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 «Интеграция средств дистанционного обучения в мобильное приложение»

Обучающийся Д.С. Тетюхин
(Инициалы Фамилия) (личная подпись)

Руководитель М.А. Тренина
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Консультант к.п.н., доцент С.А. Гудкова
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Тольятти 2025

Аннотация

Выпускная квалификационная работа на тему «Разработка мобильного приложения для интеграции средств дистанционного обучения», выполнена студентом группы МОб-2102б Тетюхиным Д. С.

Выпускная квалификационная работа посвящена вопросам автоматизации процесса дистанционного обучения за счет разработки мобильного приложения, обеспечивающего обмен учебными материалами между преподавателями и студентами, а также систему обратной связи для проверки заданий.

Структура работы представлена введением, тремя главами, заключением и списком литературы.

Во введении определены актуальность темы, цели и задачи работы, объект и предмет исследования.

В первой главе выполнен анализ существующих платформ для дистанционного обучения, рассмотрены их преимущества и недостатки, а также обоснован выбор среды разработки Android Studio. На основе проведенного анализа сформулированы требования к разрабатываемому мобильному приложению.

Во второй главе осуществлено проектирование мобильного приложения: разработана архитектура и основные функции.

В третьей главе отображено функциональное тестирование приложения.

В заключении представлены выводы о проделанной работе и перспективы дальнейшего развития приложения.

Результатом ВКР является работоспособное мобильное приложение, разработанное в Android Studio, которое готово к использованию в учебном процессе.

В работе представлены 26 рисунков, список использованной литературы содержит 20 источников. Общий объем выпускной квалификационной работы составляет 48 страниц.

Abstract

Graduation qualification work on the topic “Development of mobile application for integration of distance learning tools”, performed by the student of group MOb-2102b Tetyukhin D. S.

The graduate qualification work is devoted to the automation of the process of distance learning through the development of a mobile application that provides a convenient exchange of educational materials between teachers and students, as well as a feedback system for checking assignments.

The structure of the work is presented by the introduction, three chapters, conclusion and list of references.

The introduction defines the relevance of the topic, goals and objectives of the work, object and subject of the study.

The first chapter analyzes the existing platforms for distance learning (Google Classroom, Moodle, etc.), considers their advantages and disadvantages, and justifies the choice of development environment Android Studio. Based on the analysis, the requirements for the mobile application to be developed are formulated.

In the second chapter the design of the mobile application is realized: the architecture, user scenarios, interface and main functions (uploading/downloading files in Word format, teacher's comment system) are developed.

The third chapter justifies the choice of implementation tools (Kotlin/Java) provides a description of the key modules of the application.

The work contains 26 figures, the list of used literature contains 25 sources. The total volume of the graduate qualification work is 48 pages.

Оглавление

Введение 5
Глава 1 Постановка для исследования 8
1.1 Краткая характеристика предметной области и выявление задач ... 8
1.2 Определение технологий разработки 9
1.3 Пользовательские сценарии работы 12
Глава 2 Выбор архитектуры и разработка приложения 17
2.1 Архитектура приложения 17
2.2 Разработанное решение для приложения 19
2.3 Разработка модуля авторизации 21
2.4 Разработка основного функционала 25
Глава 3 Тестирование работоспособности приложения 32
Заключение 45
Список используемой литературы и используемых источников 46

Введение

Мобильное приложение представляет собой специализированное программное обеспечение, разработанное для работы на смартфонах, планшетах и других портативных устройствах. Сегодня такие приложения охватывают широкий спектр функций – от коммуникации и развлечений до профессиональной и образовательной деятельности. Изначально образовательные мобильные приложения выполняли ограниченную роль, предлагая пользователям лишь электронные справочники или учебники. Однако с развитием дистанционного обучения возникла необходимость в более сложных интерактивных платформах, позволяющих преподавателям и студентам обмениваться материалами, заданиями и обратной связью в режиме реального времени.

Дистанционное обучение – это форма образовательного процесса, исключающая необходимость физического присутствия в аудитории за счет использования цифровых технологий. Несмотря на его растущую популярность, одной из ключевых проблем остается отсутствие удобных мобильных решений для оперативного обмена файлами с возможностью их редактирования и комментирования.

Актуальность данного исследования обусловлена несколькими факторами. Во-первых, после пандемии COVID-19 значительно возрос спрос на мобильные образовательные приложения. Во-вторых, существующие решения зачастую либо слишком сложны (например, полноценные LMS-платформы вроде Moodle), либо не предоставляют необходимых функций для удобного обмена файлами и обратной связи. В-третьих, в условиях гибридного обучения требуется оптимизация процесса проверки заданий, что делает разработку специализированного мобильного приложения особенно востребованной.

Научная новизна проекта заключается в создании мобильного приложения, которое упрощает взаимодействие между преподавателями и студентами. Приложение позволяет загружать задания в формате Word, редактировать их непосредственно в мобильной среде, а также обеспечивает систему комментариев для оперативной обратной связи. В отличие от сложных LMS-систем, предлагаемое решение ориентировано на простоту использования и доступность, что делает его применимым не только в вузах, но и в школах, а также на курсах дополнительного образования.

Основной целью выпускной квалификационной работы является разработка мобильного приложения, предназначенного для дистанционного обучения и обеспечивающего эффективный обмен учебными материалами между преподавателями и студентами. Для достижения этой цели были поставлены задачи, включающие анализ существующих образовательных платформ, исследование процессов взаимодействия в системе «преподаватель – студент», проектирование архитектуры приложения, выбор оптимальных технологий разработки, создание пользовательского интерфейса и тестирование работоспособности системы.

Объектом исследования выступает процесс цифрового взаимодействия между преподавателями и обучающимися, осуществляемый через мобильное приложение. Предметом исследования являются современные технологии разработки подобных образовательных решений, включая их архитектурные особенности, функциональные возможности и методы интеграции в систему дистанционного обучения.

Работа состоит из введения, трех глав, заключения, списка использованных источников и приложения. В первой главе проводится анализ современных систем дистанционного обучения, рассматриваются их преимущества и недостатки, а также формируются требования к разрабатываемому приложению. Вторая глава посвящена проектированию системы, включая определение ключевых функций, выбор технологий хранения данных и разработку интерфейса. Третья глава описывает процесс

реализации приложения с использованием Android Studio и SQLite, а также содержит результаты тестирования и оценку эффективности решения.

В заключении представлены ключевые выводы, подтверждающие успешное выполнение поставленных задач. Разработанное мобильное приложение эффективно оптимизирует обмен учебными материалами, предоставляет удобные инструменты для обратной связи и способствует повышению эффективности дистанционного обучения. Его модульная архитектура позволяет легко адаптировать решение под конкретные образовательные потребности. Решение обладает значительным потенциалом для внедрения в образовательные учреждения, так как не только упрощает взаимодействие между преподавателями и студентами, но и соответствует современным требованиям.

Глава 1 Постановка для исследования

1.1 Краткая характеристика предметной области и выявление задач

«Современная образовательная среда активно трансформируется под влиянием цифровых технологий, что особенно ярко проявляется в росте спроса на мобильные решения для обучения» [3]. Разрабатываемое приложение представляет собой специализированную платформу, предназначенную для организации тестирования в учебных заведениях. Его ключевая цель – создать эффективный инструмент взаимодействия между преподавателями и студентами, обеспечивающий автоматизацию процессов создания, выполнения и проверки тестов.

Суть приложения заключается в объединении двух основных функций: для преподавателей – удобное создание и распределение тестовых заданий, а для студентов – возможность их прохождения с обратной связью. Такое решение устраняет необходимость использования бумажных носителей и ручной проверки, значительно сокращая временные затраты преподавателей и повышая объективность оценки знаний.

Одной из главных задач разработки является обеспечение простого и интуитивно понятного интерфейса, адаптированного под разные роли пользователей. Преподаватели должны иметь возможность легко составлять тесты, выбирать вопросы, настраивать критерии оценки и назначать задания конкретным учебным группам. Студенты, в свою очередь, получают доступ только к тем тестам, которые предназначены для их группы, с последующим отображением результатов.

Еще одной важной задачей является реализация надежной системы хранения и обработки данных. Приложение должно обеспечивать безопасное хранение тестов, результатов их прохождения и персональной информации пользователей. Для этого требуется интеграция с базой данных, способной

оперативно обрабатывать запросы и предоставлять актуальную статистику по выполненным работам.

Особое внимание уделяется функционалу обратной связи. После прохождения теста студент сразу видит, какие вопросы были решены правильно, а какие – нет, что способствует более осознанному усвоению материала. Преподаватель, в свою очередь, получает сводную аналитику по каждой группе, что позволяет оперативно выявлять проблемные темы и корректировать учебный процесс.

Гибкость системы проявляется в поддержке различных форматов тестовых заданий: от стандартных вопросов с выбором ответа до открытых заданий, требующих развернутого решения. Это расширяет возможности использования приложения в разных дисциплинах и образовательных программах.

«Для обеспечения стабильной работы приложения необходима реализация модульной архитектуры, которая позволит масштабировать функционал в будущем» [9]. Важно предусмотреть возможность добавления новых типов вопросов, интеграции с другими образовательными платформами и адаптации под изменяющиеся требования пользователей.

Таким образом, разрабатываемое решение не только отвечает актуальным запросам современного образования, но и создает основу для дальнейшего развития цифровых инструментов обучения. Оно сочетает в себе простоту использования, технологическую надежность и эффективность, что делает его ценным инструментом для преподавателей и студентов.

1.2 Определение технологий разработки

В сфере создания программных продуктов для образования сегодня применяются передовые методики и инструменты, позволяющие разрабатывать эффективные решения, отвечающие потребностям учебного процесса. Особое значение приобретает мобильная разработка, предлагающая

различные технологические подходы к созданию образовательных приложений. Каждый из этих подходов имеет свои особенности, преимущества и ограничения, которые необходимо учитывать при выборе оптимального решения для конкретных образовательных задач.

«Кроссплатформенная разработка представляет особый интерес для образовательных учреждений, так как позволяет создавать приложения, работающие на разных операционных системах – Android, iOS и веб-платформах» [22]. Основное достоинство этого подхода заключается в возможности использования единой кодовой базы для всех платформ, что существенно сокращает временные и финансовые затраты на разработку. «Однако применительно к образовательным приложениям важно учитывать, что кроссплатформенные решения могут демонстрировать меньшую производительность по сравнению с нативными приложениями, а также иметь ограниченный доступ к некоторым функциям мобильных устройств» [10]. Это может негативно сказаться на качестве пользовательского опыта, что критично в образовательной среде.

