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

Институт Математики, физики и информационных технологий
(наименование института полностью)

Кафедра «Прикладная математика и информатика»
(наименование)

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

WEB-дизайн и мультимедиа
(направленность (профиль)/специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему «Разработка мобильного приложения для обучения студентов основам информатики»

Студент А. А. Мавлоназаров
(И.О. Фамилия) (личная подпись)

Руководитель д.ф.-м.н., доцент А. И. Сафронов
(ученая степень, звание, И.О. Фамилия)

Консультант к.ф.н., М. М. Бажутина
(ученая степень, звание, И.О. Фамилия)

Тольятти 2022

Аннотация

Тема бакалаврской работы – «Разработка мобильного приложения для обучения студентов основам информатики».

Объектом бакалаврской работы является процесс обучения студентов основами информатики.

Предметом бакалаврской работы является этап разработки и тестирования мобильного приложения.

Актуальность бакалаврской работы заключается в необходимости разработки мобильного приложения для обучения студентов основам информатики.

Целью бакалаврской работы является разработка мобильного приложения для обучения студентов основам информатики.

В первой главе приводятся основные понятия разрабатываемого мобильного приложения, также обзор похожих мобильных приложений и их сравнение.

Во второй главе проходит процесс проектирования разрабатываемого мобильного приложения, формируются требования к нему, создаются диаграммы вариантов, последовательности, диаграммы классов мобильного приложения, а также логическую и физическую модель данных.

В третьей главе будет происходить разработка мобильного приложения, создание графического интерфейса, показ его работы и процесс тестирования.

Бакалаврская работа состоит из пояснительной записки на 65 страниц, введения в две страницы, 45 рисунков, 6 таблиц и список из 25 источников.

Abstract

The title of the bachelor's thesis is «Development of a mobile application for teaching students the basics of computer science».

The relevance of the work lies in the need to develop a mobile application for teaching students the basics of computer science.

The object of the research is the process of teaching students the basics of computer science.

The subject of the research is the stage of development and testing of a mobile application.

The aim of the study is to develop a mobile application for teaching students the basics of computer science.

In the first part provides the basic concepts of the mobile application being developed, as well as an overview of similar mobile applications and their comparison.

In the second part, the design process of the mobile application being developed is underway, requirements for it are formed, variant diagrams, sequences, class diagrams of the mobile application are created, as well as a logical and physical data model.

In the next part, the development of the mobile application, the creation of a graphical interface, showing its work and the testing process.

The bachelor's thesis consists of 65 pages, a two-page introduction, 48 figures, 6 tables and a list of 25 references.

Содержание

Введение

В настоящее время область мобильных приложений развивается в большую сторону. На сегодняшний день практически жизнь каждого человека тесно связана с мобильными устройствами. Первоначально мобильные устройства имели ограниченное количество функций, но с высоким спросом и развитием технологий в мире современные мобильные устройства стали предоставлять людям гораздо больший функционал. Основанием написания дипломной работы на данную тему послужило стремительное развитие области разработок мобильных приложений.

Современные мобильные устройства работают на операционных системах, таких, как: Android, iOS, Windows 10 Mobile и другие. Мобильное приложение, разрабатываемое для них, представляет собой программное обеспечение, работающее на разных устройствах: смартфонах, планшетах, смарт-часах и прочих мобильных устройствах.

На сегодняшний день мобильными устройствами активно пользуются школьники и студенты в учебных целях, то есть в целях получения информации для изучения различного рода материалов, например, просмотр видеоуроков, прослушивания аудиокниг и т.д.

Актуальность бакалаврской работы заключается в необходимости разработки мобильного приложения для обучения студентов основам информатики.

Объектом бакалаврской работы является процесс обучения студентов основами информатики.

Предметом бакалаврской работы является этап разработки и тестирования мобильного приложения.

Целью бакалаврской работы является разработка мобильного приложения для обучения студентов основам информатики.

