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))

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему: «Разработка многофункционального мобильного обучающего приложения»

Студент Д.А. Джафаров
(И.О. Фамилия) (личная подпись)

Руководитель к.т.н., доцент Д.Г. Токарев
(ученая степень, звание, И.О. Фамилия)

Консультант к.п.н., доцент С.А. Гудкова
(ученая степень, звание, И.О. Фамилия)

Тольятти 2024

Аннотация

Название темы бакалаврской работы: «Разработка многофункционального мобильного обучающего приложения».

Целью данной дипломной работы является разработка многофункционального мобильного обучающего приложения, обеспечивающего эффективную поддержку процесса обучения на платформе Android с возможностью работы в условиях ограниченной или отсутствующей сетевой связи.

Актуальность работы обусловлена высоким спросом на образовательные технологии и отсутствием удобных, интуитивно понятных мобильных приложений для самостоятельного обучения. Работа структурирована следующим образом: введение, три главы, заключение и приложения.

Выпускная квалификационная работа состоит из нескольких глав, каждая из которых посвящена различным аспектам разработки многофункционального мобильного обучающего приложения.

В первой главе рассматривается актуальность разработки, включая проблемы доступности и индивидуализации обучения в традиционных учебных средах. Также проводится сравнительный анализ платформ для создания мобильных приложений, выбирается оптимальная среда разработки, и обсуждаются особенности и специфика разработки приложения.

Вторая глава посвящена проектированию мобильного обучающего приложения. Основной платформой для разработки выбрана Android Studio. В ходе анализа выбран язык программирования Java [11] в сочетании с интегрированной средой разработки Android Studio, как наиболее подходящие для создания обучающих приложений.

Третья глава описывает процесс реализации приложения, включая разработку интерфейса, создание обучающих модулей и интеграцию интерактивных элементов. Описаны методы и подходы к тестированию функциональности приложения и его пользовательского интерфейса.

Abstract

The title of the graduation work is: «Development of a multifunctional mobile educational application»

The graduation work is devoted to creating a mobile educational application for the Android platform that supports the learning process even under conditions of limited or no network connectivity.

The graduation work consists of an introduction, three chapters, a conclusion, and appendices.

The key issue of the graduation work is the development of an educational application that addresses the high demand for educational technologies and the lack of intuitive mobile applications for self-study in Russian.

The first chapter examines the relevance of the development, focusing on accessibility and individualized learning in traditional educational environments. It includes a comparative analysis of platforms for mobile application development, selection of the optimal development environment, and discussion of the features and specifics of application development.

The second chapter is dedicated to the design phase of the mobile educational application. Android Studio is selected as the primary development platform, and Java is chosen as the programming language due to its suitability for educational applications.

The third chapter covers the implementation process, including interface development, creation of educational modules, and integration of interactive elements. This chapter also details the methods and approaches used to test the functionality and user interface of the application.

In conclusion, this work is significant for addressing the lack of effective mobile educational tools in the Russian language, providing a solution that can function independently of network connectivity. The approaches and solutions discussed can be applied to the broader context of educational technology development.

Содержание

Введение

В наше время мобильные приложения стали частью нашего быта, особенно в области образования и саморазвития, предоставляя улучшенные возможности для обучения. Эти инструменты обеспечивают доступ к образовательным ресурсам и позволяют учиться в удобном темпе, что выделяет их как ключевой элемент в повышении образовательной доступности и гибкости.

В рамках данной бакалаврской работы актуальной является разработка мобильного обучающего приложения, способного функционировать в оффлайн режиме и изолированных сетях, отвечающего современным требованиям к интерактивности и эффективности образовательных процессов. Результатом этой работы является приложение, позволяющее пользователям эффективно управлять своим учебным временем, планировать учебные задачи и отслеживать их выполнение на мобильных устройствах.

Для достижения этой цели были выполнены следующие задачи.

1. Анализ существующих образовательных приложений для определения ключевых требований к новому продукту.
2. Проектирование архитектуры и интерфейса мобильного приложения с учетом возможности работы в оффлайн режиме.
3. Реализация функциональности приложения, включая модули для создания и управления учебными задачами.
4. Тестирование приложения для проверки его функциональности и удобства использования.

Работа организована в три основные главы, каждая из которых описывает этапы разработки и тестирования приложения, а также предоставляет анализ конкурентов и их решений. Это исследование имеет значительную практическую ценность, поскольку способствует улучшению процесса обучения за счет использования современных мобильных технологий, особенно в средах с ограниченным доступом к сети.

1 Актуальность разработки многофункционального обучающего приложения

1.1 Проблема доступности и индивидуализации обучения в традиционных учебных средах

Современное образование сталкивается с рядом серьезных вызовов, ключевым из которых является неспособность традиционных учебных заведений предоставить гибкие и индивидуальные образовательные ресурсы для студентов с разнообразными потребностями. Эта проблема особенно остро стоит перед учащимися, которые из-за своих уникальных образовательных целей, стилей обучения или жизненных обстоятельств не могут полностью интегрироваться в стандартные учебные программы.

Доступ к образовательным ресурсам во многом зависит от географического положения, социально-экономического статуса и инфраструктурных возможностей региона. Учащиеся в отдалённых и малообеспеченных районах часто сталкиваются с ограничениями, которые снижают их шансы на получение качественного образования. Такие ограничения включают недостаток квалифицированных учителей, отсутствие современных учебных материалов и технологическую отсталость.

Традиционные методы обучения часто основываются на унифицированных подходах, которые не учитывают индивидуальные особенности и потребности каждого студента. Это приводит к тому, что ученики с нетрадиционными стилями обучения или особыми образовательными потребностями могут ощущать себя отстающими или не в состоянии полностью усваивать материал в соответствии со стандартной программой.

В ответ на эти вызовы появляется потребность в разработке мобильного обучающего приложения, которое могло бы предложить гибкий, доступный и индивидуализированный подход к образованию. Такое приложение должно интегрировать адаптивные технологии обучения, которые позволяют

модифицировать образовательный контент и методы подачи материала в зависимости от уникальных потребностей и предпочтений пользователя.

Предлагаемое мобильное приложение будет использовать алгоритмы машинного обучения для анализа поведения пользователя и адаптации учебных материалов, включая интерактивные задания, видеоуроки и самостоятельные тесты для проверки знаний. Приложение также будет включать инструменты для отслеживания прогресса и обратной связи, что позволит студентам не только видеть свои успехи, но и определять слабые стороны для дальнейшего улучшения.

Разработка такого приложения способствует улучшению качества образования, делая его более доступным, эффективным и адаптированным к нуждам каждого учащегося. Это направление развития образовательных технологий открывает новые перспективы для обучения, делая его максимально персонализированным и удобным.

1.2 Сравнительный анализ платформ для создания мобильных приложений

На современном рынке мобильных технологий представлено множество операционных систем, каждая из которых обладает своей уникальной популярностью. Разработка приложений исключительно под одну операционную систему может ограничить аудиторию и потенциально снизить успех проекта, так как большее количество пользователей, испытавших приложение, увеличивает шансы на его востребованность. Следовательно, важно тщательно выбрать операционную систему для разработки приложения. Диаграмма на рисунке 1 демонстрирует статистику использования трех наиболее популярных мобильных операционных систем за последние годы.

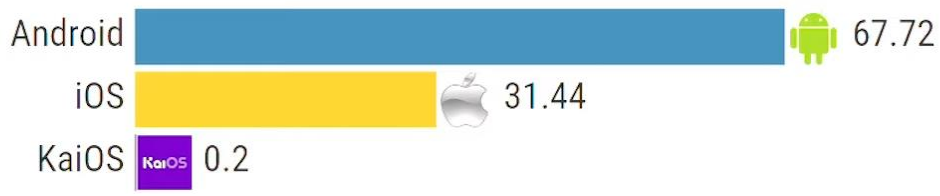


Рисунок 1 – Диаграмма доли рынка мобильных операционных систем на рынке смартфонов

На диаграмме представлены три ведущие мобильные операционные системы, которые заслуживают внимания при разработке мобильных приложений:

- KaiOS,
- iOS,
- Android.

KaiOS – это относительно новая операционная система, разработанная компанией KaiOS Technologies. Система специально разработана для кнопочных телефонов и устройств с базовыми функциями. Основное преимущество KaiOS заключается в том, что она обеспечивает доступ к интернет-сервисам и приложениям, таким как WhatsApp, YouTube и Google Maps, на устройствах, которые традиционно не поддерживают сложные функции смартфонов. С момента своего появления KaiOS заняла значительную нишу на рынках развивающихся стран, предлагая недорогие, но функциональные решения для доступа к цифровым сервисам. На текущий момент в KaiStore насчитывается более 500 тысяч приложений, что открывает определенные возможности для разработчиков, заинтересованных в целевой аудитории этой платформы. Тем не менее, учитывая ограниченный охват в мировом масштабе и снижение интереса к кнопочным телефонам, разработка под KaiOS может не предложить широкого рыночного потенциала.

«iOS — операционная система, разработанная компанией Apple для своих портативных устройств. Впервые она появилась на смартфонах iPhone и плеерах iPod в 2007 году, а с 2010 года стала устанавливаться на планшетах iPad» [9]. iOS

известна своей высокой стабильностью, безопасностью и оптимизированным пользовательским интерфейсом. App Store, магазин приложений для iOS, содержит более двух миллионов приложений и предлагает разработчикам возможности для монетизации через встроенные покупки и подписки. Однако высокие требования к качеству приложений и стоимость вступления в программу разработчиков могут быть барьером для новичков. Несмотря на эти факторы, разработка под iOS остается выгодной из-за лояльности пользователей и их готовности платить за качественный контент.

Android – самая распространенная операционная система в мире, которая используется на множестве устройств от различных производителей, включая смартфоны, планшеты, телевизоры и часы. Эта система, разработанная Google, отличается открытостью и гибкостью, предоставляя разработчикам широкие возможности для создания разнообразных приложений. Google Play, основной магазин приложений для Android, насчитывает более трех миллионов приложений, что делает его важной платформой для достижения широкой аудитории. Помимо низкой стоимости входа для разработчиков, Android предлагает меньшие ограничения на разработку и распространение приложений, что делает эту платформу особенно привлекательной для инноваций и экспериментов в мобильной разработке.

Для облегчения выбора операционной системы была создана таблица 1, в которой собраны ключевые преимущества и недостатки рассмотренных выше операционных систем.

Таблица 1 – Сравнение и анализ мобильных операционных систем

Критерии оценки	KaiOS	iOS	Android
Рыночная доля	-	+	++
Возможности и функции	-	+	++
Открытость для разработчиков	-	-	+
Количество доступных приложений	-	++	++
Поддерживаемые устройства	-	++	++
Защищенность данных	-	++	+
Способы получения дохода с продукта	-	+	+
Легкость разработки	-	+	++

На основе данных, представленных в таблице сравнения основных характеристик мобильных операционных систем, было принято решение ориентировать разработку на устройства с операционной системой Android. Этот выбор обусловлен тем, что Android лучше всего отвечает требованиям будущей системы, демонстрируя преимущества в таких важных аспектах, как широкая доля рынка, большое количество доступных приложений, высокая степень поддержки различных устройств, и относительная легкость разработки. Эти факторы делают Android идеальной платформой для достижения максимального охвата и функциональности будущего приложения.

1.3 Выбор среды разработки

Среди существующих платформ для мобильной разработки можно выделить следующие продукты:

– Android Studio,
– Eclipse & Android Development Tools (ADT),
– Visual Studio & Xamarin,

– Flutter,

– React Native.

Рассмотрим каждую платфому для разработки по отдельности:

«Android Studio — официальная интегрированная среда разработки (IDE) для разработки приложений Android. Она подходит для взаимодействия на языках Java и Kotlin. С её помощью разработчики создают приложения для мобильных, планшетов, телевизоров, часов и других устройств.

IDE содержит инструменты для разработки, отладки, тестирования и отслеживания производительности приложений. Android Studio — бесплатная, работает на Windows, Mac и Linux. Приложения можно сразу публиковать в магазине Google Play» [3].