Веб-приложения для образования обладают значительным потенциалом благодаря своей универсальности и доступности. Они не требуют установки на устройство и могут использоваться через браузер на любом современном гаджете. Это особенно ценно для образовательных учреждений, стремящихся обеспечить равный доступ к учебным материалам для всех учащихся. Однако при разработке образовательных веб-приложений следует учитывать их зависимость от интернет-соединения и ограниченные возможности интеграции с аппаратными функциями устройств, такими как камера или геолокация. Эти ограничения могут стать препятствием для реализации интерактивных и мультимедийных компонентов, что существенно влияет на эффективность учебного процесса.

Нативная разработка образовательных приложений, хотя и требует больших ресурсов, предлагает неоспоримые преимущества с точки зрения качества и функциональности. Создавая отдельные версии приложения для

каждой платформы с использованием специализированных языков программирования, разработчики могут обеспечить максимальную производительность и полную интеграцию с возможностями устройства. Для образовательных приложений это означает плавную работу даже с ресурсоемкими учебными материалами, такими как интерактивные 3D-модели или видео в высоком разрешении. Важным аспектом нативной разработки для образовательных целей является возможность создания интерфейса, который полностью соответствует ожиданиям пользователей каждой платформы. Это способствует более быстрому освоению приложения как преподавателями, так и учащимися, снижая когнитивную нагрузку и позволяя сосредоточиться непосредственно на учебном процессе. Кроме того, нативные приложения обеспечивают доступ ко всем образовательным возможностям устройства, включая работу с мультимедийными материалами и использование дополнительных аппаратных функций. При разработке образовательных мобильных решений особое внимание следует уделять вопросам стабильности и безопасности. Нативный подход в этом отношении предоставляет значительные преимущества, так как позволяет минимизировать зависимость от сторонних библиотек и снижает риск возникновения проблем совместимости. Это особенно важно для образовательных учреждений, где надежность и безопасность работы с персональными данными учащихся являются приоритетными задачами.

«Для реализации мобильного приложения дистанционного обучения был выбран нативный подход с использованием следующих технологий: среда разработки – Android Studio, платформа – Android, язык программирования – Kotlin, интерфейс – Compose, база данных – Room, архитектура – MVVM. Выбор языка программирования Kotlin обусловлен его официальной поддержкой для Android-разработки и современными языковыми конструкциями, которые значительно упрощают создание надежного и поддерживаемого кода. Kotlin обладает полной совместимостью с Java, что позволяет использовать все богатство существующих Android-библиотек и

фреймворков» [2]. Особенно ценными для образовательного приложения оказались такие особенности Kotlin, как null-безопасность, что минимизирует количество критических ошибок в работе приложения, и корутины, обеспечивающие плавную работу с сетевыми запросами и операциями с базой данных. «Для хранения локальных данных приложения была выбрана легковесная реляционная база данных Room, которая является частью Android Jetpack и предоставляет абстракцию над SQLite» [1]. Это решение идеально подходит для мобильных образовательных приложений, обеспечивая надежное хранение учебных материалов, результатов тестирования и пользовательских настроек даже при отсутствии интернет-соединения. Важным преимуществом стало полная поддержка Room в Android SDK, что упростило интеграцию базы данных в приложение. Использование Compose для создания интерфейсов обеспечивает гибкость и возможность быстрой адаптации пользовательского опыта, что особенно важно для образовательных приложений, ориентированных на разные категории пользователей.

Таким образом, выбор нативной разработки с использованием современных технологий и подходов позволяет создать высококачественное образовательное приложение, отвечающее потребностям пользователей и обеспечивающее эффективный учебный процесс.

1.3 Пользовательские сценарии работы

Пользовательский интерфейс представляет собой критически важный компонент программного обеспечения, определяющий эффективность взаимодействия человека с системой. Грамотно реализованный интерфейс существенно повышает уровень пользовательского опыта, способствуя не только удовлетворенности от работы с приложением, но и глубине вовлеченности в процесс его использования. Напротив, несоблюдение принципов эргономичного проектирования и игнорирование устоявшихся

стандартов использования неизбежно приводит к возникновению значительных трудностей в процессе эксплуатации программного продукта.

«При проектировании интерфейсных решений для мобильных приложений необходимо руководствоваться тремя фундаментальными принципами» [16]. Первый касается когнитивной доступности – интерфейс должен обладать логичной структурой, где все визуальные элементы выполнены в единой стилистике, а интерактивные компоненты имеют четкое визуальное выделение и эргономичное расположение. «Второй принцип относится к отзывчивости системы – интерфейс должен обеспечивать мгновенный и предсказуемый отклик на действия пользователя с минимальными задержками и плавными анимационными переходами» [17]. Третий принцип затрагивает эффективность навигации – организация меню и переходов должна позволять пользователю оперативно получать доступ к ключевым функциональным возможностям, расположенным не далее, чем в двух шагах от основного экрана.

На рисунке 1 приведена диаграмма вариантов использования.

Рисунок 1 – Диаграмма вариантов использования

«Диаграмма вариантов использования обеспечивает концептуальное представление системы, фокусируясь на её внешних функциональных возможностях и взаимодействиях с пользователями» [19]. Этот подход позволяет получить целостное понимание логики работы приложения с точки зрения конечного пользователя, не углубляясь в технические детали реализации. В свою очередь, диаграмма взаимодействия модулей раскрывает внутреннюю организацию системы, детализируя её структурные компоненты, распределение функциональных обязанностей между ними и существующие взаимосвязи. «Такой анализ особенно важен для оценки архитектурной сложности системы, выявления потенциальных точек отказа и понимания зависимостей между компонентами, что имеет критическое значение как на этапе разработки, так и при последующем сопровождении программного обеспечения» [15]. Диаграмма взаимодействия модулей приведен на рисунке 2.

«Ключевое различие между этими типами диаграмм заключается в уровне абстракции: если первая описывает "что делает система" для пользователя, то вторая показывает "как она это делает" на уровне технической реализации. Оба представления дополняют друг друга, обеспечивая комплексное понимание системы с разных перспектив» [3].

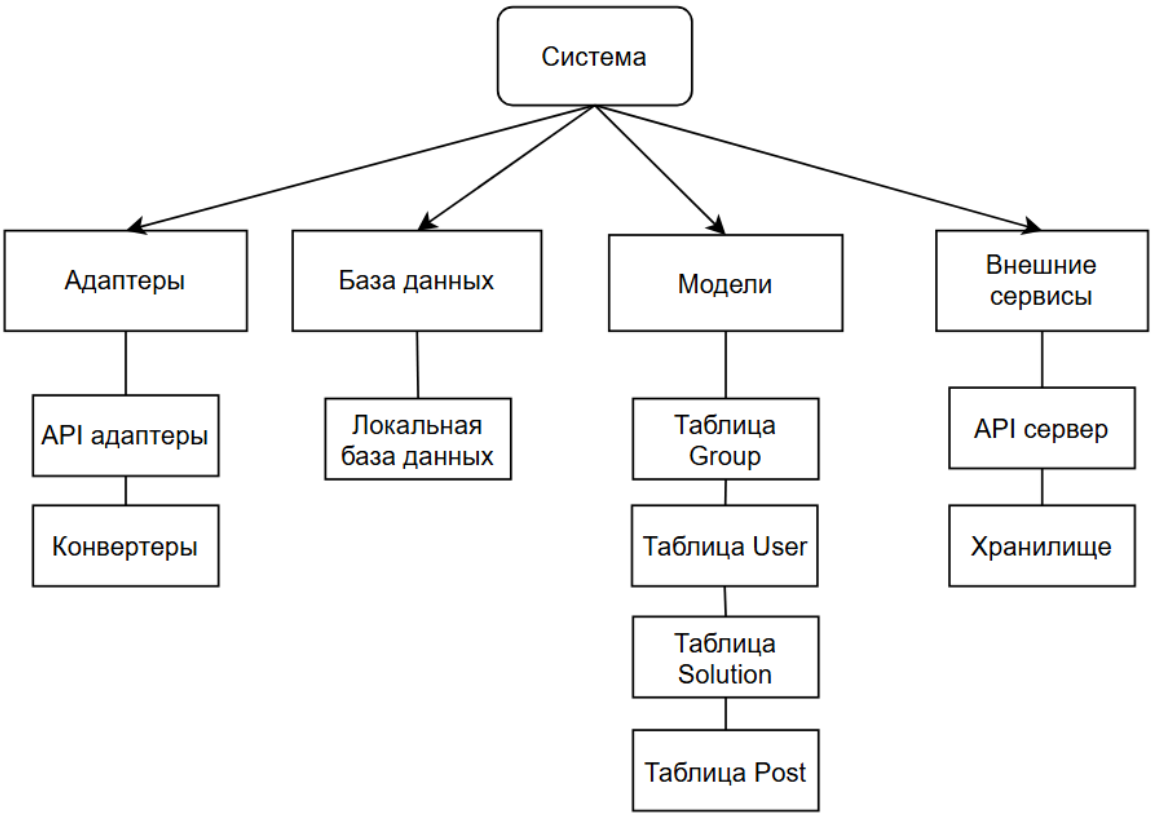

Рисунок 2 – Диаграмма взаимодействия модулей ПО

Модуль адаптеров выполняет функции преобразования данных между различными слоями приложения. «Инкапсулирует логику адаптации данных из моделей для UI-слоя, а также обеспечивает взаимодействие с внешними сервисами и локальным хранилищем» [21]. Способствует снижению связанности компонентов системы.

База данных обеспечивает персистентное хранение информации, используя локальную базу данных. Реализует операции CRUD, кэширование данных и синхронизацию с удаленными источниками. Предоставляет абстракцию над механизмами хранения, изолируя остальные компоненты от специфики работы с СУБД. «Модель содержит ядро приложения, включая доменные сущности и бизнес-правила» [13]. «Определяет структуры данных и основные алгоритмы обработки информации» [5]. Обеспечивает

независимость ключевых бизнес-процессов от конкретной реализации UI или механизмов хранения.