Для достижения данной цели необходимо выполнить следующие задачи:

– изучить предметную область по созданию мобильного приложения;

– выбрать методы разработки мобильного приложения;

– разработать логическую модель мобильного приложения;

– реализовать мобильное приложение;

– провести тестирование мобильного приложения.

Бакалаврская работа состоит из 3 разделов.

В первом разделе приводятся основные понятия разрабатываемого мобильного приложения, также обзор похожих мобильных приложений. Кроме того, формируются требования к разрабатываемому мобильному приложению.

Во второй главе приводится обзор средств разработки мобильного приложения, создание диаграммы вариантов использования и классов мобильного приложения, а также логическую и физическую модель данных.

В третьей главе будет происходить разработка мобильного приложения, создание графического, показ его работы и процесс тестирования.

Бакалаврская работа состоит из пояснительной записки на 64 страниц, введения в две страницы, 45 рисунков, 6 таблиц и список из 20 источников.

1 Современные подходы к разработке мобильных приложений для обучения основ информатики

1.1 Постановка задачи

Необходимо разработать мобильное приложение для обучения студентов основам информатики. Далее определим основные требования для нашего приложения, т.к. это упростит разработку программного продукта, поскольку мы будем знать требования для разрабатываемой системы. Следуя этим требованиям, можно избежать всякого рода ошибок при разработке мобильного приложения.

Существует метод классификации требований к создаваемому программному продукту, который называется «FURPS». Эти требования были разработаны компанией Hewlett-Packard (HP).

Согласно этому методу существуют ряд требований, которые перечислены ниже:

- Functionality – функциональные требования;
- Usability – требования к удобству пользования;
- Reliability – требования к надежности;
- Performance – требования к производительности;
- Supportability – требования к поддержке [8].

Ниже будут перечислены требования для каждого раздела.

Functionality – Функциональные требования:

- просмотр контента;
- прохождение теста;
- просмотр результата.

Usability – Требования к удобству пользования:

- удобство пользования графическим интерфейсом приложения;
- простота управления контентом приложения.

Reliability – Требования к надежности:

- бесперебойность работы;
- отказоустойчивость.

Performance – Требования к производительности:

- скорость выполнения требований пользователя;
- малое количество используемых ресурсов устройства.

Supportability – Требования к поддержке:

- простая установка приложения;
- совместимость версии.

Таким образом, были сформированы все необходимые требования к разрабатываемому мобильному приложению, которые описаны по методологии «FURPS»

1.2 Основное назначение мобильных технологий

Android — это "революция открытого исходного кода" в области сотовой телефонии и мобильных вычислений [12]. В наше время сложно уже отнести мобильные приложения к каким-либо направлениям, так как, сейчас уже нет сферы, где бы не пригодился мобильный помощник. Мобильные приложения прочно вошли в жизнь каждого человека. Общение в социальных сетях, заказ такси, прослушивание музыки, чтение книг – всё это доступно с мобильного устройства. Успешным приложением для каждого человека будет тот, который нужен был именно ему. Например, если студент любит читать книги, но не хочет с собой носить в бумажном виде, то в современном мире можно за пару «кликов» на мобильном устройстве установить приложение с удобным и понятным интерфейсом, где можно найти нужную для него книгу и читать, либо слушать аудиокнигу в режиме офлайн. Стремительный рост продаж устройств на базе Android открывает выдающиеся возможности перед разработчиками приложений Android [11].

1.3 Современное состояние рынка мобильных приложений

В настоящее время, сфера разработки мобильных приложений растет в геометрической прогрессии из-за чего она становится в самостоятельный рынок, которая ранее была узконаправленной сферой. Данная отрасль расширяется с каждым днем, где увеличивается количество разработчиков и соответственно, увеличивается количество создаваемых мобильных приложений в геометрической прогрессии.

Самыми популярными платформами на которой создаются мобильные приложения, являются два гиганта мобильной разработки - Android и iOS. Они доминируют с большим отрывом на глобальном рынке мобильных устройств. У таких систем существуют самые популярные магазины приложений, такие, как «App Store» продемонстрированный на рисунке 1 у операционной системы iOS и «Google Play» продемонстрированный на рисунке 2 у операционной системы Android, соответственно.