Android Studio имеет встроенный статический анализатор кода Lint, задача которого идентифицирует потенциальные проблемы с производительностью и совместимостью версий, делая её незаменимым инструментом для разработчиков, стремящихся к созданию качественных Android-приложений.

Eclipse — это многопрофильная среда разработки с открытым исходным кодом, управляемая Eclipse Foundation. Хотя первоначально не предназначенная для мобильной разработки,

Eclipse может быть адаптирована под эти нужды с помощью плагина ADT, который добавляет специализированные инструменты для Android разработки. Помимо базовых возможностей IDE, таких как редактирование кода, отладка и профилирование, ADT вносит удобства для работы с Android, включая интеграцию с Android SDK, создание и управление AVD (Android Virtual Devices), и многие другие функции.

Однако, из-за узкоспециализированного фокуса Android Studio, Eclipse с ADT постепенно теряет свою популярность среди разработчиков Android.

«Xamarin — это платформа с открытым исходным кодом, предназначенная для построения современных производительных приложений для iOS, Android и Windows с .NET. Платформа Xamarin представляет собой уровень абстракции, который обеспечивает управление взаимодействием между общим кодом и

кодом базовой платформы. Xamarin выполняется в управляемой среде, которая реализует такие возможности, как выделение памяти и сборка мусора.

Благодаря Xamarin в среднем 90 % кода приложения может использоваться без изменений на разных платформах. С помощью этого шаблона разработчик может написать всю бизнес-логику на одном языке (или использовать существующий код приложения), но при этом получить характеристики производительности, оформление и поведение, характерные для каждой соответствующей платформы.

Приложения Xamarin можно писать на ПК или Mac и компилировать в собственные пакеты приложений, например в файлы с расширением .apk для Android или .ipa для iOS» [14].

«Flutter — комплект средств разработки и фреймворк с открытым исходным кодом для создания мобильных приложений под Android и iOS, веб-приложений, а также настольных приложений под Windows, macOS и Linux с использованием языка программирования Dart, разработанный и развиваемый корпорацией Google» [20]. Flutter предлагает мощные возможности для реализации динамичных и адаптивных пользовательских интерфейсов с компонентной архитектурой и разнообразными предустановленными виджетами.

Особенностью Flutter является его рендеринговый движок, который рисует интерфейс напрямую на канве, что обеспечивает высокую производительность и плавность анимаций. Фреймворк поддерживает горячую перезагрузку, что позволяет мгновенно видеть результаты изменений в коде без необходимости полной перекомпиляции и перезапуска приложения.

Flutter активно развивается и поддерживается сообществом, что делает его привлекательным выбором для создания красивых, эффективных и функциональных приложений.

«React Native - это популярная платформа мобильных приложений на основе JavaScript, которая позволяет создавать мобильные приложения с собственным интерфейсом для iOS и Android. Фреймворк позволяет создавать

приложения для различных платформ, используя одну и ту же кодовую базу.

React Native был впервые выпущен Facebook в качестве проекта с открытым исходным кодом в 2015 году. Всего за пару лет он стал одним из лучших решений, используемых для мобильной разработки. Разработка React Native используется для поддержки некоторых ведущих мобильных приложений в мире, включая Instagram, Facebook и Skype» [13]. React Native поддерживает модульную архитектуру и декларативный стиль программирования, что облегчает масштабирование приложений и их поддержку. Кроме того, благодаря широкому сообществу и библиотеке готовых компонентов и плагинов, разработчики могут быстро реализовывать сложные функциональные возможности.

React Native также известен своей возможностью «горячей перезагрузки» и поддержкой множества сторонних библиотек, что делает его мощным инструментом для создания мобильных приложений с богатым пользовательским интерфейсом и отличной пользовательской функциональностью.

Для упрощения процесса выбора наиболее подходящей среды разработки была создана таблица 2. В этой таблице собраны и систематизированы ключевые преимущества и недостатки основных платформ, рассмотренных ранее. Это позволяет более объективно оценить каждую из них с учетом специфики вашего проекта. Подробный анализ этих характеристик поможет принять обоснованное решение при выборе оптимальной платформы для разработки мобильного приложения.

Таблица 2 – Сравнительный анализ сред разработки под ОС Android

Критерии оценивания	Android Studio	Eclipse & ADT	Visual Studio & Xamarin	Flutter	React Native
Специализация	+	-	-	+	-
Многоязычная поддержка	+	+	+	+	+
Взаимодействие с другими системами	+	-	+	+	+
Эффективность выполнения задач	+	-	+	+	-
Простота и эргономичность интерфейса	+	-	+	+	+
Наличие и доступность материалов	+	-	-	-	+
Межплатформенная совместимость	-	-	+	+	+
Активность и помощь сообщества	+	-	-	-	+

Изучив таблицу сравнения и рассмотрев основные критерии, можно заключить, что Android Studio является предпочтительным выбором для разработки приложений под Android благодаря её специализированности, высокой производительности, удобному пользовательскому интерфейсу, широкой поддержке языков программирования (включая Kotlin и Java), а также обширному доступу к ресурсам и поддержке сообщества. Данные преимущества делают Android Studio наиболее адекватной средой для создания качественных и функциональных приложений для платформы Android, которые гарантируют эффективность [15] и удобство разработки.

1.4 Особенности и специфика разработки

Android — это масштабная и разнообразная платформа, требующая от разработчиков глубокого технического понимания и широкого набора навыков. Создание приложений для этой операционной системы включает в себя не только использование различных языков программирования, но и освоение уникальных принципов архитектуры системы. Разработчики сталкиваются с множеством технических вызовов, таких как обеспечение совместимости с различными устройствами, оптимизация производительности и управление ресурсами. Кроме того, учитывать необходимо и множество специфических требований и рекомендаций, предъявляемых к разработке приложений для Android, чтобы обеспечить безупречную работу программы на различных устройствах и версиях операционной системы.

1. Особенности хранения и работы с приложениями - одна из ключевых особенностей Android заключается в том, что после установки приложение может занимать в памяти устройства в два или даже в четыре раза больше места, чем его исходный размер. Это связано с процессом распаковки и установки приложения, включая оптимизацию выполнения и компиляцию во время установки. Также при работе с файлами на встроенной флеш-памяти устройства, необходимо учитывать, что скорость доступа к файлам может сильно снижаться при низком количестве свободного места. Это объясняется особенностями управления флеш-памятью, где процессы записи и удаления данных становятся менее эффективными при заполнении памяти.

2. Управление памятью и процессами - Android управляет оперативной памятью так, что процессы приложений могут использовать ограниченное количество памяти в зависимости от устройства. Это ограничение помогает оптимизировать работу системы при одновременном выполнении множества задач.

3. Многозадачность и управление приложениями - Android позволяет запускать множество приложений одновременно, при этом операционная

система управляет доступной оперативной памятью и ресурсами процессора. В любой момент времени активным может быть только одно приложение, показываемое на экране, но пользователь может легко переключаться между приложениями.

На рисунке 2 показана внутренняя архитектуры операционной системы Android.

1. Ядро Linux - на самом базовом уровне находится ядро Linux, которое предоставляет системе основные сервисы, такие как управление памятью, процессами и драйверами устройств. Ядро обеспечивает стабильность и безопасность всей системы.
2. Библиотеки и Android Runtime - выше ядра располагаются основные системные библиотеки, включая те, которые отвечают за аудио, видео и обработку изображений. Рядом с библиотеками находится Android Runtime (ART), который заменил Dalvik в последних версиях Android. ART оптимизирует приложения с помощью компиляции в машинный код при установке, что позволяет увеличить производительность приложений и уменьшить потребление батареи
3. Framework приложений - На следующем уровне находится Framework приложений, который представляет собой набор API, предоставляющих разработчикам доступ к функциям устройства, таким как камера, GPS и данные сенсоров. Этот слой также управляет жизненным циклом приложений, обеспечивая их стабильную работу в различных условиях.
4. Приложения - На верхнем уровне архитектуры находятся приложения. Это могут быть как предустановленные программы, так и приложения, загруженные пользователем. Android предоставляет разработчикам большую свободу в создании приложений, которые могут взаимодействовать с различными системными компонентами и другими приложениями через разработанные API.

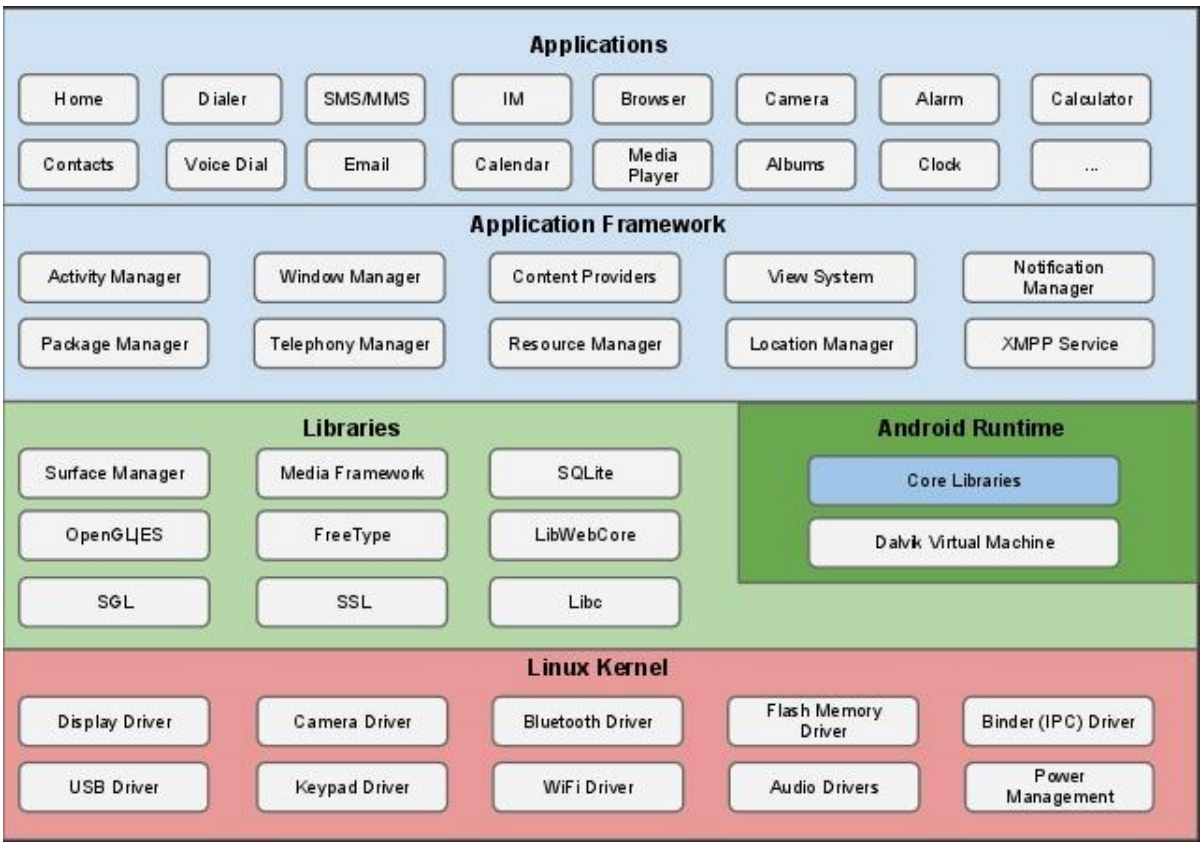


Рисунок 2 – Архитектурное представление операционной системы Android

В мобильной операционной системе Android, приложения управляются через специальные компоненты, известные как «Activity», которые выполняют функции аналогичные окнам в операционной системе Windows. Эти Activity служат в качестве основного интерфейса для взаимодействия пользователя с приложением. Каждое Activity имеет свой уникальный жизненный цикл, который описывает различные состояния, через которые оно проходит — от момента создания до его завершения. Этот жизненный цикл включает в себя ряд ключевых этапов, таких как запуск, пауза, возобновление и уничтожение. Ниже на рисунке 3 наиболее наглядно отображен жизненный цикл приложения в ОС Android.