Предоставляет набор вспомогательных функций и утилит общего назначения, используемых различными компонентами системы. Включает инструменты для валидации, преобразования форматов, работы с датами и другие инфраструктурные сервисы. Способствует устранению дублирования кода и соблюдению принципа DRY.

Вывод по первой главе.

В первой главе мы проанализировали предметную область и на основе этого анализа разрабатывается мобильное приложение, которое упростит обмен учебными материалами между преподавателями и учениками, сделав процесс обучения более удобным и эффективным.

Глава 2 Выбор архитектуры и разработка приложения

2.1 Архитектура приложения

«Выбор архитектуры приложения является одним из ключевых этапов в процессе разработки программного обеспечения, поскольку он определяет не только структуру кода, но и влияет на все последующие стадии жизненного цикла продукта» [20]. Правильно подобранная архитектура создает прочную основу для разработки масштабируемого, поддерживаемого и надежного решения, способного адаптироваться к изменяющимся требованиям пользователей и бизнеса.

В современной практике разработки мобильных приложений особое внимание уделяется проверенным временем архитектурным паттернам, которые отражают накопленный опыт сообщества разработчиков и предлагают оптимальные способы организации кодовой базы. Их применение способствует четкому разделению ответственности между компонентами системы, что в итоге ведет к созданию более качественного и устойчивого программного продукта.

«В частности, в разработке Android-приложений широко используется паттерн MVVM (Model-View-ViewModel), который представляет собой современный и гибкий подход к организации кода» [18]. В основе MVVM лежит концепция двустороннего связывания данных, благодаря чему ViewModel выступает посредником между моделью и представлением, автоматически синхронизируя состояние пользовательского интерфейса с данными бизнес-логики. Это существенно упрощает управление состоянием приложения и повышает его отзывчивость, что особенно важно в образовательных решениях с динамическим контентом и интерактивными элементами. Иллюстрация работы MVVM приведена на рисунке 3.

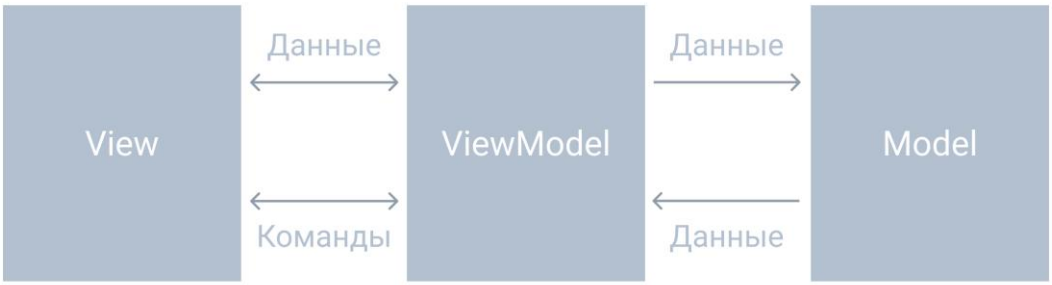

Рисунок 3 – Иллюстрация архитектуры MVVM

Компонент Model в MVVM отвечает за работу с данными и бизнес-логику, инкапсулируя все правила и алгоритмы приложения, а также обеспечивая взаимодействие с различными источниками информации. При этом Model остается полностью независимой от пользовательского интерфейса, что позволяет легко модифицировать визуальную часть без затрагивания бизнес-логики. View же отвечает за отображение данных и взаимодействие с пользователем, при этом не содержит бизнес-логики, что облегчает тестирование и поддержку интерфейса.

«ViewModel играет ключевую роль, связывая Model и View, управляя состоянием интерфейса и обрабатывая пользовательские действия. Благодаря этому слой ViewModel становится центром бизнес-логики представления, что позволяет добиться высокой степени разделения ответственности и улучшает читаемость и поддержку кода. Такой подход способствует более эффективному тестированию, поскольку логику можно проверять отдельно от визуальной части» [3].

Одним из значимых преимуществ MVVM является его тесная интеграция с современными фреймворками Android, такими как Jetpack Compose и LiveData, что обеспечивает удобные инструменты для создания

реактивных и отзывчивых пользовательских интерфейсов. Это особенно важно в образовательных приложениях, где требуется быстрое обновление данных и гибкая адаптация интерфейса под потребности различных категорий пользователей. Кроме того, архитектура MVVM способствует параллельной работе специалистов разных профилей – разработчики могут сосредоточиться на реализации бизнес-логики и данных в ViewModel и Model, в то время как дизайнеры и фронтенд-разработчики работают над визуальной частью View. Такое разделение труда ускоряет процесс разработки и повышает качество конечного продукта. MVVM также обеспечивает высокую гибкость при внесении изменений: модификация одного компонента минимально влияет на остальные, что облегчает поддержку и развитие приложения. Это критично для образовательных систем, которые часто требуют обновления функционала и интерфейса в соответствии с изменяющимися методическими требованиями и пользовательскими ожиданиями. В контексте образовательных приложений MVVM демонстрирует особую эффективность, обеспечивая логичное и прозрачное разделение на три ключевых компонента. Model управляет учебными материалами, обработкой контента и аналитикой успеваемости, ViewModel координирует процесс обучения и взаимодействие с пользователем, а View предоставляет интуитивно понятный и адаптивный интерфейс для студентов и преподавателей. Такой подход способствует созданию надежного и удобного программного продукта, способствующего эффективному учебному процессу.

2.2 Разработанное решение для приложения

Разработанное мобильное приложение представляет собой комплексное решение для организации образовательного процесса, объединяющее функциональность обмена учебными материалами, проверки заданий и управления учебными группами. Система предоставляет участникам

образовательного процесса удобный инструмент для взаимодействия и координации учебной деятельности.

Пользовательский интерфейс реализован с использованием современных подходов к проектированию Android-приложений, обеспечивая адаптивность и соответствие актуальным стандартам материального дизайна. Процесс разработки осуществлялся в профессиональной среде Android Studio, что гарантировало соблюдение лучших практик создания мобильных приложений.

Ключевые функциональные возможности приложения включают механизмы аутентификации пользователей, управления учебными группами, обмена учебными материалами, проверки выполненных тестов и предоставления обратной связи. Особое внимание уделено реализации интуитивно понятного интерфейса с продуманной навигационной структурой.

Блок схема работы приложения приведен на рисунке 4.

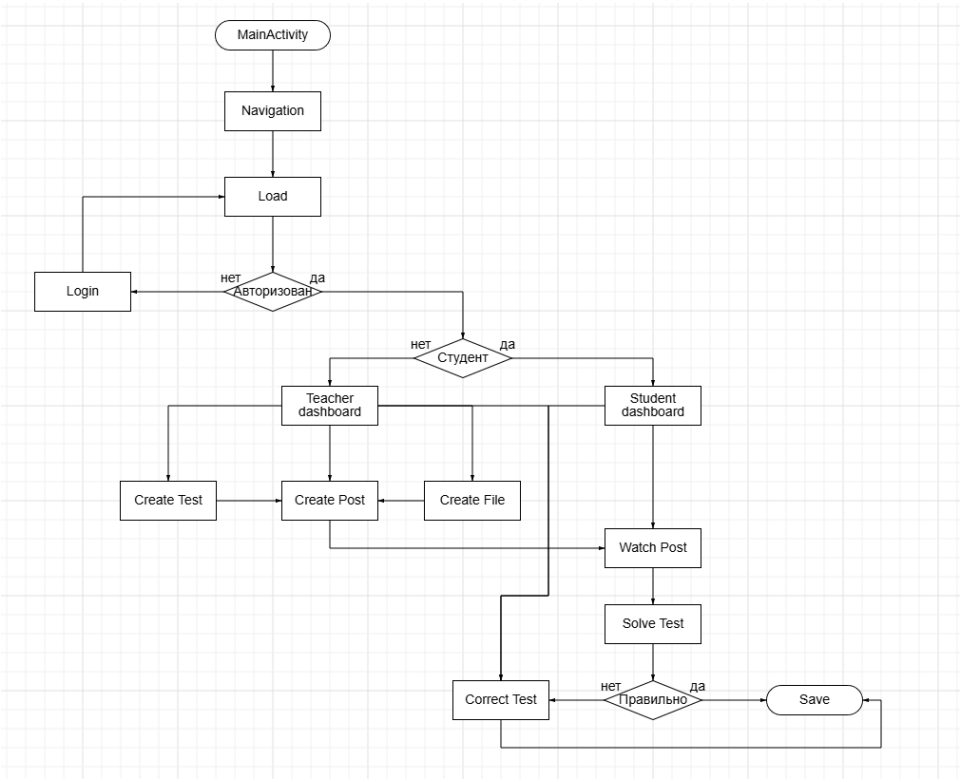

Рисунок 4 – Блок схема работы приложения

«Архитектурное решение на основе паттерна MVVM в сочетании с продуманной моделью данных обеспечивает высокую производительность системы и возможность ее дальнейшего расширения» [8]. Приложение демонстрирует соответствие современным требованиям к пользовательскому опыту, сочетая простоту взаимодействия с богатым функционалом, необходимым для организации эффективного образовательного процесса.

2.3 Разработка модуля авторизации

Разработка приложения начинается с создания нового проекта в среде Android Studio. В качестве шаблона выбирается "Empty Views Activity" (приведен на рисунке 5). «Это обеспечивает минимальную стартовую конфигурацию. После указания основных параметров проекта, таких как имя пакета, минимальная версия SDK и другие метаданные, система автоматически генерирует базовую структуру файлов и каталогов» [18]. Заполнение данный проекта приведен на рисунке 6.

