. Практический курс / З. Фрайман. — М: Едиториал УРСС, 2020. — 504 с.

16. Сьюзан, Давид Келлер UML 2 для начинающих. 3-е изд. Библиотека программиста / Сьюзан Келлер, Давид Келлер. – М.: ДМК Пресс, 2010. – 382 с

17. Develop for Android [Электронный ресурс]. URL: https://developer.android.com/develop (дата обращения 16.03.2025)

18. Herv Guihot. Pro Android Apps Performance Optimization / H. Guihot. – Apress, 2022. - 194 с.

19. Camilus Raynaldo, Android UI Design with XML / C. Raynaldo. – Webucator, 2017. - 232 с.

20. Guide to app architecture. [Электронный ресурс]. URL: https://developer.android.com/topic/architecture (дата обращения 11.02.2025).

21. SQLite Documentation [Электронный ресурс]. – URL: https://www.sqlite.org/docs.html (дата обращения: 12.05.2025).

22. The Collaborative Interface Design Tool [Электронный ресурс]. URL: https://www.figma.com (дата обращения 21.04.2025)