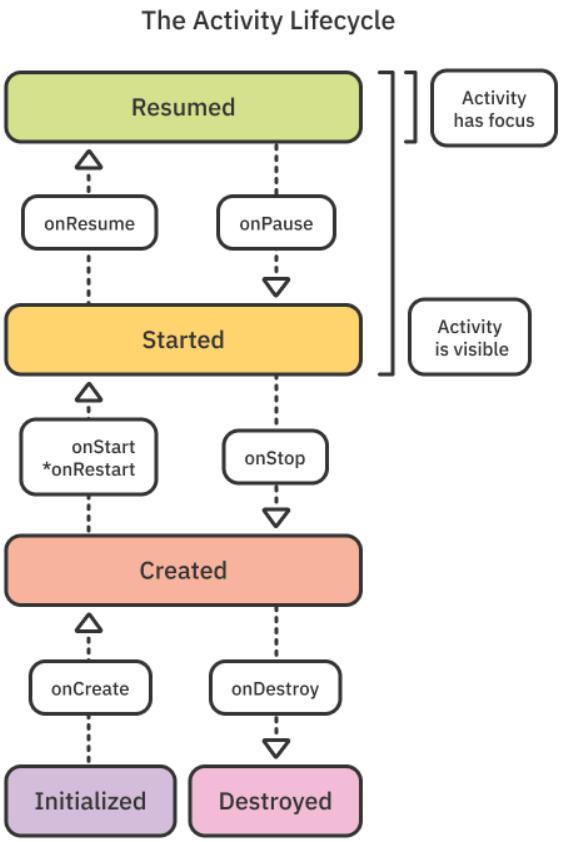


Рисунок 3 – Жизненный цикл приложения в ОС Android

Когда впервые запускается Activity в Android приложении, система вызывает метод onCreate(). В этот момент Activity начинает своё существование. Здесь выполняется вся начальная настройка: создается интерфейс, инициализируются переменные и готовятся данные. После того как начальная настройка завершена, система вызывает onStart(), делая Activity видимым для пользователя. Если Activity уже было запущено и остановлено, то вместо onStart() может быть вызван onRestart(), что также приведёт к переходу в состояние видимости и активности.

Следующим шагом в жизни Activity является onResume(). Данный метод вызывается сразу после onStart() или когда Activity возвращается в активное состояние после паузы, вызванной другими окнами или диалогами, которые забирали фокус. В onResume() Activity полностью взаимодействует с пользователем, что делает его идеальным моментом для запуска анимаций, музыки или видео.

Если появляется другое Activity или приложение переходит в фоновый режим, то вызывается метод onPause(). В этот момент следует сохранить все необходимые данные и приостановить действия, которые не должны продолжаться, пока пользователь не взаимодействует непосредственно с приложением. «Если Activity становится полностью невидимым, следующим методом будет onStop(). Это означает, что пользователь либо закрыл Activity, либо перешел на другой экран, который полностью его перекрыл» [21].

Конечным этапом в жизненном цикле Activity является onDestroy(). Этот метод вызывается, когда Activity уничтожается системой для освобождения ресурсов или когда пользователь явно закрывает приложение. Здесь происходит окончательная очистка: освобождаются ресурсы, закрываются соединения к базам данных.

Вывод по 1 главе:

В ходе анализа проблем доступности и персонализации обучения были выявлены ключевые ограничения, характерные для традиционных учебных сред. На основе этого анализа было решено разработать мобильное приложение, способное предложить гибкий и индивидуализированный подход к обучению.

Сравнительный анализ существующих мобильных операционных систем и инструментов разработки показал, что платформа Android наиболее подходящая благодаря своей распространенности, функциональности и доступности для разработчиков. Учитывались также такие особенности Android, как эффективное управление памятью и поддержка многозадачности.

На рынке отсутствуют многофункциональные обучающие приложения, способные эффективно решать проблемы доступности и индивидуализации обучения. Поэтому было принято решение о разработке подобного мобильного приложения

2 Логическое моделирование и проектирование системы

2.1 Формирование требований к системе

При проектировании мобильной обучающей системы, находящейся в разработке, следует учитывать не только её визуальное привлекательное оформление, но и максимальную простоту и понятность для пользователей. Рассмотрим ключевые требования к будущей системе в соответствии с классификацией FURPS+ (Таблица 2.1).

«FURPS — классификация требований к программным системам.

Образована от первых букв слов:

– Functionality — Функциональные требования: свойства, возможности, безопасность. Являются основными, по этим требованиям строятся диаграммы вариантов использования (Use case diagram);

– Usability — Требования к удобству использования (UX): человеческий фактор, эстетика, последовательность, документация;

– Reliability — Требования к надежности: частота возможных сбоев, отказоустойчивость, восстанавливаемость, предсказуемость устойчивости;

– Performance — Требования к производительности: время отклика, использование ресурсов, эффективность, мощность, масштабируемость;

– Supportability — Требования к поддержке: возможность поддержки, ремонтопригодность, гибкость, модифицируемость, модульность, расширяемость, возможность локализации» [19].

Символ «+» в аббревиатуре FURPS включает дополнительные ограничения проектирования:

– «design – ограничения проектирования;

– implementation – ограничения разработки;

– interface – ограничения на интерфейсы;

– physical – физические ограничения. » [10]

Эти требования указывают, что должно быть выполнено в процессе

разработки приложения. В настоящее время различают два типа требований:

– «Функциональные требования - описывают конкретные действия или поведение системы. Они отвечают на вопрос «Что система должна делать?» и прямо связаны с функциональностью. Эти требования обычно определяются через сценарии использования, пользовательские истории или спецификации.

– Нефункциональные требования касаются характеристик системы, таких как производительность, безопасность, надежность и удобство использования. Они определяют, как система должна вести себя, и влияют на общее качество пользовательского опыта» [12].

Требования к разрабатываемой мобильной обучающей системе представлены в таблице 3.

Таблица 3 – Требования к приложению

Требование	Статус	Полезность	Риск	Стабильность
Функциональные требования				
Приложение обязано работать на операционных системах Android версии 7.0 и выше.	Одобренные	Важная	Средний	Средняя
Следует гарантировать надежное соединение с сервером.	Одобренные	Критичное	Средний	Средняя
Приложение должно предоставлять результаты тестирования.	Одобренные	Важная	Средний	Низкая

Продолжение таблицы 3

Требование	Статус	Полезность	Риск	Стабильность
Функциональные требования				
Пользовательский интерфейс приложения должен быть понятным и легким в использовании.	Одобренные	Критичное	Низкий	Низкая
Требования к удобству использования				
Размер шрифта в приложении должен быть не менее 12 пикселей.	Одобренные	Критичное	Низкий	Низкая
Требования к надежности				
Приложение должно продолжать функционировать вплоть до следующего обновления.	Предложенные	Критичное	Низкий	Низкая
Требования к производительности				
Приложение должно отвечать в течение не более 3 секунд.	Предложенные	Важное	Средний	Низкая
Требования к поддержке				
Все выявленные неисправности должны устраняться в течение 24 часов.	Предложенные	Критичное	Средний	Средняя
Ограничения реализации				
Разработка должна вестись в Android Studio с использованием Java.	Предложенные	Критичное	Средний	Средняя

После определения требований к будущей системе необходимо приступить к моделированию мобильной обучающей системы. В этом этапе важно учесть все пользовательские потребности и функциональные особенности, которые были выявлены в процессе анализа.

2.2 Обзор и анализ существующих мобильных обучающих приложений

Следующий шаг - обзор и анализ мобильных обучающих приложений, чтобы определить, какая платформа лучше всего соответствует установленным требованиям. Кратко рассмотрим некоторые из них:

Coursera — ведущая платформа онлайн-образования, предоставляющая доступ к курсам от лучших университетов и компаний. Поддерживает мобильные устройства и офлайн-доступ, что делает обучение удобным в любом месте. Возможность отслеживания успеваемости присутствует, но создание собственных курсов и тестов не предусмотрено. Платформа не поддерживает работу в изолированных сетях.

Moodle — бесплатная и открытая платформа управления обучением, широко используемая во всем мире. Предлагает инструменты для создания курсов и тестов, поддерживает мобильные устройства и русский язык. Moodle также позволяет работать в изолированных сетях и отслеживать успеваемость студентов.

edX — платформа, созданная Гарвардом и MIT, поддерживающая мобильные устройства и офлайн-доступ. Широкий выбор курсов от ведущих образовательных учреждений компенсирует отсутствие возможности создания собственных курсов. Платформа не поддерживает работу в изолированных сетях и русский язык.

Udemy — платформа, где любой может создать и опубликовать свои курсы. Поддерживает мобильные устройства, но бесплатная модель распространения ПО и поддержка русского языка отсутствуют. Платформа не

позволяет работать в изолированных сетях и не предоставляет функции отслеживания успеваемости студентов.

Skillshare — платформа для обучения креативным и бизнес-навыкам, поддерживающая создание курсов и мобильные устройства. Бесплатная модель распространения ПО и русскоязычная поддержка отсутствуют, также нет функций для отслеживания успеваемости и работы в изолированных сетях.

Quizlet — платформа для создания обучающих карточек и тестов, поддерживающая мобильные устройства и офлайн-режим. Русскоязычная поддержка делает её доступной для русскоязычных пользователей. Платформа не позволяет работать в изолированных сетях и не предоставляет функции отслеживания успеваемости.

MasterClass — платформа с курсами от известных экспертов в различных областях. Поддерживает мобильные устройства, но не позволяет создавать собственные курсы и тесты. Нет функций офлайн-доступа, работы в изолированных сетях и русскоязычной поддержки.

Mimo — платформа для изучения программирования и цифровых навыков, поддерживающая мобильные устройства и офлайн-режим. Создание курсов и тестов не предусмотрено, как и поддержка русского языка и работа в изолированных сетях.

Проанализируем представленные мобильные обучающие приложения на основе ранее установленных критериев. Результаты анализа сведены в таблицу 4.

Стоит отметить, что некоторые платформы обеспечивают работу в изолированных сетях и предоставляют возможность офлайн-доступа, что является ключевыми характеристиками для определенных пользователей. Эти особенности делают такие приложения особенно полезными в условиях ограниченного или отсутствующего Интернет-соединения.

Таблица 4 – Сравнение мобильных обучающих систем

Критерии	Мобильные обучающие приложения							
	Coursera	Moodle	edX	Udemy	Skillshare	Quizlet	MasterClass	Mimo
Бесплатная лицензия	+	+	+	-	-	+	-	+
Создание образовательных курсов	-	+	-	+	-	-	-	-
Подготовка тестовых заданий	-	+	-	+	-	+	-	-
Поддержка мобильных платформ	+	+	+	+	+	+	+	+
Локализация на русский язык	+	+	+	-	-	-	-	-
Отслеживание учебной деятельности	+	+	+	+	-	-	-	+
Поддержка оффлайн доступа	+	-	+	+	-	+	-	+
Поддержка работы в изолированных сетях	-	+	-	-	-	-	-	+

Проанализировав таблицу, можно заключить, что большинство мобильных обучающих приложений в значительной степени соответствуют установленным требованиям. Наибольшие ограничения вызывают два критерия – наличие бесплатной модели распространения ПО и поддержка офлайн-доступа. Изучив результаты, можно сделать вывод, что многие из представленных платформ не полностью отвечает всем требованиям.

Учитывая это, принято решение разработать собственное мобильное обучающее приложение, которое будет включать все необходимые функции. Приложение будет работать в изолированных сетях и предоставлять офлайн-доступ, что обеспечит пользователям непрерывное обучение независимо от наличия интернет-соединения.

2.3 Выбор технологии логического моделирования мобильного приложения

«Логическое моделирование представляет собой процедуру проверки функционирования логической схемы с помощью компьютера. Его основная цель состоит в том, чтобы проверить функцию проектируемой логической схемы без ее физической реализации» [6].