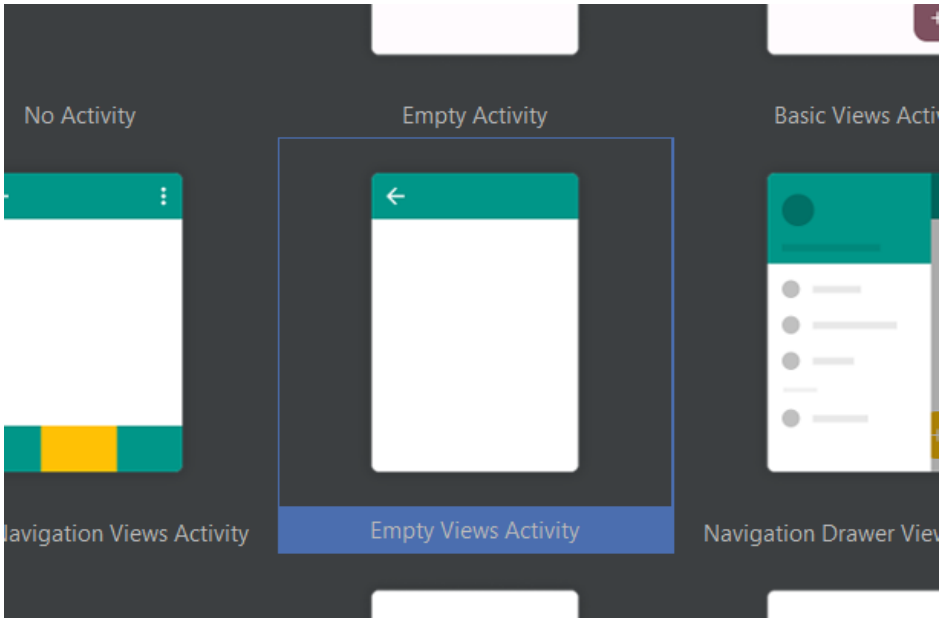

Рисунок 5 – Выбор начального экрана

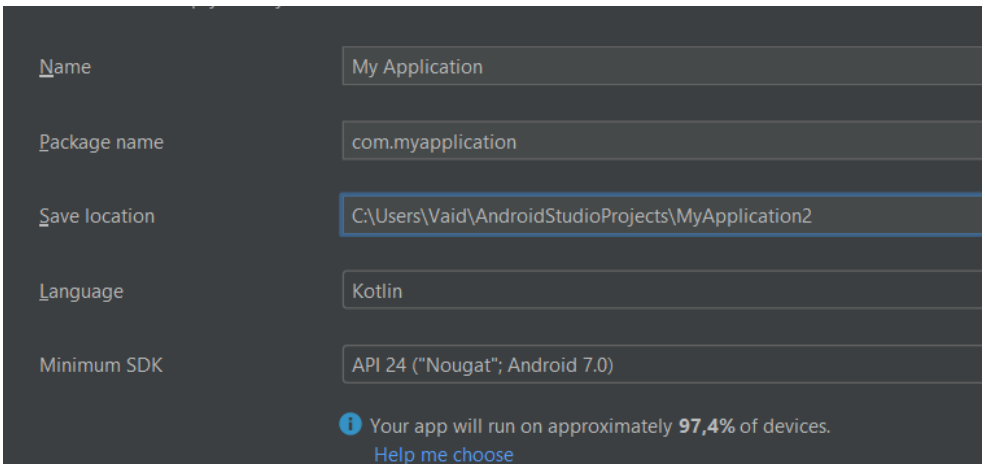

Рисунок 6 – Заполнение данных

«Конфигурация системы сборки осуществляется через файл “build.gradle.kts”» [23]. Файл представляет собой декларативное описание всех аспектов сборки проекта на языке Kotlin (рисунок 7).

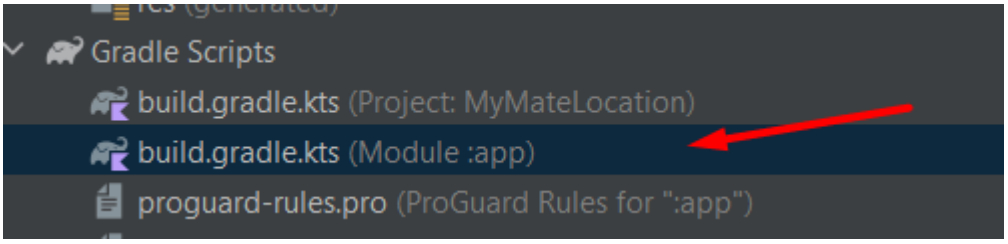

Рисунок 7 – Файл Gradle

Этот файл выполняет несколько важных функций: определяет внешние зависимости, настраивает параметры компиляции, задает целевые платформы и версии инструментов, а также управляет подключением необходимых плагинов. Особое внимание уделяется подбору библиотек, которые включают:

– компоненты для работы с Android SDK;
– библиотеки навигации между экранами;
– поддержку асинхронных операций через сопрограммы;
– специализированные инструменты для работы с данными.

Реализация механизма авторизации начинается с проектирования структуры пользовательских данных, где ключевым элементом выступает разделение ролей (преподаватель/студент). Для этого создается модель пользователя, содержащая обязательные аутентификационные поля – уникальный идентификатор, номер телефона в качестве логина и хэшированный пароль, а также флаг роли, определяющий уровень доступа в системе. «Особое внимание уделяется безопасности хранения конфиденциальных данных – пароли никогда не сохраняются в открытом виде, а преобразуются через криптографические алгоритмы хэширования» [12].

«Процесс аутентификации реализуется через последовательность проверок: при вводе номера телефона и пароля система сначала валидирует формат данных, затем отправляет запрос к базе данных для поиска соответствий» [11]. Успешная проверка инициирует создание авторизационного токена, который сохраняется в защищенном хранилище устройства и используется для последующих запросов к API. Одновременно обновляется глобальное состояние приложения (isAuth), что позволяет другим модулям отслеживать текущий статус сессии.

«Механизм сессии строится на принципах минимально необходимых привилегий – после входа система загружает только те данные пользователя, которые требуются для работы конкретной роли» [25]. Для преподавателей это может быть список учебных групп, для студентов – активные тестовые задания. Состояние текущего пользователя (User) сохраняется в памяти приложения через паттерн Singleton или инструменты типа Android ViewModel, обеспечивая доступ к профилю из любого модуля без повторных запросов к серверу. Фрагмент модуля авторизации приведен на рисунке 8.

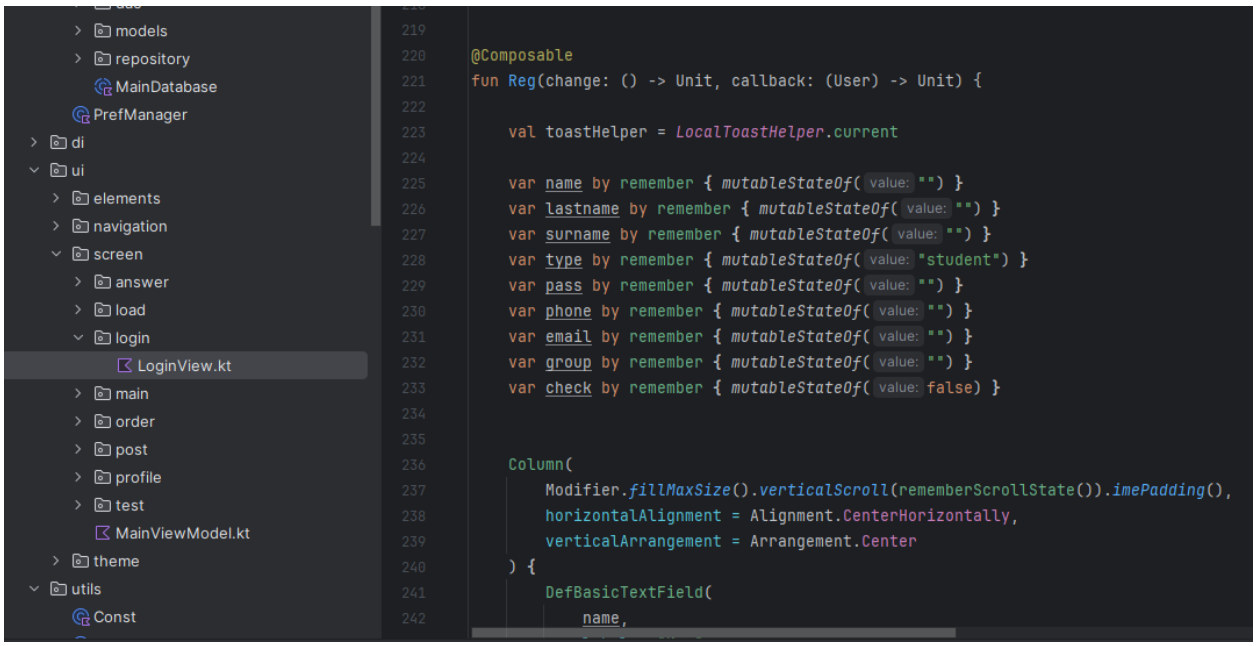

Рисунок 8 – Фрагмент модуля авторизации

Регистрация новых пользователей требует дополнительных уровней проверки – помимо стандартной валидации номера телефона и сложности пароля, система может предусматривать подтверждение через SMS-код или модерацию запросов преподавателями. Особенностью реализации является автоматическое определение роли на этапе регистрации через специальные пригласительные коды или утверждение администратором, что предотвращает несанкционированное создание преподавательских аккаунтов. Навигационный поток завершает процесс авторизации – после успешной проверки учетных данных система перенаправляет пользователя на соответствующий его роли основной экран (TeacherDashboard/StudentDashboard). Эта логика инкапсулирована в отдельный модуль маршрутизации, который учитывает не только роль, но и состояние "первого входа", что позволяет при необходимости показывать onboarding-экран или запрашивать дополнительные профильные данные. Все переходы сопровождаются анимацией для плавности пользовательского опыта.

2.4 Разработка основного функционала

«Центральный экран приложения проектируется как многофункциональная информационная панель, адаптирующаяся под роль пользователя» [12]. Для преподавателей интерфейс дашборда включает инструменты быстрого доступа к созданию учебных материалов и аналитике выполнения заданий, тогда как студенты видят преимущественно список активных тестов и учебных материалов. «Архитектура экрана строится на принципах реактивного программирования – при изменении данных в базе (например, публикации нового теста) интерфейс автоматически обновляется без необходимости ручного обновления» [4]. Реализация динамического контента требует разработки сложной системы фильтрации, позволяющей сортировать материалы по дате публикации, тематике или статусу выполнения. Особое внимание уделяется оптимизации запросов к серверу – применяется пагинация для постепенной загрузки контента и кэширование часто используемых данных. Визуальное оформление строится на карточном дизайне с четкой иерархией информации, где каждый элемент содержит превью содержания и ключевые метки (дедлайн, статус, количество вопросов). Модуль основного экрана приведен на рисунке 9.