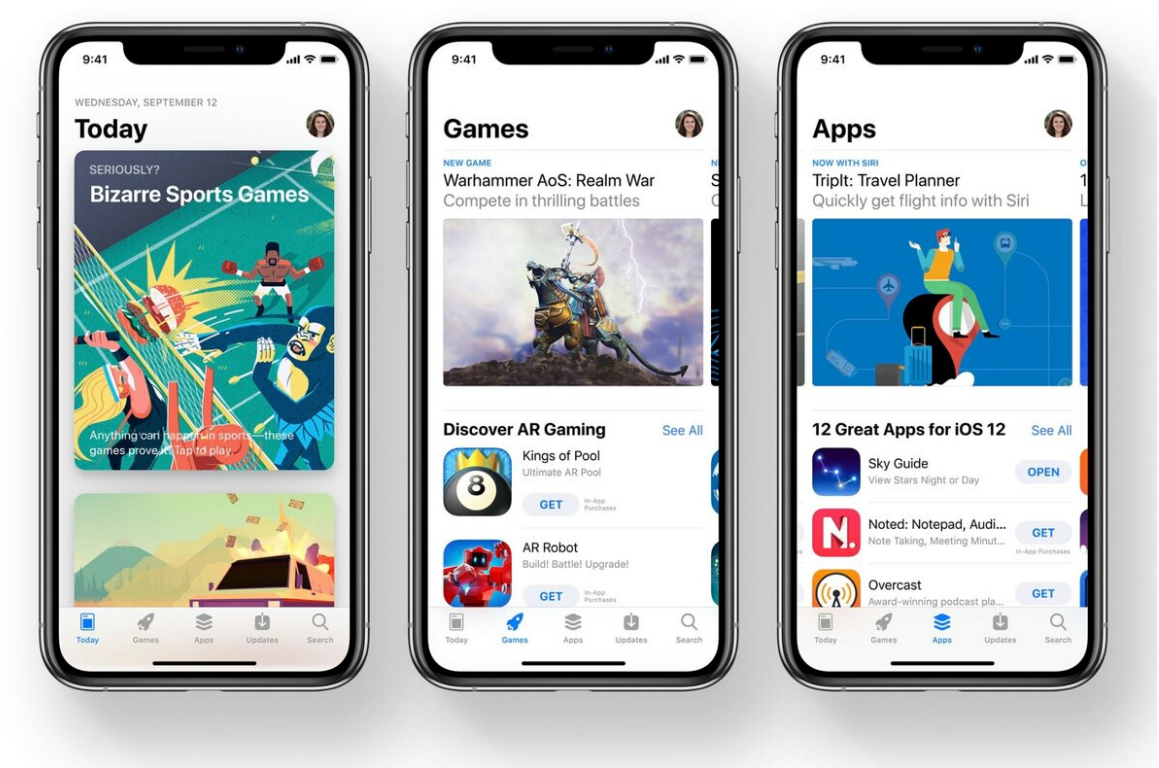

Рисунок 1 – Магазин приложений «App Store»