«CASE средства (Computer - Aided Software Engineering) – это инструмент, который позволяет автоматизировать процесс разработки информационной системы и программного обеспечения. Разработка и создание информационных систем управления предприятием связаны с выделением бизнес-процессов, их анализом, определением взаимосвязи элементов процессов, оптимизации их инфраструктуры и т.д. Основной целью применения CASE средств является сокращение времени и затрат на разработку информационных систем, и повышение их качества.

Многие современные CASE средства предоставляют возможности для моделирования практически всех предметных областей деятельности организаций. В составе этих средств существуют инструменты для описания моделей бизнес-процессов за счет различных диаграмм, схем, графов и таблиц» [17].

После отбора наиболее подходящей технологии для логического моделирования наступает ключевой этап — создание диаграмм вариантов использования [8] и диаграмм последовательности для мобильной образовательной платформы. Этот процесс не только помогает визуализировать основные сценарии взаимодействия пользователей с системой, но и дает возможность более глубоко понять и проанализировать процесс обучения, а также потребности различных пользовательских категорий.

В ходе анализа разрабатываемой системы мы обращаем внимание на три ключевые роли:

– администратор: обеспечение функциональности системы, управление учетными записями пользователей и эффективное распределение ролей в соответствии с их функциональными обязанностями;

– пользователь: опыт использования, обеспечиваемый возможностью легкого доступа к материалам курсов, удобным прохождением тестов и удобной проверкой результатов;

– преподаватель: создание и управление курсами, а также доступ к результатам тестирования студентов для эффективного обучения и оценки успеваемости.

Цель моделирования состоит в выявлении основных этапов обучения и оптимального взаимодействия пользователей с образовательным приложением. В таблице 5 перечислены ключевые сценарии, которые будут реализованы в приложении.

Таблица 5 – Описание прецедентов

Прецедент	**Краткое описание**
Внесение нового материала	Преподаватель добавляет в курс новый теоретический материал.
Создание тестовых вопросов	Преподаватель разрабатывает и внедряет в курс новые тестовые задания.
Изучение учебного контента	Пользователь знакомится и изучает теоретический материал, представленный преподавателем.
Прохождение тестов	Пользователь выполняет тестовые задания для проверки усвоенных знаний.
Анализ результатов тестирования	Пользователь просматривает свои результаты после выполнения тестов.
Управление учетными записями	Администратор добавляет новые учетные записи пользователей и назначает им соответствующие роли.

На рисунке 4 представлена диаграмма вариантов использования.

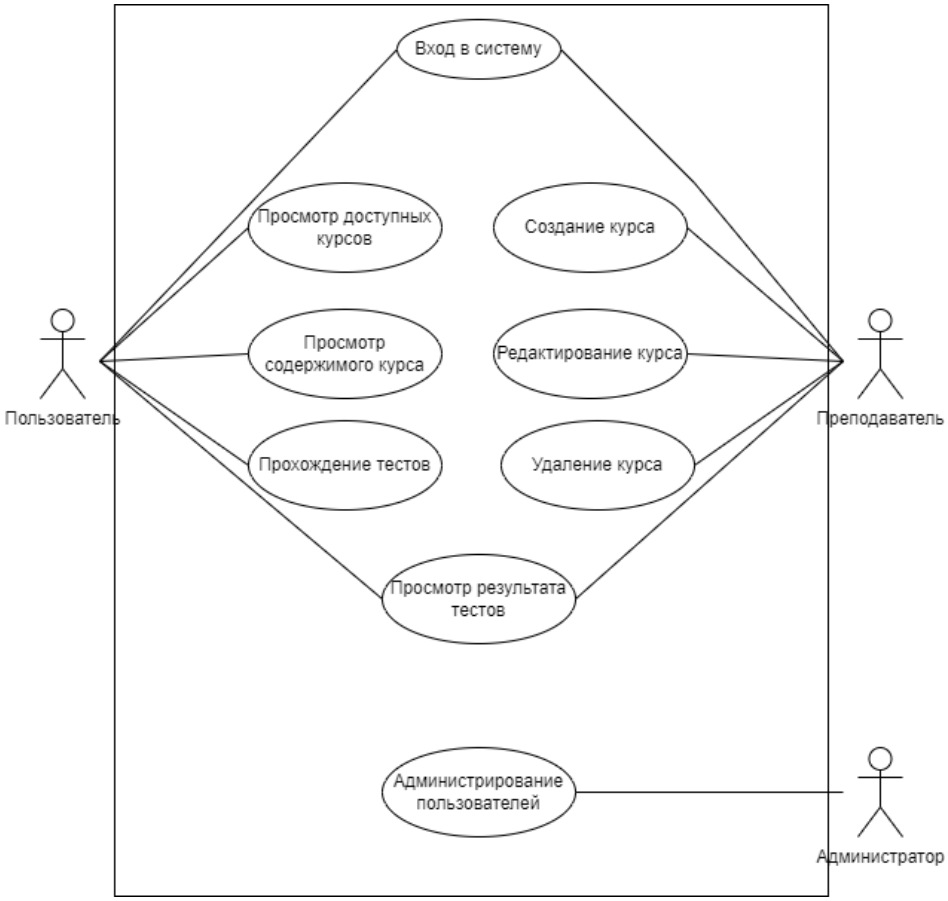


Рисунок 4 – Диаграмма вариантов использования

В системе предусмотрена функция для преподавателей по созданию и наполнению учебных курсов, а также соответствующих тестов к ним. После изучения теоретических материалов пользователи могут приступить к тестированию. Для наглядного представления взаимодействия между компонентами системы мы разработали диаграмму использования.

На диаграмме каждый субъект системы выполняет специфические задачи. Преподаватели занимаются разработкой курсов и включением в них теоретических материалов и тестовых заданий. Администраторы отвечают за создание и администрирование учётных записей пользователей. Пользователи, в свою очередь, выбирают интересующий их раздел для изучения, после чего система предоставляет необходимые материалы. По завершении обучения

пользователи могут пройти тестирование, и система отобразит их результаты по завершению теста.

2.4 Проектирование модели данных

«Чтобы начать проектирование модели данных, необходимо сформировать как концептуальную, так и логическую модели» [1]. «Концептуальная модель базы данных — это наглядная диаграмма, нарисованная в принятых обозначениях и подробно показывающая связь между объектами и их характеристиками» [4]. Для начала дизайна модели данных мы должны уточнить основные сущности, такие как задачи, рассылка уведомлений и выполненные задачи. Концептуальная схема базы данных представлена на рисунке 5.

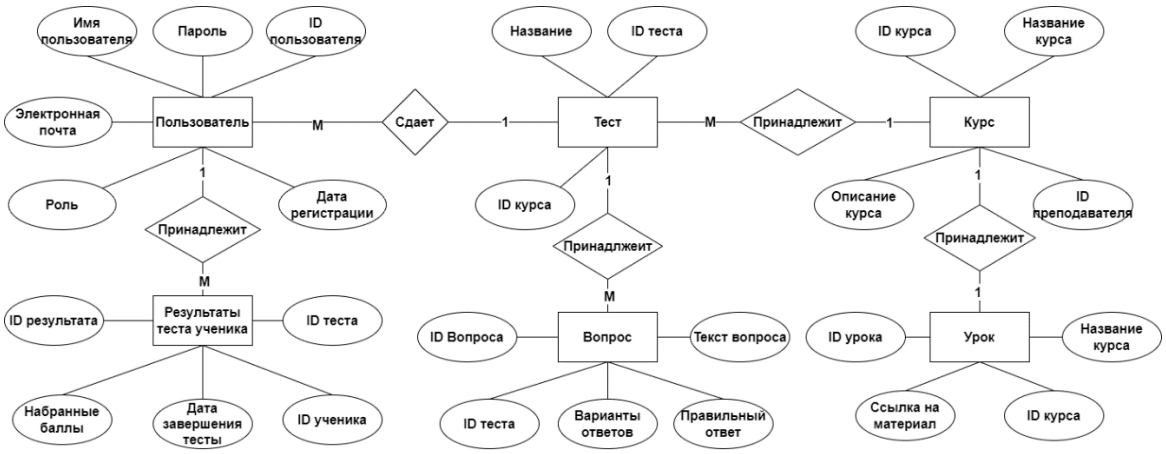


Рисунок 5 – Концептуальная модель данных

На диаграмме каждая сущность обладает своими характеристиками и взаимосвязями с другими сущностями.

– Сущность «Пользователь» включает атрибуты: имя пользователя, пароль, электронная почта, роль и дата регистрации. Связь «Принадлежит» указывает на то, что один пользователь может иметь несколько результатов теста. Атрибуты сущности «Результаты теста ученика» включают ID результата, набранные баллы и дату завершения теста.

– Сущность «Тест» имеет атрибуты: ID теста и ID курса. Связь «Сдает»

показывает, что один пользователь может сдавать несколько тестов, а каждый тест может быть сдан несколькими пользователями.

− Сущность «Курс» включает атрибуты: ID курса, название курса, описание курса и ID преподавателя. Связь «Принадлежит» указывает, что курс может включать в себя несколько тестов;

− Сущность «Вопрос» имеет атрибуты: ID вопроса, текст вопроса, варианты ответов и правильный ответ. Каждый вопрос связан с конкретным тестом через атрибут ID теста;

− Сущность «Урок» включает атрибуты: ID урока, название курса, ссылка на материал и ID курса. Эта сущность показывает, что каждый урок относится к определенному курсу.

2.5 Архитектура программного продукта

«Мобильное приложение будет разработано с учетом архитектуры Architecture Components, которая была анонсирована командой Android на конференции Google I/O в 2017 году. Архитектура приложения будет включать в себя новые библиотеки, которые помогут разработать надежное и тестируемое приложение. Этот набор библиотек поможет решить общие проблемы изменения конфигурации, утечки памяти» [18]. На рисунке 6 представлена схема Architecture Components. «Компоненты архитектуры могут использоваться как в совокупности, так и независимо друг от друга». [2]

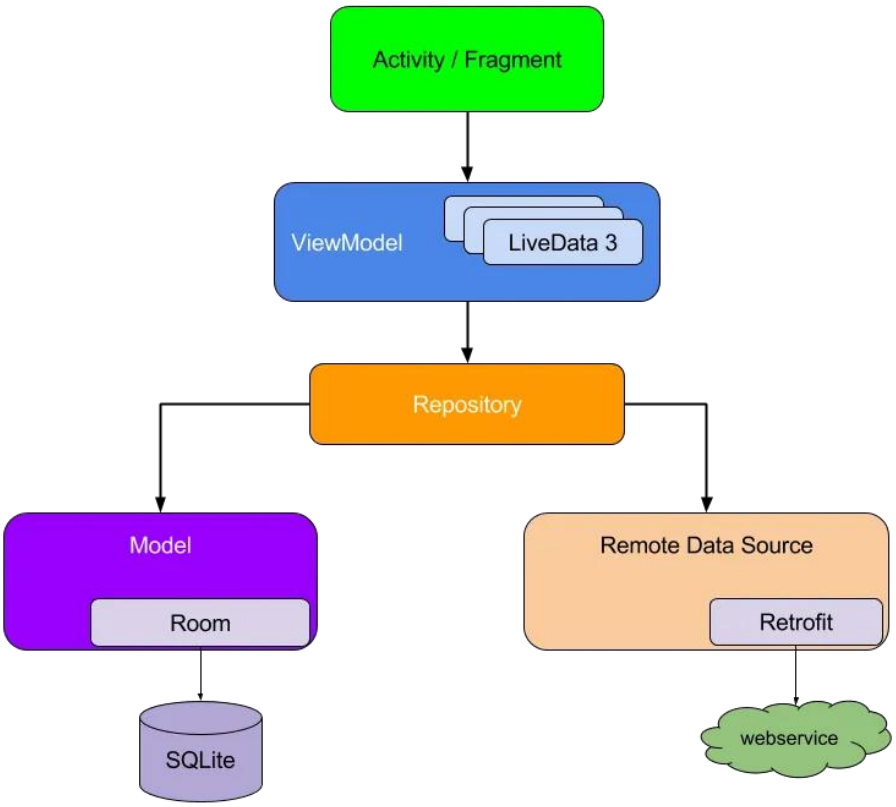


Рисунок 6 – Схема Architecture Components

После выбора архитектурного подхода для мобильного приложения можно визуализировать классы и их взаимодействие, используя структуру классов приложения. Эта структура представляет сложную систему в упрощенной форме. На рисунке 7 представлена структура классов мобильного приложения.