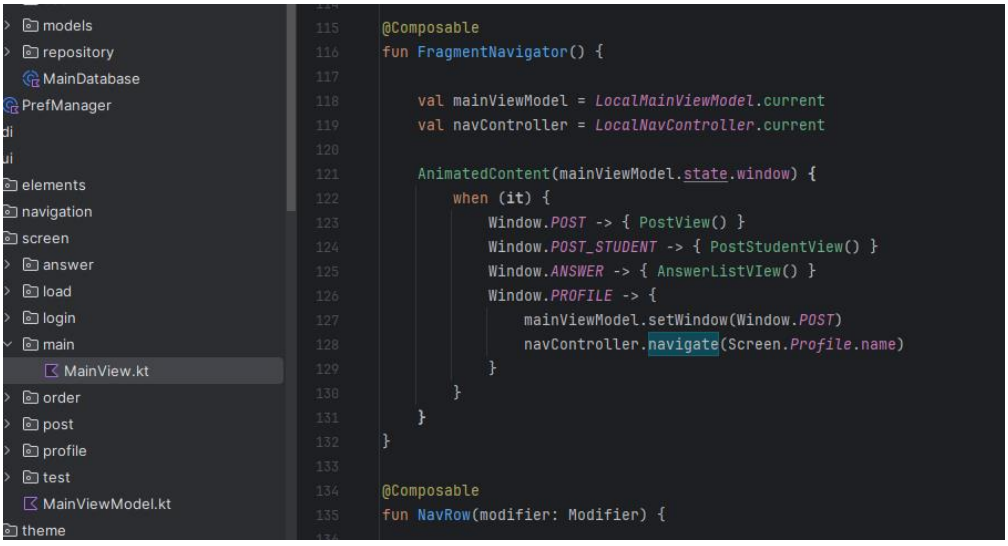

Рисунок 9 – Фрагмент модуля авторизации

Разработка модуля создания учебных материалов начинается с проектирования гибкой структуры данных, способной хранить разнородный контент – текстовые пояснения, тестовые вопросы различных типов (один правильный ответ, множественный выбор, открытый ответ) и прикрепленные файлы. Редактор тестов реализуется через систему динамических форм, где преподаватель может добавлять неограниченное количество вопросов, устанавливать баллы за каждый вариант ответа и настраивать параметры доступности. Техническая сторона публикации материалов включает валидацию введенных данных перед сохранением, обработку прикрепленных файлов (сжатие изображений, проверку форматов документов) и фоновую синхронизацию с сервером. «Особую сложность представляет механизм назначения тестов группам студентов – система должна обеспечить точное соответствие между выбранными учебными группами и списком студентов, имеющих доступ к материалу» [6]. Этот модуль приведен на рисунке 10.

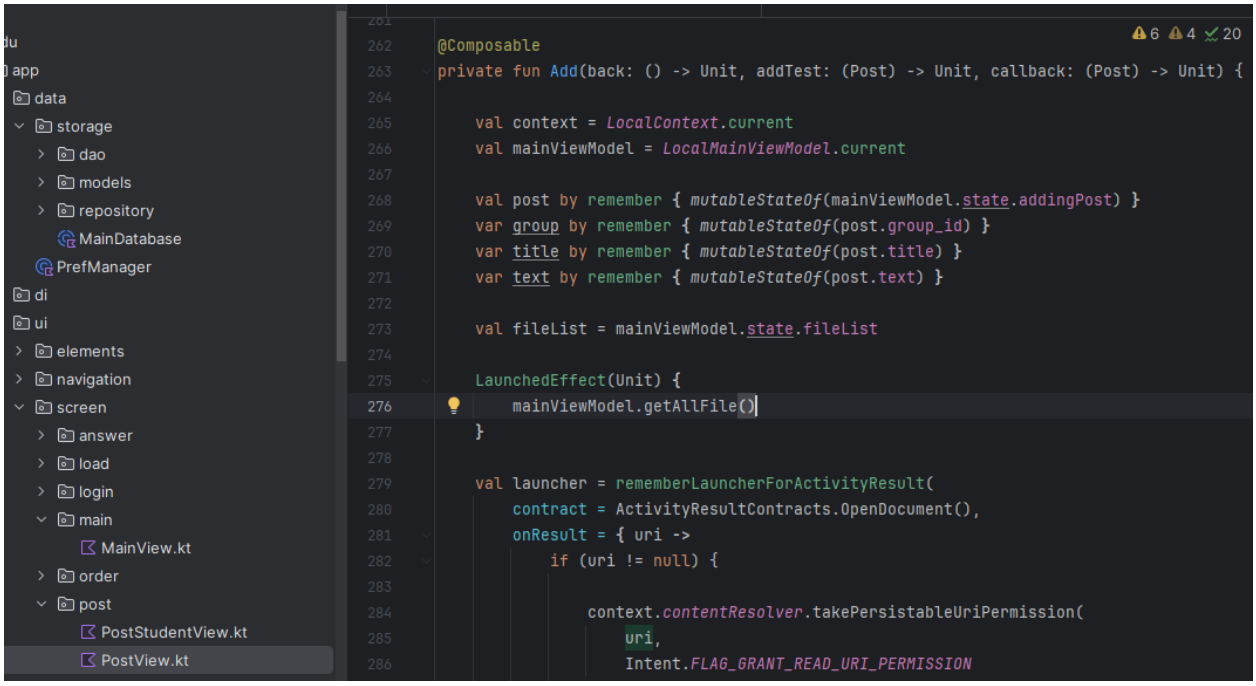

Рисунок 10 – Фрагмент модуля разработки тестов

Функционал прохождения тестов для студентов требует тщательной проработки пользовательского опыта (рисунок 11). Каждый тест отображается с четкими указаниями о времени выполнения, системе оценки и возможном количестве попыток. При разработке интерфейса решения вопросов применяются паттерны, минимизирующие случайные ошибки – подтверждение перед отправкой, возможность пометки сложных вопросов для последующего возврата, таймер обратного отсчета для временных ограничений.

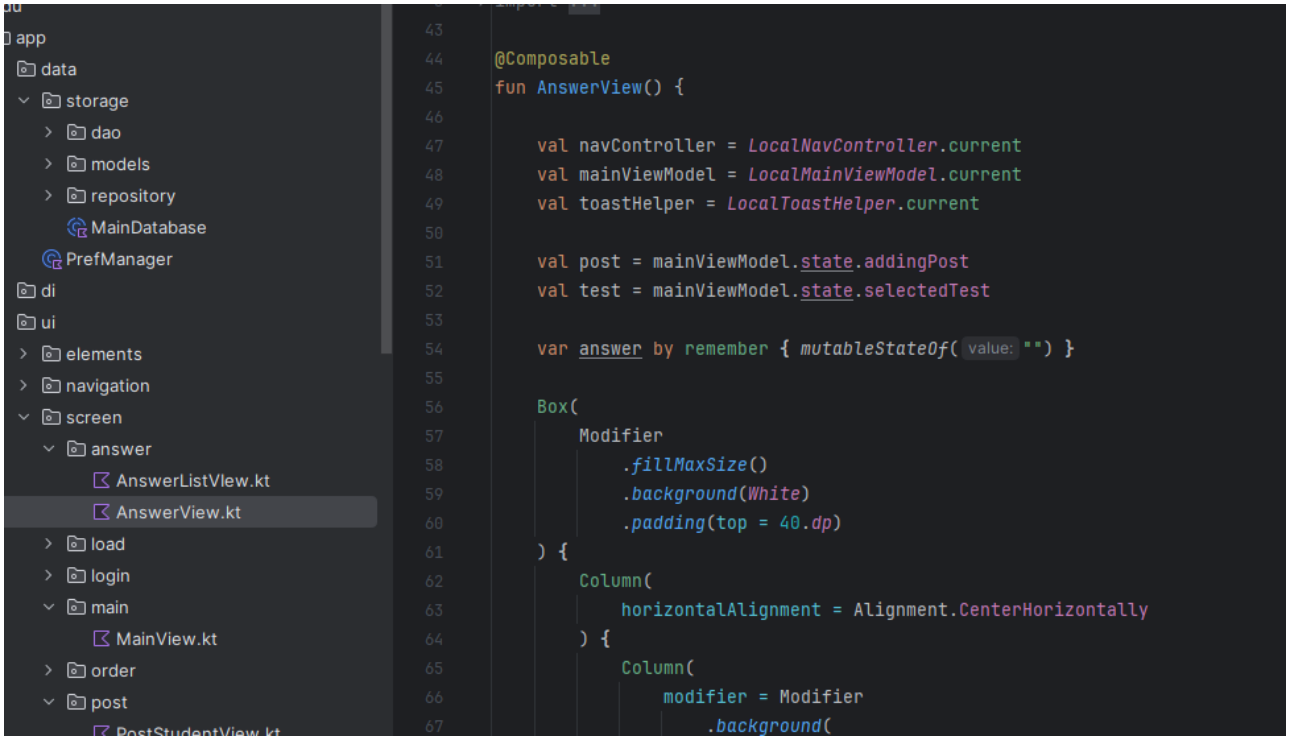

Рисунок 11 – Фрагмент модуля прохождения тестов

Система проверки ответов строится на комбинации автоматической и ручной оценки. Тестовые вопросы с однозначными ответами проверяются мгновенно по заранее заданным критериям, тогда как открытые задания требуют внимания преподавателя. Интерфейс проверки включает инструменты для комментирования конкретных фрагментов ответа, рубрикаторы для критериальной оценки и возможность отправки

доработанного задания на повторную проверку. Эта система приведена на рисунке 12.