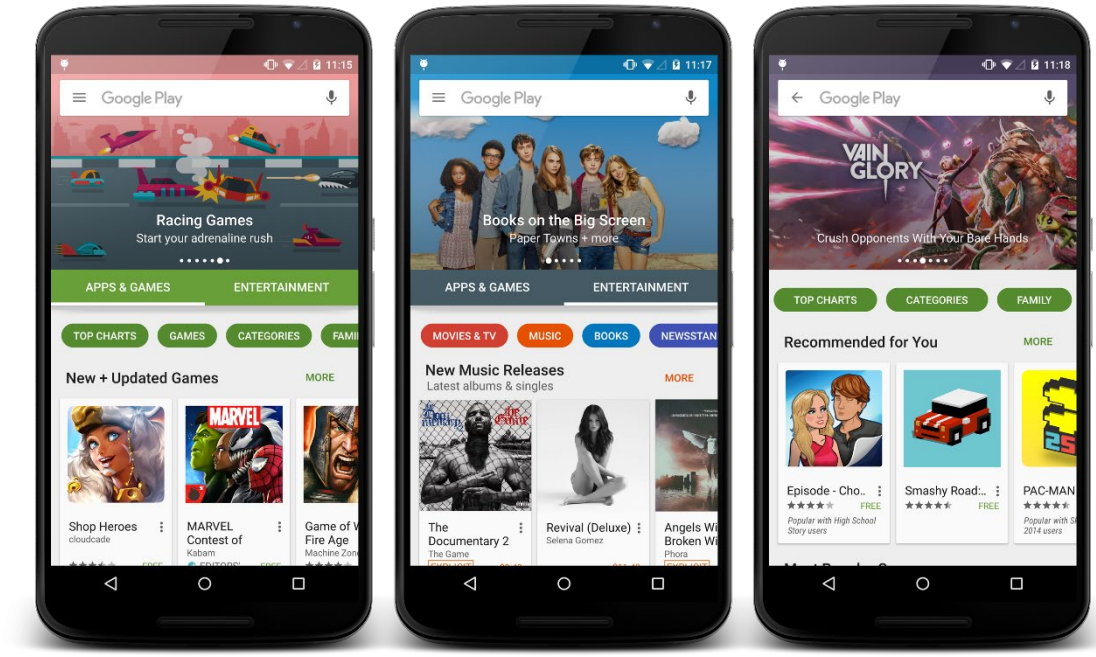

Рисунок 2 – Магазин приложений «Google Play»

С развитием технологий по производству высокопроизводительных процессоров, которые включают в себя мощные графические видеокарты, а также качественные дисплеи и быстрый интернет превратили мобильные устройства в игровые устройства. Также, согласно исследованиям компании «eMarketer», с каждым днем люди пользуются всё чаще мобильными устройствами. Так, почти 70% времени люди проводят в мобильных приложениях, график продемонстрирован на рисунке 3.

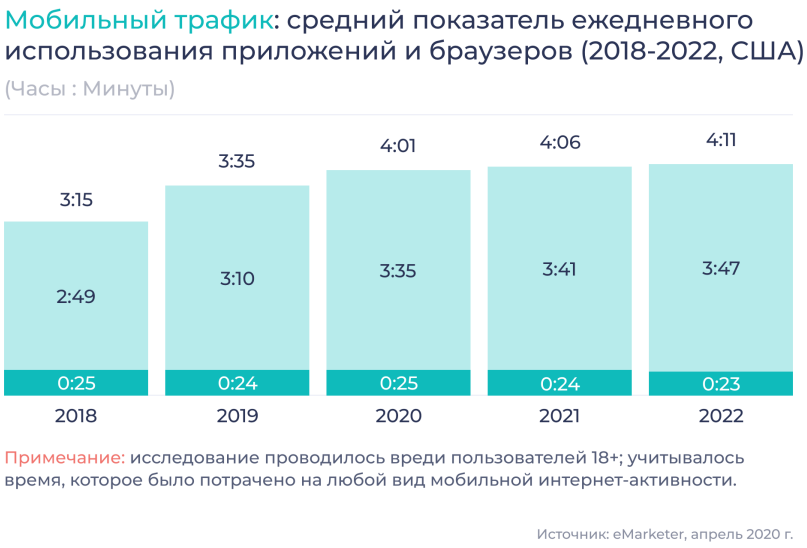

Рисунок 3 – Использование приложений

По данным исследования компании «Statista» можно узнать, что в среднем в магазин приложений «Google Play» загружаются около 10000 новых приложений, а в магазин приложений «App Store» более 30000 новых приложений, продемонстрированный на рисунке 4.