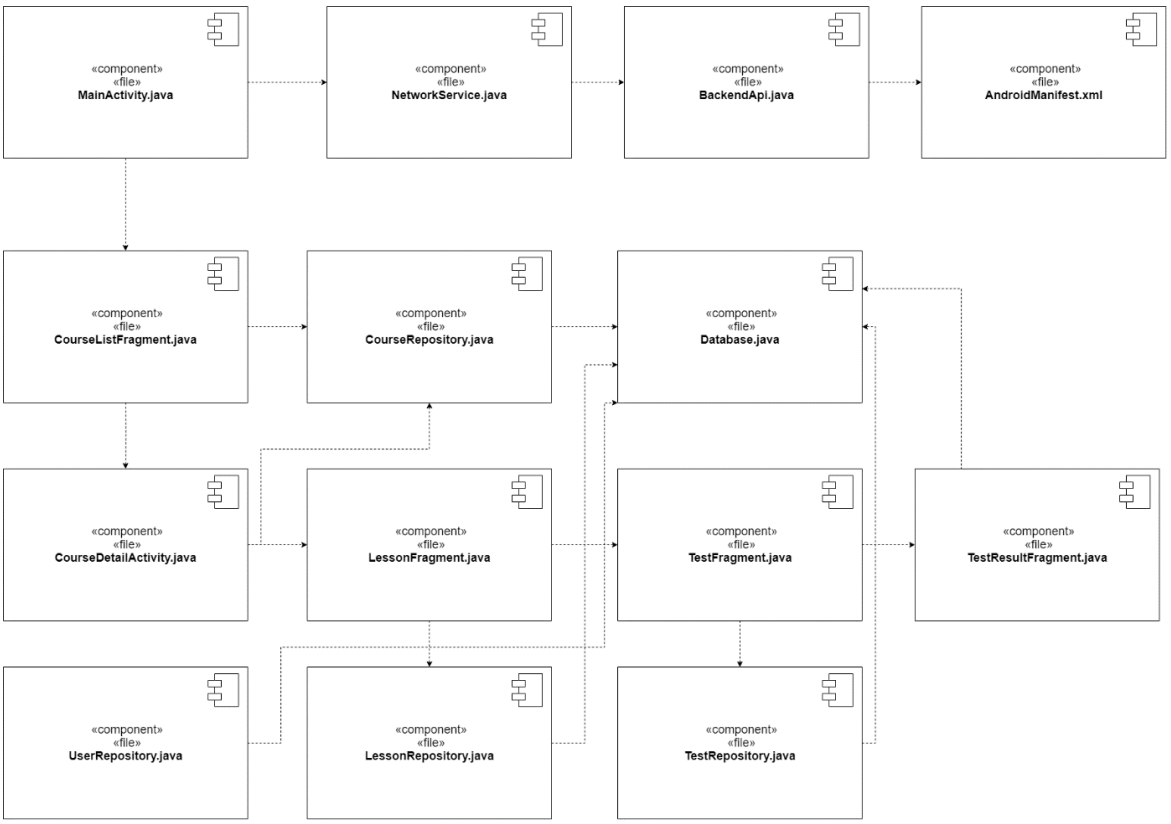


Рисунок 7 – Структура классов мобильного приложения

Ниже, на рисунке 8 продемонстрирована диаграмма развертывания.

В левой части диаграммы находится «docker_host», который является сервером для запуска контейнеров. На нем располагаются две сущности: «service_network», представляющий сеть для связи между сервисами, и «db_volume», являющийся разделом для хранения записей базы данных.

Внутри «service_network» есть два контейнера:

– «CSApp», который содержит артефакт «app.jar». Этот контейнер выполняет роль сервиса, написанного на Java,

– «PostgreSQL DB», который является базой данных [22] PostgreSQL [7].

Справа находится клиентская часть, обозначенная как «Мобильный телефон», которая взаимодействует с приложением через протокол HTTP [23] на порту 80. В мобильном приложении отображаются данные в формате HTML [24].

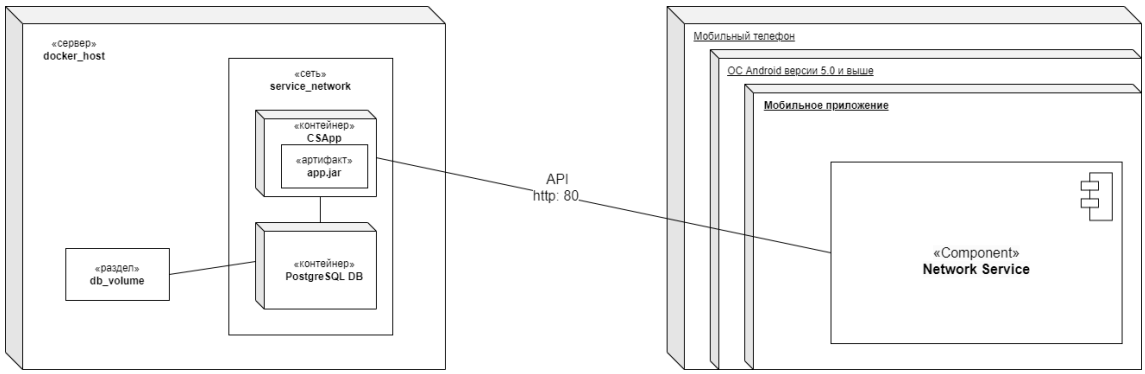


Рисунок 8 – Диаграмма развертывания мобильного приложения и Backend сервиса

С завершением диаграммы развертывания завершается проектирование мобильного приложения, и теперь можно перейти к его реализации и последующему тестированию, а также к разработке backend-сервиса. Важно учитывать все детали, описанные в диаграмме, чтобы избежать возможных проблем на этапе реализации и обеспечить качественное функционирование системы.

Вывод по главе 2:

В данной главе рассмотрены ключевые аспекты проектирования мобильной обучающей системы с использованием модели FURPS+. Были определены основные требования: удобство использования, надежность, производительность и поддерживаемость. Проектирование взаимодействия пользователей с системой осуществлялось с помощью UML и CASE инструментов, что повысило эффективность разработки.

Был проведен обзор существующих мобильных обучающих приложений. Анализ показал, что большинство из них соответствуют базовым требованиям, но не удовлетворяют критериям работы в изолированных сетях и офлайн-доступа. На основе анализа требований была разработана структура данных и архитектура приложения, включая использование компонентов Architecture Components для управления жизненным циклом данных.

Диаграммы вариантов использования и последовательности наглядно продемонстрировали взаимодействие между различными участниками системы

и компонентами архитектуры. Эти диаграммы позволили более точно определить сценарии использования и взаимодействие компонентов.

После тщательного проектирования, включающего определение архитектуры, структуры данных и сценариев взаимодействия, можно приступить к практической реализации системы. Это обеспечит высокое качество обучения и удобство использования для конечных пользователей, отвечая всем установленным требованиям и стандартам.

На этапе реализации следует уделить особое внимание тестированию системы на различных устройствах и платформах, чтобы обеспечить её совместимость и стабильную работу. Для этого будет проводиться тестирование на различных устройствах с различной архитектурой и размером экрана, что позволит выявить и устранить потенциальные проблемы до выхода системы в эксплуатацию.

3 Реализация многофункционального мобильного обучающего приложения

3.1 Выбор технологии разработки приложения

При разработке мобильного приложения для платформы Android было решено использовать язык программирования Java и среду разработки Android Studio. Java выбрана как основной язык из-за его широкой поддержки для Android и богатого набора инструментов разработки. Android Studio, в свою очередь, предоставляет удобную интегрированную среду для разработки, отладки и тестирования приложений для Android.

Android Studio включает в себя SDK. «Android SDK – это одно из средств разработки ПО для Андроида. Включает в себя не только среду программирования, но и специализированный эмулятор.

Представляет собой набор инструментов разработчика Android приложений. Включает в себя компоненты, которые пригодятся при написании иного ПО. Пример – Fastboot или ADB, которые потребуются для прямого взаимодействия с устройством опытным пользователем» [16].

Управление версиями API является ключевым аспектом при разработке под Android. Android SDK предоставляет API для взаимодействия с различными версиями Android OS, что обеспечивает совместимость приложений с разными устройствами и версиями операционной системы.

Для обеспечения оптимального пользовательского опыта необходимо учитывать разнообразие устройств Android с различными размерами экранов и разрешениями. Приложение должно корректно отображаться на всех устройствах, поэтому важно использовать адаптивный дизайн и создавать различные версии макетов экранов.

Для тестирования приложения на различных устройствах и версиях Android OS разработчики могут использовать эмуляторы, предоставляемые Android Studio. Эмуляторы позволяют запускать приложение на виртуальных

устройствах с различными характеристиками, что помогает выявить и устранить проблемы совместимости и интерфейса. На рисунках 9-10 представлена статистика распределения версий ОС Android по состоянию на октябрь 2023 года.

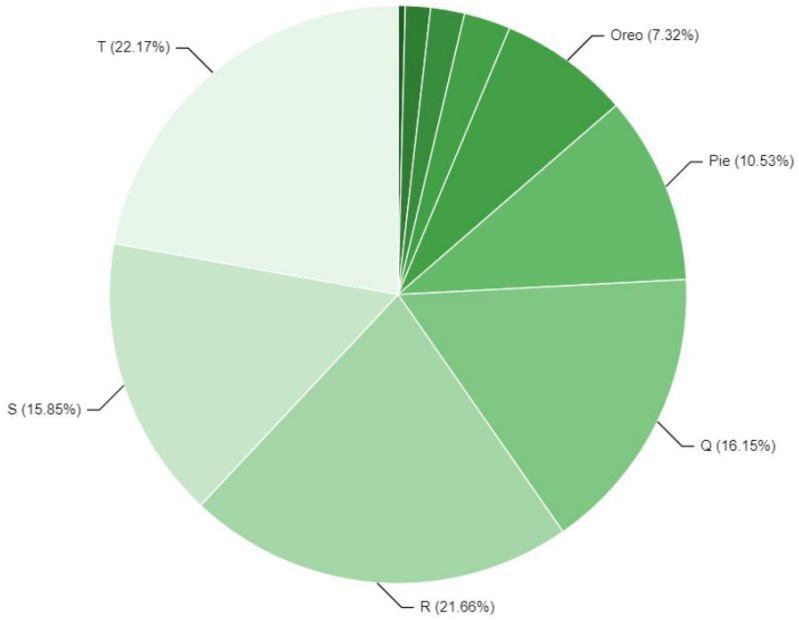


Рисунок 9 – Статистика распределения версий ОС Android по состоянию на октябрь 2023