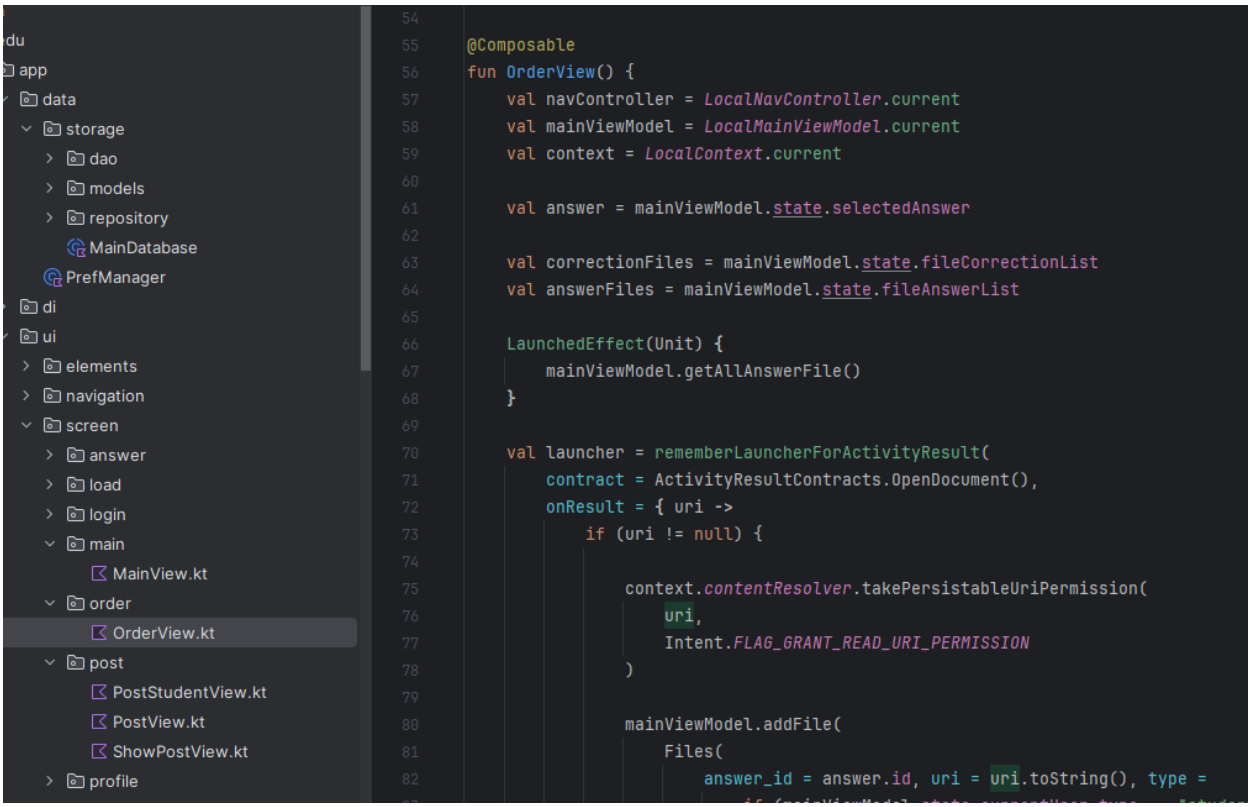

Рисунок 12 – Фрагмент модуля проверка тестов

Аналитический модуль предоставляет преподавателям наглядные отчеты по результатам тестирования – от общей статистики по группе до детализации по каждому вопросу.

Особенности синхронизации данных между клиентом и сервером играют ключевую роль в стабильности приложения. Реализуется механизм очереди запросов на случай потери соединения, позволяющий отложить отправку ответов до восстановления связи. Версионность тестов и результатов обеспечивает целостность данных при одновременной работе нескольких преподавателей с одним курсом.

Масштабируемость архитектуры закладывается через микросервисный подход к ключевым функциям. Каждый значимый модуль (редактор тестов,

система проверки, аналитика) разрабатывается как независимый компонент с четко определенным API, что упрощает добавление нового функционала в будущем. Например, система может быть расширена поддержкой мультимедийных вопросов или интеграцией с системами прокторинга для контроля честности выполнения заданий. Оптимизация производительности достигается за счет стратегий "ленивой" загрузки ресурсов, когда тяжеловесные элементы (графика, подробная статистика) подгружаются только при непосредственном обращении. Для работы с большими объемами тестовых данных применяются специализированные алгоритмы поиска и индексации, обеспечивающие быстрый отклик интерфейса даже на устройствах с ограниченными ресурсами. Система уведомлений дополняет основной функционал, оперативно информируя пользователей о новых тестах, изменении сроков сдачи или поступлении результатов проверки. «Реализация учитывает особенности мобильных устройств – от оптимизации энергопотребления фоновых процессов до персонализации частоты и формата оповещений в зависимости от пользовательских предпочтений» [14]. Непрерывное тестирование сопровождает разработку на всех этапах – от модульных тестов отдельных компонентов до интеграционных проверок всей системы. Особое внимание уделяется пограничным сценариям: одновременное редактирование теста несколькими преподавателями, прерывание выполнения теста из-за разряда батареи, работа при нестабильном интернет-соединении. Полученные результаты позволяют создавать устойчивую систему, готовую к реальным условиям эксплуатации в образовательных учреждениях.

«В современных версиях Android доступ к файловой системе организован через механизм Storage Access Framework (SAF), который обеспечивает безопасное и контролируемое взаимодействие приложений с пользовательскими файлами» [24]. Когда пользователь выбирает файл через системный файловый менеджер, приложение получает не прямой путь к файлу, а универсальный идентификатор (Uri), предоставляющий временный

доступ к содержимому. Такой подход исключает возможность несанкционированного доступа к защищенным разделам файловой системы, сохраняя при этом гибкость работы с документами из различных источников – локальных папок (Downloads, Pictures), съемных носителей (SD-карт) и облачных хранилищ (Google Drive, Dropbox). Работа с файлами приведена на рисунке 13.

```
val launcher = rememberLauncherForActivityResult(
    contract =
        ActivityResultContracts.GetContent()
) { uri: Uri? ->
    if (uri != null)
        uriToBitmap(context, uri) {
            mainViewModel.setImage(bitmapToBase64(it))
        }
}

Column(
    Modifier
        .fillMaxSize()
        .background(White)
        .verticalScroll(rememberScrollState()),
    horizontalAlignment = Alignment.CenterHorizontally
) {
```

Рисунок 13 – Работа с файловой системой

«Storage Access Framework не только ограничивает доступ приложений к файлам, но и стандартизирует работу с ними» [7]. Приложение взаимодействует с файлами через ContentResolver, используя полученный Uri для чтения или записи данных. Это позволяет системе контролировать каждую операцию и гарантировать, что доступ предоставляется только к явно выбранным пользователем файлам. Кроме того, SAF автоматически обрабатывает интеграцию с облачными сервисами, предоставляя единый интерфейс для работы с файлами независимо от их физического

расположения. Такой механизм особенно важен для образовательных приложений, где требуется работа с учебными материалами из разных источников без компрометации безопасности пользовательских данных.

По итогу был разработан весь необходимый функционал для приложения, исходя из выявленных задач.

Выводы по второй главе.

В данной главе была обоснована и реализована архитектура MVVM, которая обеспечила четкое разделение ответственности между компонентами приложения Использование современных технологий, таких как Kotlin, Android Jetpack, а также Room для локального хранения данных, позволило создать производительное и масштабируемое решение.

Глава 3 Тестирование работоспособности приложения

Функциональное тестирование является ключевым этапом проверки работоспособности приложения, направленным на верификацию соответствия реализованного функционала заявленным требованиям. В контексте образовательного приложения этот процесс охватывает все основные сценарии использования, включая авторизацию пользователей, управление учебными материалами, прохождение тестов и анализ результатов. Тестирование проводится на реальных устройствах, что позволяет оценить поведение системы в условиях, максимально приближенных к эксплуатационным.

Особое внимание уделяется проверке корректности работы функций для разных ролей – преподавателей и студентов. Для преподавательского интерфейса важна проверка создания и редактирования тестов, включая добавление вопросов различных типов, настройку параметров оценки и назначение заданий учебным группам. Студенческий функционал тестируется на предмет удобства прохождения тестов, отображения результатов и получения обратной связи.

Тестирование проводится с использованием как позитивных сценариев (стандартные условия работы), так и негативных (проверка обработки ошибок и граничных случаев). Это позволяет убедиться в стабильности приложения при неправильном вводе данных, отсутствии Интернет-соединения или других нештатных ситуациях. Результаты функционального тестирования служат основой для доработки приложения и подтверждения его готовности к использованию в учебном процессе.

Рассмотрим компонент для входа пользователей на рисунке 14.

Рисунок 14 – Вход пользователя

В данном окне осуществляется вход пользователя в систему с помощью данных указанных при регистрации. Если данные не находятся, то пользователю предлагается регистрация аккаунта. При переходе на экран регистрации, пользователь видит экран, представленный на рисунке 15.

Рисунок 15 – Регистрация пользователя

На этом экране в зависимости от типа пользователя, будет осуществлена регистрация в системе. Пользователи вводят персональные данные и указывают почту, пароль и номер телефона при регистрации. После нажатия кнопки “Регистрация”, пользователь заносится в базу данных и осуществляется вход в систему. На рисунке 16 представлен экран главного компонента приложения после входа.

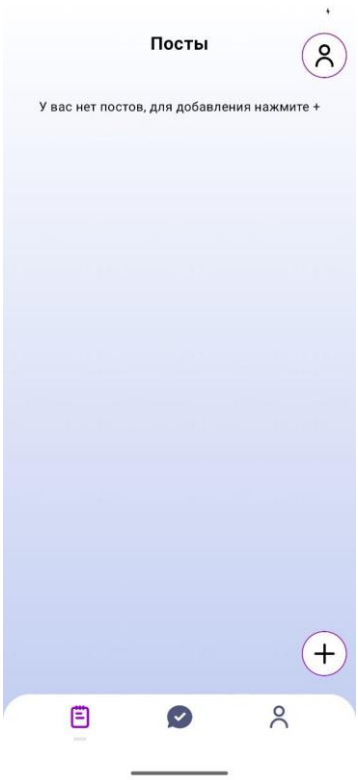

Рисунок 16 – Главный экран приложения

Свои данные пользователь может просматривать и редактировать во фрагменте профиля, а также выйти из аккаунта (рисунок 17).

Рисунок 17 – Фрагмент профиля пользователя

Для создания теста в главном экране приложения нужно выбрать кнопку “Добавить”, далее появляется экран, в котором вводим название теста и название группы, которому этот тест назначается. Также дополнительно к тесту можно прикрепить дополнительные материалы. Этот экран представлен на рисунке 18.

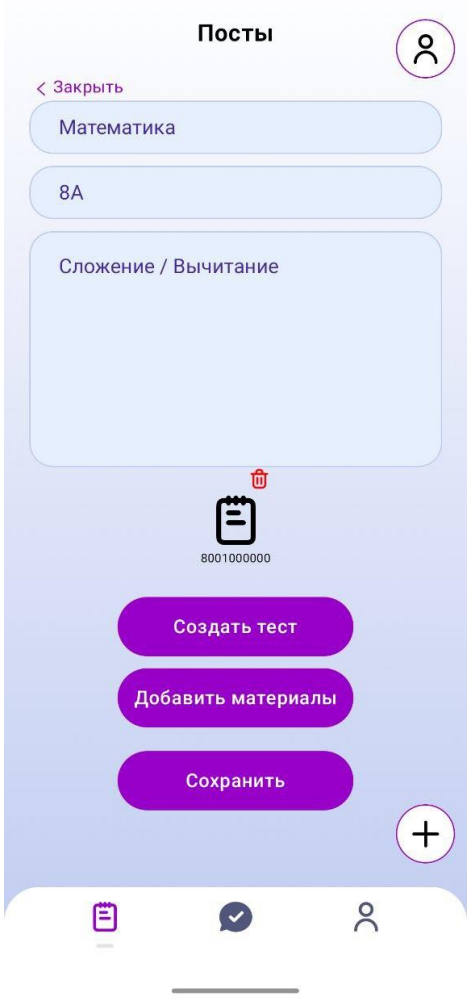

Рисунок 18 – Экран создания тестов

При нажатии кнопки “Создать тест” мы переходим к экрану, в котором добавляем вопросы и определяем сам вопрос и правильный ответ (рисунок 19).

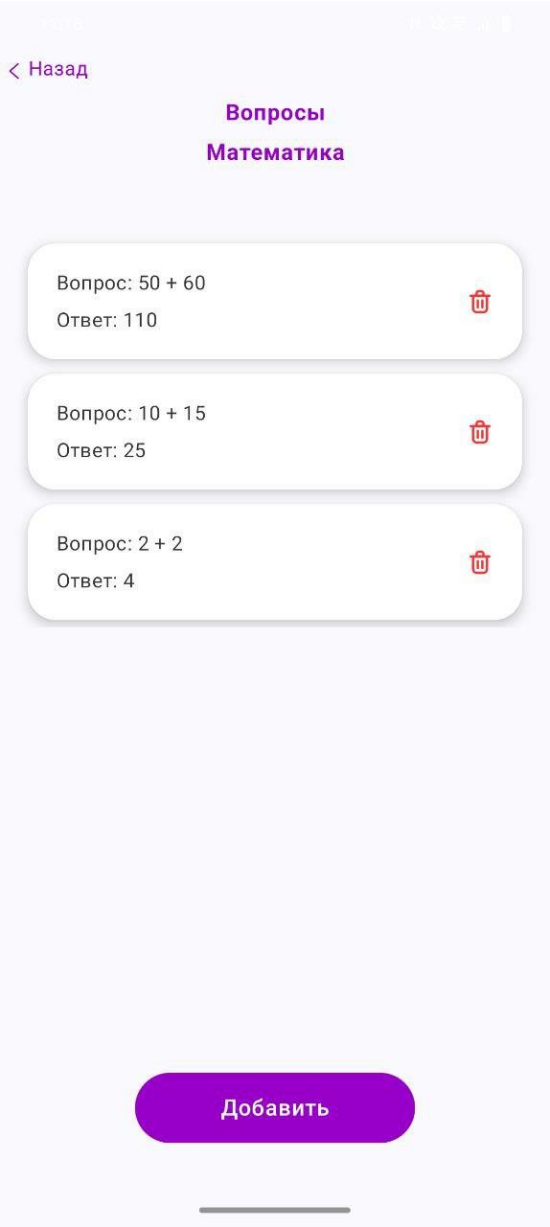

Рисунок 19 – Добавление вопросов

По итогу составления и назначение теста, у студента в главном экране появляется тест, который назначен для его группы и который можно проходить. Главный экран студента приведен на рисунке 20.

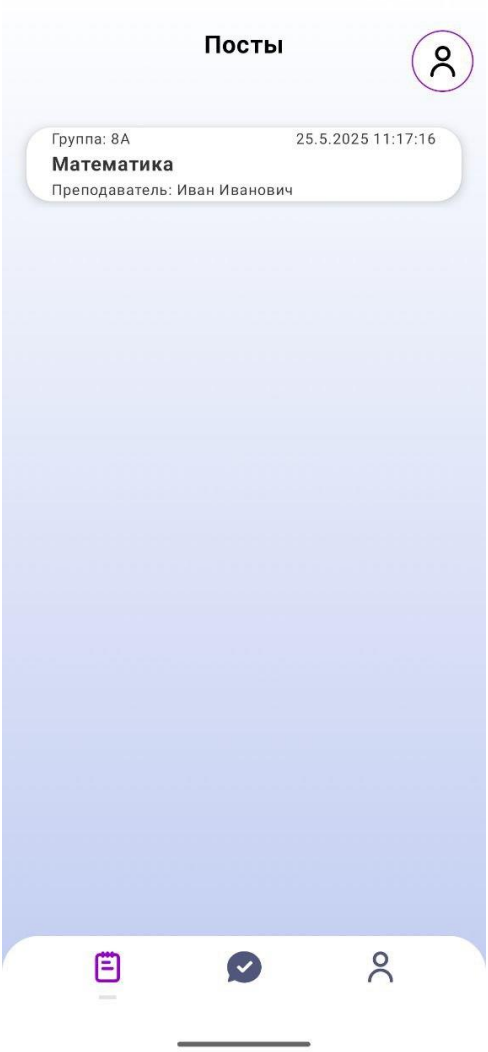

Рисунок 20 – Добавление вопросов

При переходе на тест, появляется содержание, прикрепленные материалы и список вопросов которые нужно пройти. При переходе на элемент вопроса, появляется поле для ввода ответа, исходя из правильности ответа, каждый элемент меняет свой цвет. Это приведено на рисунке 21.

Рисунок 21 – Прохождение теста

Преподаватель может видеть результаты теста, а также кто из студентов и когда его проходил (рисунок 22).

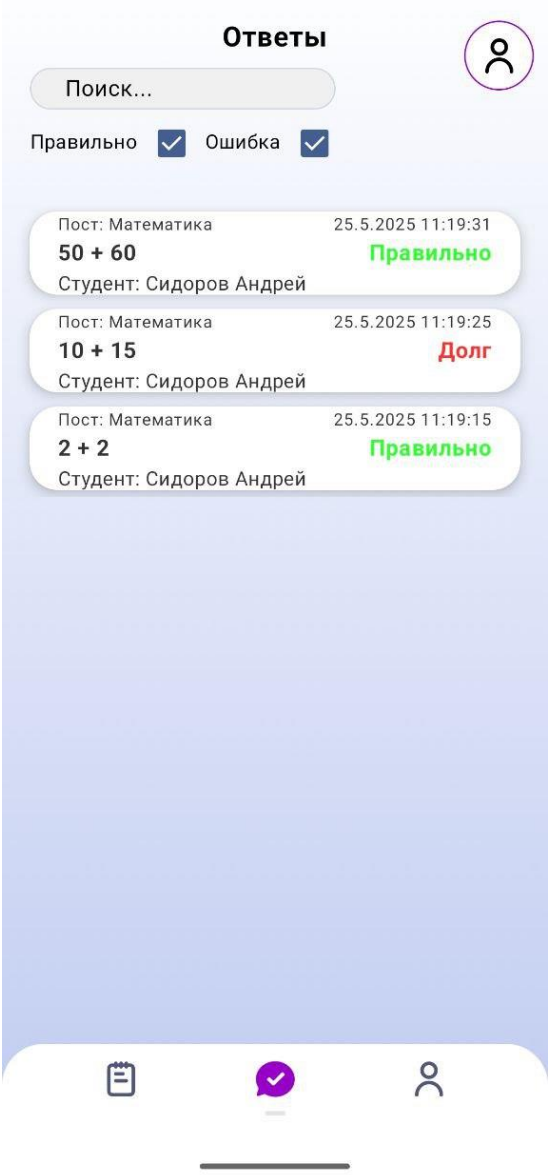

Рисунок 22 – Прохождение теста

По каждому из неправильных ответов, преподаватель может оставить дополнительный отзыв или задачи, которые должен выполнить студент. Для этого нажимается кнопка “Добавить задачи” и выбирается файл для прикрепления. Этот функционал приведен на рисунке 23.

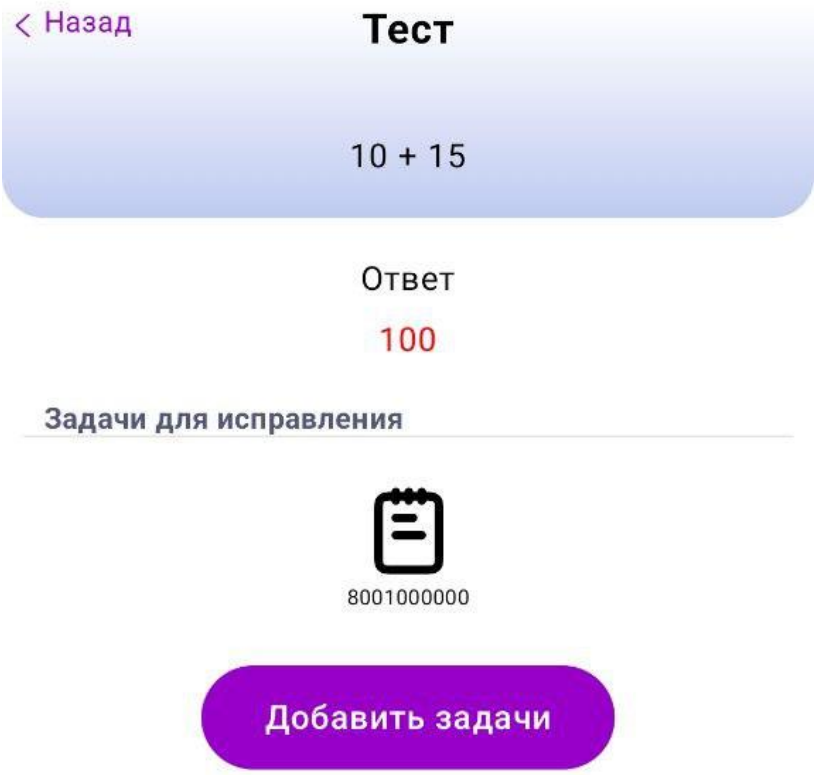

Рисунок 23 – Добавление дополнительных задач

В свою очередь студент видит задачу и прикрепляет ответ к ней. Преподаватель по итогу просматривает ответ и появляется возможность закрыть задачу (рисунок 24).