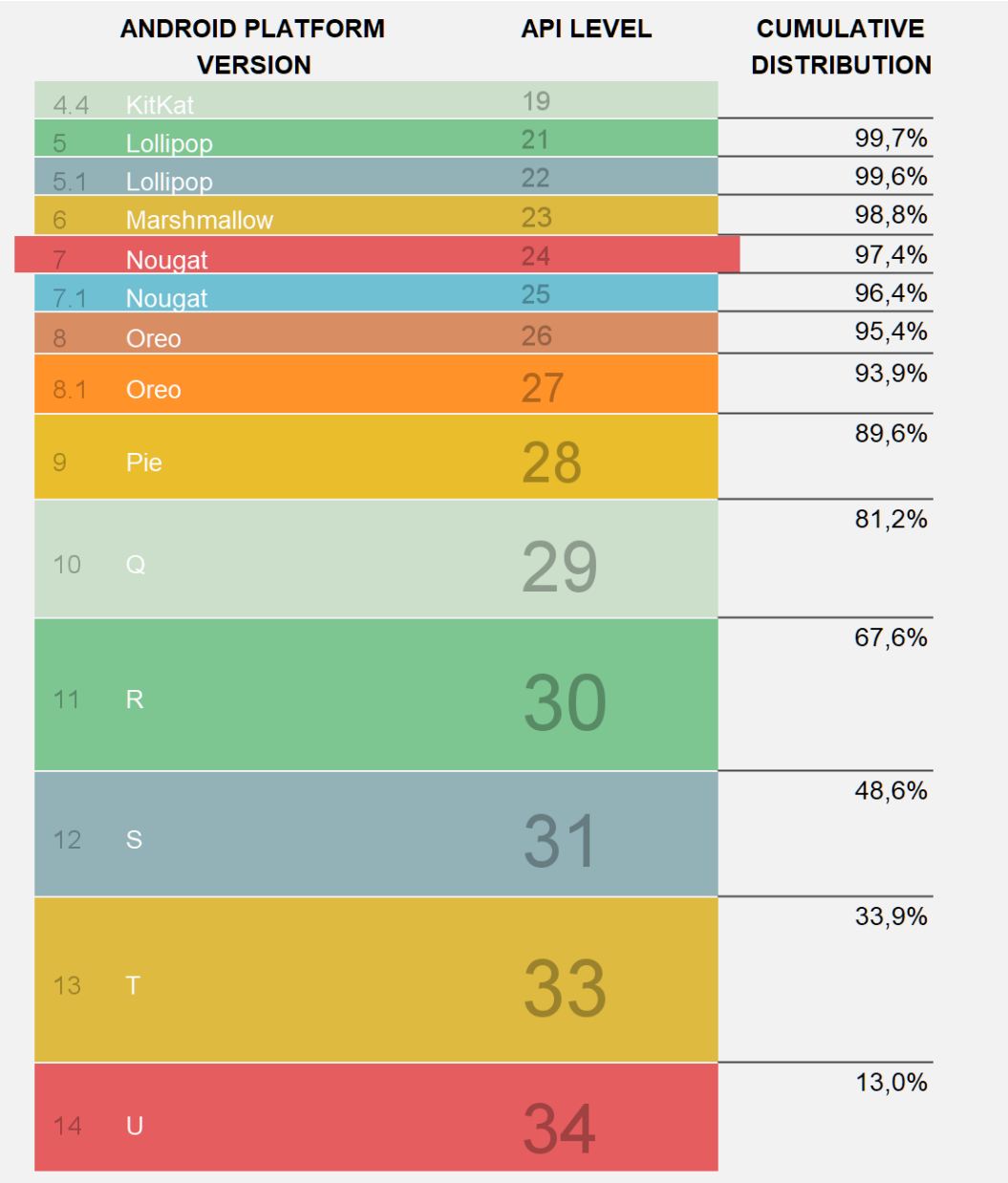


Рисунок 10 – Статистика распределения версий API ОС Android по состоянию на октябрь 2023

При разработке нового проекта в Android Studio можно установить минимально поддерживаемую версию Android, начиная с версии 7.0 (Nougat). Это решение позволяет ориентироваться на более современные устройства, обеспечивая совместимость с широким спектром оборудования. Разрабатываемая мобильная образовательная система будет предназначена для работы на устройствах, использующих Android 7.0 и более новые версии операционной системы. Для обеспечения надежности и функциональности

приложения на всех поддерживаемых платформах, оно будет тщательно тестироваться на различных версиях Android. Это включает в себя проверку производительности, стабильности и корректности работы на устройствах с разными конфигурациями и версиями ОС, начиная с Android 7.0 и выше. Такой подход гарантирует, что система будет функционировать должным образом на максимально возможном числе устройств.

3.2 Разработка программного обеспечения

3.2.1 Создание проекта в Android Studio

Разработка приложений для операционной системы Android включает множество этапов и процессов, одним из которых является создание различных активностей. Активность представляет собой отдельный экран пользовательского интерфейса, и каждая из них разрабатывается с помощью XML-файлов. Эти файлы содержат код, который определяет расположение и внешний вид визуальных элементов на экране, таких как кнопки, текстовые поля и изображения.

Процесс разработки начинается с создания нового проекта в интегрированной среде разработки (IDE) Android Studio. В самом начале необходимо задать название проекта, что является важным шагом, так как имя проекта будет использоваться во многих местах кода. Затем нужно выбрать тип устройства, для которого разрабатывается приложение. Среди возможных устройств можно выбрать смартфоны, планшеты, телевизоры на базе Android TV или умные часы. Выбор устройства влияет на множество аспектов разработки, начиная от размера экрана и заканчивая доступными функциями и особенностями интерфейса.

На рисунке 11 показано окно выбора устройства, где разработчик может определить целевую платформу для своего приложения. Этот шаг является критически важным, так как от выбранного устройства зависят параметры проекта. Например, для смартфонов и планшетов необходимо учитывать

размеры экранов, чтобы интерфейс был удобным и приятным для пользователей. Для телевизоров на базе Android TV нужно учитывать управление с помощью пульта дистанционного управления, а для умных часов – небольшой размер экрана и сенсорное управление.

Выбор устройства на начальном этапе разработки закладывает основу для дальнейших действий и решений. Это определяет, какие технологии и подходы будут использоваться при создании интерфейса и логики приложения. После выбора устройства и создания проекта разработчик переходит к следующему этапу – созданию активностей и разработке интерфейса с использованием XML. Этот процесс требует внимательного планирования и тестирования, чтобы приложение работало корректно и удобно на выбранном типе устройства.

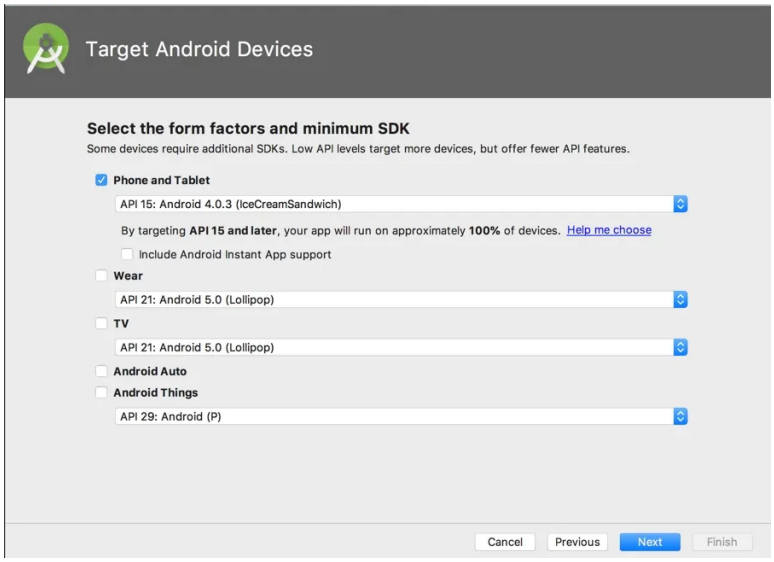


Рисунок 11 – Окно выбора устройств

Поскольку данное приложение разрабатывается для мобильных устройств, важно выбрать соответствующее устройство из списка в процессе настройки проекта. Это ключевой момент, так как выбор правильного устройства обеспечивает оптимальную настройку проекта для заданной платформы. В Android Studio, в процессе создания нового проекта, вам будет предложено выбрать тип устройства, и для данного случая следует выбрать мобильное устройство, чтобы проект был настроен под смартфоны или планшеты.

После выбора мобильного устройства следующим этапом станет выбор шаблона для создания мобильного приложения. Android Studio предоставляет различные шаблоны, которые помогают начать разработку быстрее, предоставляя базовые структуры и компоненты для различных типов приложений. На рисунке 12 показано окно выбора шаблона, где можно увидеть различные варианты, такие как Empty Activity, Basic Activity, Bottom Navigation Activity и другие.

Каждый шаблон предоставляет определенную базовую функциональность и структуру кода, что позволяет разработчику сосредоточиться на добавлении уникальных функций и дизайна, не беспокоясь о начальной настройке. Например, шаблон Empty Activity создаст минимальную структуру проекта с одной активностью, что идеально подходит для простых приложений. В то время как шаблон Basic Activity добавит уже готовую навигационную панель и базовую структуру интерфейса, что может быть полезно для более сложных приложений.

Выбор подходящего шаблона является важным шагом, так как это влияет на начальную структуру вашего проекта и определяет, какие компоненты и библиотеки будут подключены по умолчанию. Это позволяет ускорить процесс разработки и избежать необходимости создавать базовые элементы с нуля.

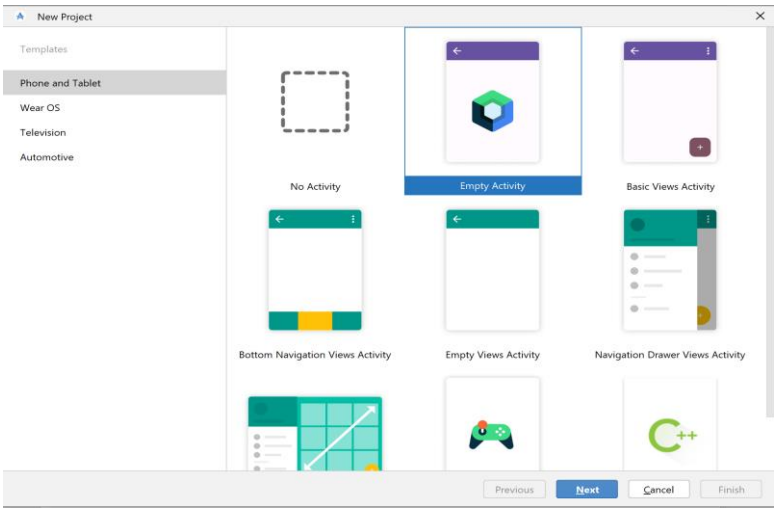


Рисунок 12 – Окно выбора шаблонов

Для старта разработки мобильной обучающей системы с нуля

рекомендуется выбрать шаблон «Empty Views Activity», который предоставляет базовую структуру проекта без предустановленных элементов интерфейса. Это позволит создать приложение с минимальным набором компонентов, исключая излишний функционал и облегчая начальную настройку.

На завершающем этапе создания проекта необходимо указать имя класса, который будет использоваться в качестве главного экрана приложения. Этот шаг критически важен, поскольку от выбора имени класса зависит основной функционал и структура приложения. Рекомендуется выбирать информативное и легко запоминающееся имя, отражающее основную цель и функционал приложения (рисунок 13).