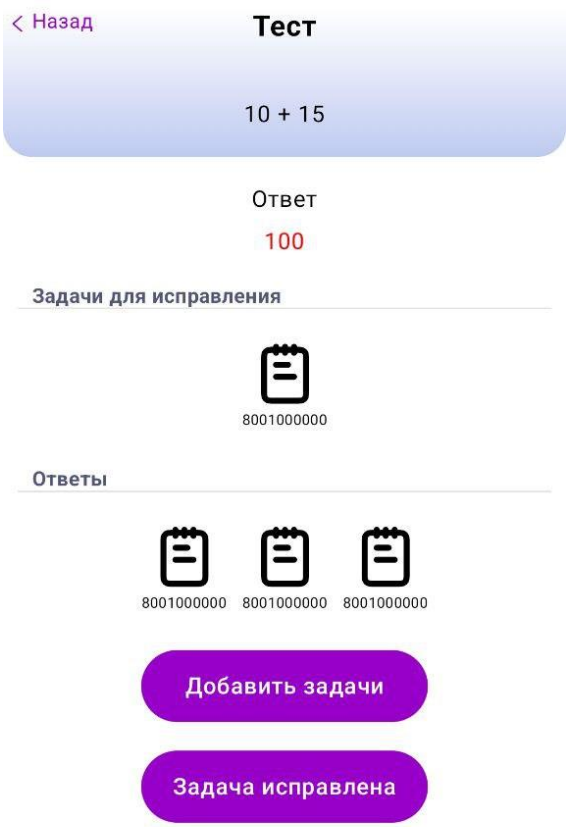

Рисунок 24 – Просмотр ответов на дополнительные задачи

Если дополнительная задача была закрыта, то по итогу в главных экранах преподавателя и студента появляются отметки об исправлении задачи. Экраны приведены на рисунках 25 и 26.

Рисунок 25 – Главный экран студента после исправления

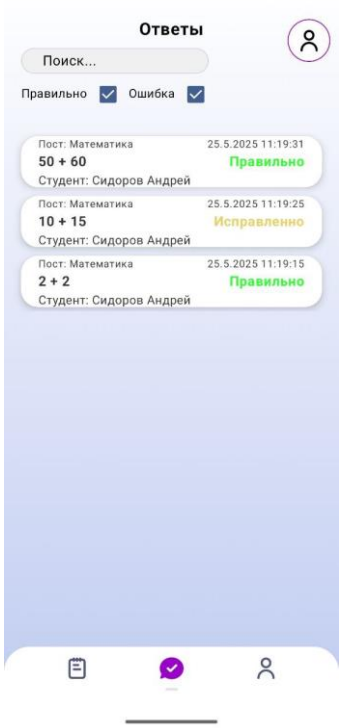

Рисунок 26 – Главный экран преподавателя после исправления

По итогу тестирования на реальном устройстве, приложение подтвердила стабильную работу всех ключевых функций: авторизация пользователей, создание и прохождение тестов, добавление задач, а также обмен файлами выполняются без ошибок. Интерфейс демонстрирует отзывчивость на различных устройствах, а бизнес-логика корректно обрабатывает данные.

Выводы по главе 3.

В ходе третей главы было протестировано мобильное приложение, которое выполняет все поставленные задачи. Был описан полный цикл процесса тестирования, а итоги показали полную работоспособность приложения.

Заключение

В рамках этой работы разработано и успешно реализовано мобильное приложение, предназначенное для организации учебного процесса. Приложение объединяет в себе ключевые функции, необходимые для эффективного взаимодействия преподавателей и студентов: управление группами, публикацию учебных материалов, отправку и проверку заданий, а также систему обратной связи.

Архитектурные решения, такие как паттерн MVVM и модульная структура, позволили достичь высокой гибкости, производительности и масштабируемости системы, а также упростили её дальнейшее развитие и поддержку.

Особое внимание уделялось удобству пользовательского интерфейса, что подтвердилось в ходе тестирования – приложение демонстрирует интуитивно понятную навигацию, быстрый отклик и стабильную работу на различных устройствах.

Использование современных технологий, таких как Kotlin и Android Jetpack, позволило разработать надежное, безопасное и производительное решение, полностью соответствующее актуальным стандартам и лучшим практикам мобильной разработки.

Результаты работы подтверждают, что приложение готово к внедрению в образовательных учреждениях. Оно не только автоматизирует рутинные процессы, но и способствует повышению вовлеченности участников за счет удобных инструментов взаимодействия.

В перспективе система обладает значительным потенциалом для расширения функциональности. Ее можно дополнить такими востребованными возможностями, как видеоконференции, углубленная аналитика успеваемости учащихся и интеграция с популярными образовательными платформами.

Список используемой литературы и используемых источников

1. Астапчук, В. А. Базы данных: проектирование и реализация : учебное пособие / В. А. Астапчук, Е. Н. Павенко, И. В. Эстрайх. – Новосибирск : НГТУ, 2023. – ISBN 978–5–7782–4917–2. – Текст : электронный // Лань : электронно–библиотечная система. – URL: https://e.lanbook.com/book/404294 (дата обращения: 04.06.2025). – Режим доступа: для авториз. пользователей. – С. 27.

2. Афанасьев А. В. Актуальность языка программирования Java в разработке мобильных приложений в 2020 году // Информационные технологии в процессе подготовки современного специалиста. – 2020. – С. 25–29.

3. Афанасьев А. Н., Войт Н. Н. Разработка компонентно–сервисной платформы обучения: диаграммы классов программного компонента сценария на UML–языке // Вестник Ульяновского государственного технического университета. – 2012. – № 2 (58). – С. 32–36.

4. Бежик А. А., Мажей Я. В. Архитектура приложений. Чем является? Для чего используется? Основные виды и критерии хорошей архитектуры // Столыпинский вестник. – 2022. – Т. 4. – № 9. – С. 4998–5008.

5. Вирт Н. Алгоритмы и структуры данных = Algoritms and data structure. – М.: Мир, 1989. – 360 с. – ISBN 5–03–001045–9.

6. Воронина М.И. Тестирование Android-приложений: от юнит-тестов до UI-автоматизации. - М.: Альфа-книга, 2024. - 416 с. - ISBN 978-5-9922-3054-1.

7. Вязовик Н., Жилин Е. Программирование на Java. – М.: Интуит.ру, 2003.

8. Гринченко, Н. Н. Проектирование моделей данных : учебное пособие / Н. Н. Гринченко, Н. И. Хизриева, С. Н. Баранова. – Рязань : РГРТУ, 2022. – 48 с. – Текст : электронный // Лань : электронно–библиотечная система. – URL: https://e.lanbook.com/book/380387 (дата обращения: 04.06.2025). – Режим доступа: для авториз. пользователей. – С. 4.

9. Громов А.И., Семенов П.К. Архитектурные паттерны в разработке Android-приложений. - М.: Юрайт, 2024. - 543 с. - (Серия: Высшее образование). - ISBN 978-5-534-15892-7. - Текст: электронный // Юрайт: электронная библиотека. - URL: https://urait.ru/book/arhitekturnye-patterny-v-razrabotke-android-prilozheniy-587789 (дата обращения: 15.06.2025).

10. Зиятдинова А., Староверова Н. А. Аналитический обзор и сравнение возможностей операционных систем для мобильных устройств // Фундаментальные исследования. – 2015. – № 9–2. – С. 227–231.

11. Иванов В. В., Лубова Е. С., Черкасов Д. Ю. Аутентификация и авторизация // Проблемы современной науки и образования. – 2017. – № 2 (84). – С. 31–33.

12. Козлов Е.Н. Безопасность данных в мобильных приложениях: практическое руководство. - СПб: БХВ-Петербург, 2023. - 280 с. - ISBN 978-5-9775-4102-3.

13. Микляев И. А., Ундозерова А. Н., Кудаева М. В. Универсальная логическая модель базы данных // Arctic Environmental Research. – 2010. – № 1. – С. 93–98.

14. Петров А.В. Современные технологии мобильной разработки: учебное пособие. - М.: ДМК Пресс, 2023. - 320 с. - ISBN 978-5-93700-158-2.

15. Практикум по информатике / Н. М. Андреева, Н. Н. Василюк, Н. И. Пак, Е. К. Хеннер. – 3–е изд., стер. – Санкт–Петербург : Лань, 2024. – 248 с. – ISBN 978–5–507–47299–4. – Текст : электронный // Лань : электронно–библиотечная система. – URL: https://e.lanbook.com/book/359810 (дата обращения: 04.05.2025). – Режим доступа: для авториз. пользователей. – С. 106.

16. Сидорова К.Л. Проектирование пользовательских интерфейсов образовательных приложений // Современные информационные технологии в образовании. - 2024. - № 1(15). - С. 45-52. - ISSN 2305-8405.

17. Шибанов, С.В. Автоматизированное проектирование пользовательских интерфейсов / С.В. Шибанов, А.А. Пашкин // Вестник

Пензенского государственного университета. – 2016. – № 4. – С. 67–73. – ISSN 2410–2083. – Текст : электронный // Лань : электронно–библиотечная система. – URL: https://e.lanbook.com/journal/issue/301580 – С. 1.

18. Android Studio [Электронный ресурс]. – Режим доступа: https://developer.android.com/studio/intro (дата обращения: 04.03.2025).

19. Diagrams.net [Электронный ресурс]. – Режим доступа: https://en.wikipedia.org/wiki/Diagrams.net (дата обращения: 02.04.2025).

20. Eeles P. Capturing architectural requirements // IBM Rational developer works. – 2005.

21. Firebase for Android [Электронный ресурс]. – Режим доступа: https://firebase.google.com/docs/android/setup (дата обращения: 27.02.2025).

22. Goadrich M. H., Rogers M. P. Smart smartphone development: iOS versus Android // Proceedings of the 42nd ACM technical symposium on Computer science education. – 2011. – С. 607–612.

23. Mobile Operating System Market Share [Электронный ресурс]. – Режим доступа: https://gs.statcounter.com/os–market–share/mobile/worldwide (дата обращения: 15.04.2025).

24. Nimodia C., Deshmukh H. R. Android operating system // Software Engineering. – 2012. – Т. 3. – № 1. – С. 10.

25. OneSignal SDK [Электронный ресурс]. – Режим доступа: https://documentation.onesignal.com/docs/android–sdk–setup (дата обращения: 17.03.2025).