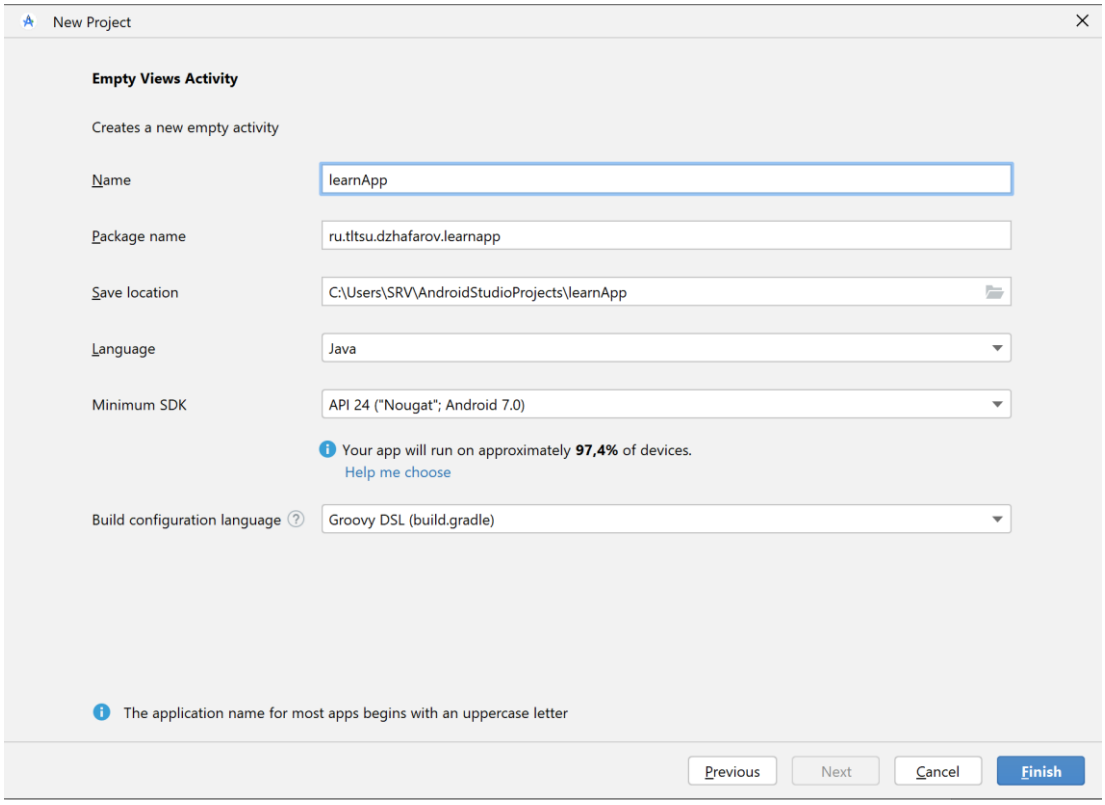
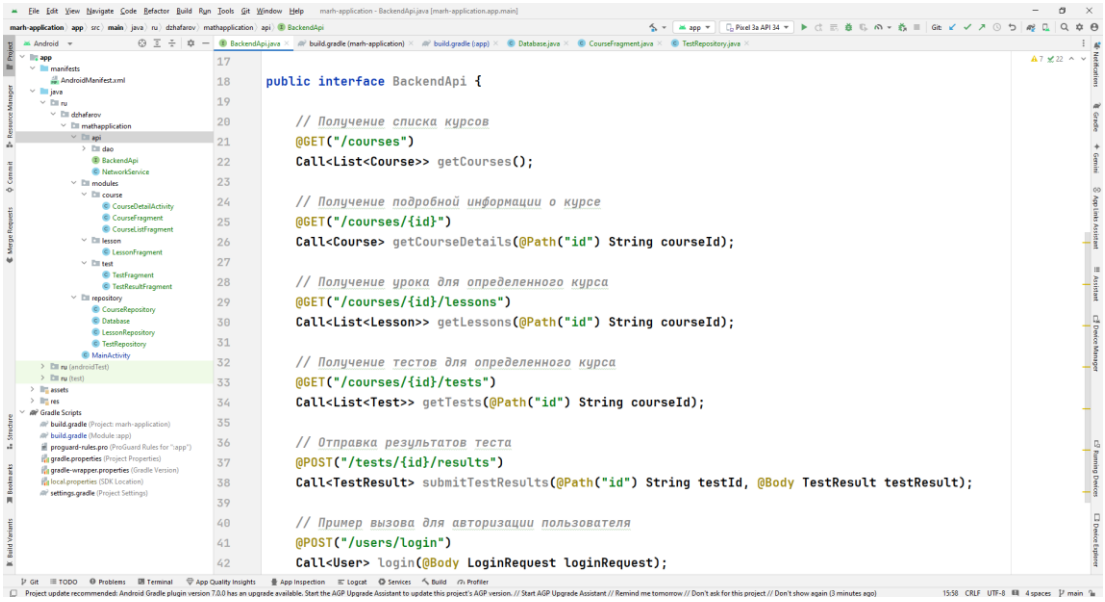


Рисунок 13 – Окно создания шаблона

После создания проекта запускается окно разработки приложения (Рисунок 14).



```
public interface BackendApi {

    // Получение списка курсов
    @GET("/courses")
    Call<List<Course>> getCourses();

    // Получение подробной информации о курсе
    @GET("/courses/{id}")
    Call<Course> getCourseDetails(@Path("id") String courseId);

    // Получение урока для определенного курса
    @GET("/courses/{id}/lessons")
    Call<List<Lesson>> getLessons(@Path("id") String courseId);

    // Получение тестов для определенного курса
    @GET("/courses/{id}/tests")
    Call<List<Test>> getTests(@Path("id") String courseId);

    // Отправка результатов теста
    @POST("/tests/{id}/results")
    Call<TestResult> submitTestResults(@Path("id") String testId, @Body TestResult testResult);

    // Пример вызова для авторизации пользователя
    @POST("/users/login")
    Call<User> login(@Body LoginRequest loginRequest);
```

Рисунок 14 – Окно проекта

В левой части окна проекта отображается структура проекта (рисунок 15).

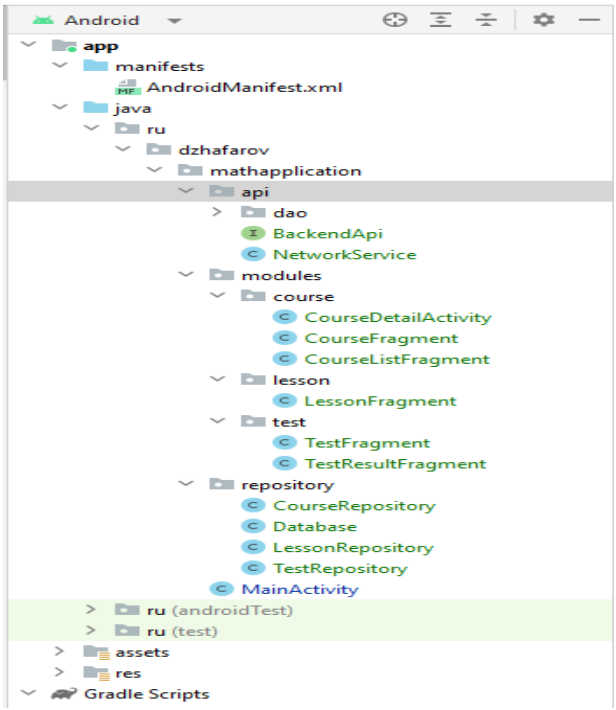


Рисунок 15 – Структура проекта

Android-проект имеет четко организованную структуру, которая включает в себя несколько ключевых директорий и файлов. Основной файл проекта -

AndroidManifest.xml, располагается в папке «manifests». Этот файл содержит информацию о приложении, такую как разрешения, необходимые для его работы, и список компонентов приложения, включая активности, службы, приемники и поставщики контента.

В папке «java» находятся все классы приложения, включая главный класс MainActivity, который является точкой входа в приложение. Кроме того, здесь располагаются другие классы, отвечающие за различные аспекты функциональности приложения, такие как обработка данных, взаимодействие с базой данных и управление пользовательским интерфейсом.

Ресурсы приложения хранятся в папке «res». Эта директория содержит подпапки для различных типов ресурсов, таких как изображения, макеты пользовательского интерфейса, строки, цвета и стили. Например, папка «drawable» содержит изображения, используемые в приложении, а папка «layout» содержит XML-файлы, определяющие структуру экранов приложения.

Для автоматизации сборки проекта используется директория «gradle scripts». Здесь находятся файлы сценариев, которые управляют процессом сборки приложения и управлением зависимостями. Gradle позволяет оптимизировать и упростить процесс разработки, обеспечивая гибкую настройку среды разработки и сборки приложения.

3.2.2 Описание функциональности приложения

При запуске приложения на экран отображается экран авторизации (Рисунок 16), в случае успешной авторизации открывается главный интерфейс приложения (Рисунок 17)

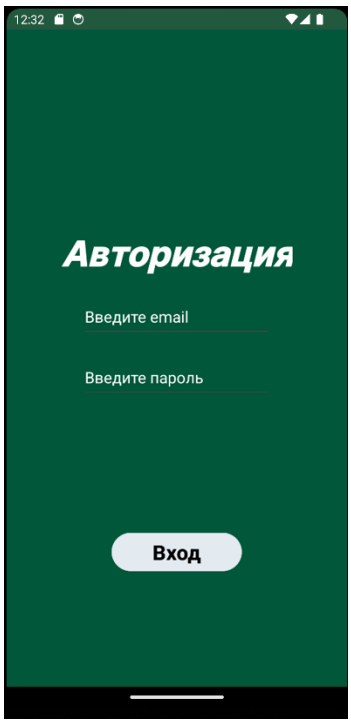


Рисунок 16 – Экран авторизации

Рисунок 17 – Главный интерфейс приложения

Пользовательский интерфейс обеспечивает простой и удобный доступ к функциональности приложения, позволяя пользователям легко ориентироваться.

На рисунке 17 отображен интерфейс мобильного приложения с четырьмя ключевыми кнопками:

- курсы: Нажав на эту кнопку, пользователь переходит к разделу обучающих курсов и материалов;
- задания: При выборе этой кнопки пользователь получает доступ к разделу с упражнениями и заданиями для закрепления знаний;
- тесты: Эта кнопка открывает доступ к разделу с тестами и опросами для проверки уровня знаний;
- настройки: При нажатии на эту кнопку пользователь может настроить параметры приложения и предпочтения.

При активации кнопки «Курсы» появляется дополнительное окно, в котором пользователь может выбрать конкретные темы для изучения (рисунок 18).

Рисунок 18 – Разделы теоретического материала

На рисунке 19 пользователь открывает раздел курсов, где представлен список тем каждого курса для изучения. Каждая тема включает в себя информацию о своем содержании, ключевые концепции, материалы для чтения, а также задания для закрепления знаний. Пользователь может выбрать конкретную тему, которая его интересует, и начать изучение материала. Такой подход обеспечивает пользователям удобный доступ к содержанию курсов и позволяет им гибко управлять своим обучением, выбирая темы в соответствии с их интересами и потребностями.

Рисунок 19 – Перечень тем курса

При нажатии на выбранное название открывается теоретический материал, связанный с ним (Рисунок 20).

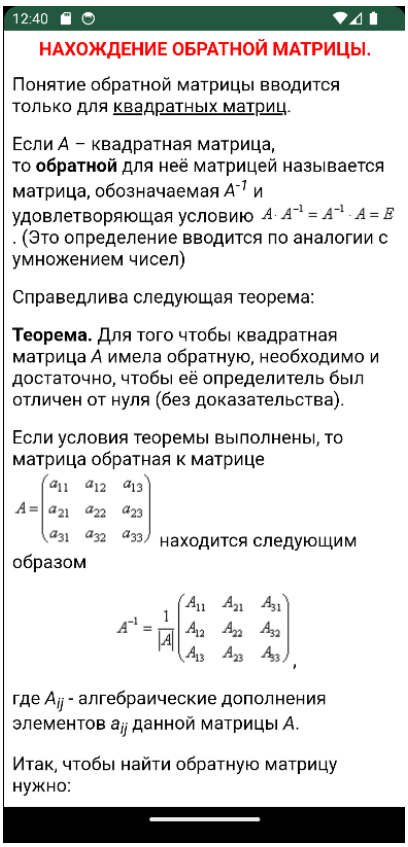


Рисунок 20 – Теоретический материал темы

Теоретический материал представляет из себя набор HTML файлов, которые загружаются с backend сервера.

В Android Studio предусмотрен встроенный инструментарий для управления встроенным веб-браузером, который открывает HTML-страницы, называемый WebView. Этот компонент позволяет отображать веб-контент прямо внутри приложения, обеспечивая интеграцию с веб-ресурсами и интерактивными веб-страницами.

WebView загружает HTML-страницы с backend-сервера, что позволяет использовать все возможности веб-технологий, такие как HTML, CSS и JavaScript, для создания богатых и динамичных интерфейсов. Это особенно полезно, когда необходимо интегрировать в приложение веб-функциональность, не переписывая код на нативном языке.

Поддержка JavaScript в WebView позволяет нативному коду и веб-контенту обмениваться данными и событиями, создавая более гибкие и

функциональные приложения.

После изучения теоретического материала пользователь может пройти тестирование.

Окно тестирования представлено на рисунке 21.

Рисунок 21 – Экран тестирования

Тестовые задания сдержат вопросы по теоретическому разделу.

После прохождения тестирования пользователь может посмотреть результат теста (Рисунок 22).

Рисунок 22 – Результат тестирования

Тем самым было разработано многофункциональное мобильное обучающее приложение.

3.3 Тестирование мобильного приложения

После завершения разработки мобильного приложения важным этапом стало проведение тестирования, направленного на выявление потенциальных ошибок и улучшение общей производительности продукта. В процессе тестирования были предприняты меры по повышению эффективности, включая создание эмуляторов различных смартфонов и планшетов. Эти эмуляторы имитировали устройства с разными диагоналями экрана и работающие на различных версиях операционной системы Android. Такой подход позволил

выявить и устранить возможные проблемы, обеспечив стабильную и качественную работу приложения на широком спектре устройств.

Каждый эмулятор был настроен с учетом конкретных моделей устройств, которые пользователь может использовать. Были использовали такие эмуляторы, как Android Virtual Device (AVD) и Genymotion, чтобы точно воспроизвести работу приложения на разных типах устройств. Настройка каждого эмулятора включала в себя подбор разрешений экрана, версий ОС Android и других параметров, чтобы максимально точно эмулировать разнообразие устройств, которые могут использоваться пользователями.

После успешного завершения тестирования на эмуляторах программный продукт перешел к следующему этапу проверки - тестированию на реальных устройствах с различными версиями операционной системы Android. Это позволило более точно оценить работу приложения в реальных условиях эксплуатации и выявить любые несовместимости или проблемы, которые могли бы остаться незамеченными на эмуляторах.

На рисунке 23 изображен раздел теоретического материала приложения, отображенный на планшете с диагональю экрана 10,95 дюйма и разрешением 2560x1600 пикселей в горизонтальной ориентации. Это демонстрирует, как приложение адаптируется к различным устройствам и обеспечивает оптимальное визуальное восприятие для пользователей с разными типами устройств и экранами.

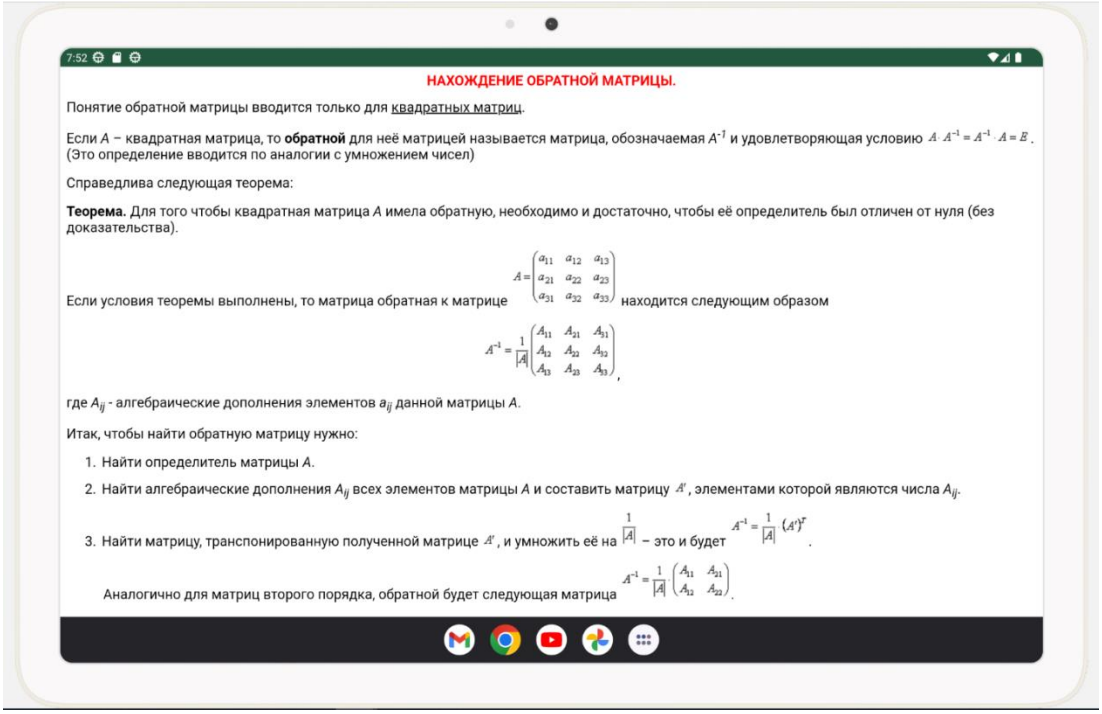


Рисунок 23 – Экран теоретического материала на планшете

Мобильная обучающая система характеризуется всеми ключевыми качествами, присущими хорошему программному обеспечению:

- высокая эффективность,
- удобная гибкость,
- возможность расширения,
- повторное использование компонентов.

Эффективность системы отражает её надежность и производительность. Гибкость позволяет легко и оперативно вносить изменения в программное обеспечение. Возможность расширения предоставляет возможность добавлять новые функции и элементы. Повторное использование отдельных частей программы делает её более универсальной и экономит время при разработке новых проектов.

Также проводилось бета-тестирование [5] с участием реальных пользователей, которое показало, что приложение функционирует без ошибок и полностью соответствует заявленным требованиям.

Вывод по 3 главе:

В процессе работы над третьей главой были тщательно выбраны и обоснованы инструменты и технологии для разработки мобильной обучающей системы. После выбора средств разработки была проведена реализация системы с детальным описанием ключевых функций и возможностей приложения. Особое внимание уделялось обеспечению интуитивного и удобного интерфейса, а также интеграции всех необходимых обучающих инструментов.

Система прошла обширное тестирование на различных устройствах, чтобы гарантировать её надежность и совместимость. Для этого были использованы реальные устройства, такие как смартфон POCO X3 Pro, а также эмуляторы Android Virtual Device (AVD), имитирующие работу на смартфоне Pixel 8 Pro и планшете Pixel Tablet. Эмуляторы позволили протестировать работу приложения на различных экранах и версиях операционной системы Android.

Все тестируемые устройства, как реальные, так и виртуальные, успешно запустили приложение, продемонстрировав его корректное функционирование. Результаты тестирования подтвердили, что система обладает высокой степенью совместимости и стабильности, обеспечивая пользователям надежную работу и качественное обучение на различных платформах и устройствах.

Заключение

В современном обществе мобильные приложения становятся неотъемлемой частью нашей повседневной жизни, и особенно важны они в области образования и саморазвития. Предоставляя доступ к образовательным ресурсам в любое время и в любом месте, они играют ключевую роль в повышении образовательной доступности и гибкости.

В рамках данной бакалаврской работы было разработано мобильное обучающее приложение, предназначенное для работы в оффлайн режиме и в изолированных сетях. Это приложение соответствует современным требованиям к интерактивности и эффективности образовательных процессов, предоставляя пользователям возможность управлять своим учебным временем, планировать задачи и отслеживать их выполнение на мобильных устройствах. Благодаря этому, пользователи могут продолжать свое обучение даже в условиях отсутствия интернет-соединения, что особенно важно в удаленных или ограниченных ресурсами районах.

Для достижения поставленных целей были выполнены следующие задачи.

1. Анализ существующих образовательных приложений: Проведен анализ функциональности и интерфейсов существующих приложений для определения ключевых требований к разрабатываемому продукту.

2. Проектирование архитектуры и интерфейса приложения: Разработана архитектура и интерфейс приложения с учетом возможности работы в оффлайн режиме. Были выбраны наиболее подходящие технологические решения и платформы для разработки.

3. Реализация функциональности приложения: Разработаны модули для создания и управления учебными задачами, а также другие необходимые функции приложения.

4. Тестирование приложения: Проведено тестирование приложения для проверки его функциональности, удобства использования и соответствия всем поставленным требованиям.

Разработка мобильного обучающего приложения стала значимым этапом в сфере образования и технологий. Созданный продукт представляет собой инструмент, способствующий улучшению доступности образования и гибкости обучения. Сочетание оффлайн возможностей с современными технологиями обеспечивает пользователям комфорт и удобство в обучении в любых условиях.

Продукт прошел через все этапы разработки с тщательным анализом требований, проектированием, реализацией и тестированием. Результаты тестирования подтвердили не только его функциональность, но и соответствие высоким стандартам качества. В дальнейшем планируется расширение функционала приложения и адаптация его под новые образовательные стандарты и потребности пользователей.

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. Исаев Г. Проектирование информационных систем / Г. Исаев. – Москва: Омега-Л, 2012. – 431 с.
3. Как пользоваться Android Studio. [Электронный ресурс]. URL: https://practicum.yandex.ru/blog/kak-polzovatsya-android-studio/ (дата обращения 21.04.2024).
4. Концептуальная модель базы данных — диаграмма связи между объектами [Электронный ресурс]. URL: https://webonto.ru/kontseptualnaya-model-bazyi-dannyih/ (дата обращения: 10.04.2024).
5. Куликов С. С. Тестирование программного обеспечения. Базовый курс / С. С. Куликов. – 3-е изд. - Минск: Четыре четверти. 2020. 312 с.
6. Моделирование, тестирование и диагностика цифровых устройств [Электронный ресурс]. URL: https://intuit.ru/studies/courses/3440/682/lecture/14036 (дата обращения 20.04.2024).
7. 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.
8. Нотации моделирования бизнес-процессов [Электронный ресурс]. URL: https://leanvector.ru/blog-eksperta/notatsii-modelirovaniya-biznes-protsessov/ (дата обращения: 12.04.2024).
9. Обзор iOS [Электронный ресурс]. URL: https://blog.skillfactory.ru/glossary/ios/ (дата обращения: 10.04.2024).
10. Требования к системе: классификация FURPS+ [Электронный ресурс]. URL: https://sysana.wordpress.com/2010/09/16/furps/ (дата обращения 14.04.2024).

11. Троцкий Д. В., Городецкий В. И. Сценарная модель знаний и язык описания процессов для оценки и прогнозирования ситуаций. Труды Санкт-Петербургского института информатики и автоматизации РАН. 2009. Выпуск 8. С. 94—127.

12. Чем отличаются функциональные и нефункциональные требования? [Электронный ресурс]. URL: https://dzen.ru/a/ZXwTmuUQoQmOACih (дата обращения 14.04.2024).

13. Что такое React Native? Комплексное руководство 2021 [Электронный ресурс]. URL: https://habr.com/ru/articles/596183/ (дата обращения 21.04.2024).

14. Что такое Xamarin? [Электронный ресурс]. URL: https://learn.microsoft.com/ru-ru/previous-versions/xamarin/get-started/what-is-xamarin (дата обращения 19.04.2024).

15. Экономическое обоснование эффективности дипломного проекта [Электронный ресурс]. URL: https://pandia.ru/text/80/171/56140.php (дата обращения 21.04.2024).

16. Android SDK: особенности и характеристики [Электронный ресурс]. URL: https://otus.ru/journal/android-sdk-osobennosti-i-harakteristiki/ (дата обращения 16.04.2024).

17. CASE средства [Электронный ресурс]. URL: https://www.kpms.ru/Automatization/CASE_tools.htm (дата обращения 15.04.2024).

18. Flutter [Электронный ресурс]. URL: https://ru.wikipedia.org/wiki/Flutter (дата обращения 21.04.2024).

19. FURPS. [Электронный ресурс]. URL: https://ru.wikipedia.org/wiki/FURPS/ (дата обращения 14.04.2024).

20. Guide to app architecture. [Электронный ресурс]. URL: https://developer.android.com/topic/architecture (дата обращения 21.04.2024).

21. OneSignal SDK [Электронный ресурс]. URL: https://documentation.onesignal.com/docs/android-sdk-setup (дата обращения 10.05.2024).

22. SQL-Academy Курс по SQL [Электронный ресурс]. URL: https://sql-academy.org/ru (дата обращения: 21.04.2024).

23. Wil van der Aalst, Process Mining Data Science in Action Second Edition. M. : Abstract, 2016. 165 с

24. Wil van der Aalst, Process Mining Discovery, Conformance and Enhancement of Business Processes. M. : Abstract, 2014. 25 с.

Приложение А

Ссылка на репозиторий с исходным кодом

https://gitlab.com/DavidRezcov/ms-ets