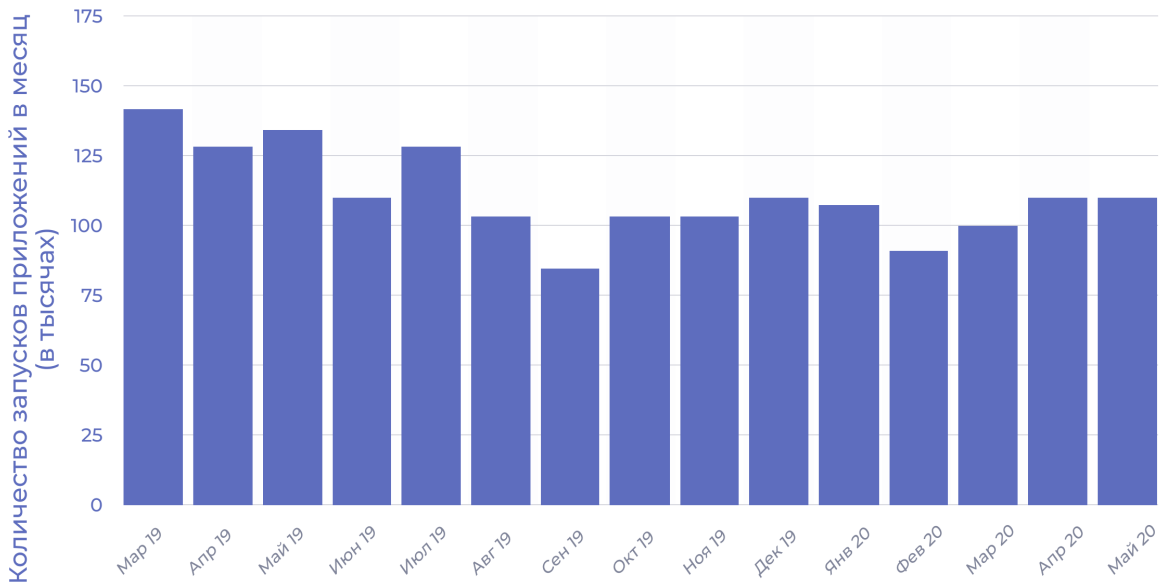

Рисунок 4 – Количество загрузок новых приложений

Можно сделать вывод, что мобильные технологии – это востребованная технология абсолютно в любой сфере.

1.4 Анализ видов мобильных приложений для обучения основ информатики

Современные технологии в образовании используются всё чаще. Особенно после событий 2020 года, где каждый школьник и студент был вынужден учиться на дистанционной основе. Благодаря этому, образование в каком-то смысле перешло в цифровую среду.

Существуют большое количество разнообразных мобильных приложений для обучения. Каждые из которых можно разделить на несколько типов:

– приложения, работающие в режиме онлайн;
– приложения, работающие в режиме офлайн.

Приложения, которые работают в режиме онлайн – это такие приложения, для которых нужен активное подключение интернета, так как вся необходимая информация для правильной работы приложения может находиться на серверной части приложения. Такие приложения можно отнести к архитектуры клиент-серверного. У подобной архитектуры приложения есть свои плюсы, в виде предоставления свежей информации при запуске. Но, также имеется и явный недостаток такого решения в виде потребления трафика интернета при подключении, что в свою очередь может повлиять на скорость запуска приложения.

Приложения, работающие без активного подключения к интернету, обладают большим преимуществом в виде отсутствия потребления интернет-трафика, а также отсутствия внедрения вредоносного кода при подключении к серверу приложения.

Если зайти в магазин приложений «Google Play» и зайти в категорию «образование», то можно обнаружить, что насчитываются тысячи мобильных приложений. Они предназначены для изучения различного рода учебного материала, которые имеют популярности среди обучающихся. Причина такой популярности заключается в простоте использования и свободе выбора огромного количества доступных знаний, которые открывают для школьников и студентов неограниченные возможности в обучении посредством мобильных приложений.

Существуют два вида образовательных приложений, где одни предназначены для изучения конкретного предмета, в нашем случае предмет «Информатика» и другие, которые выполняют роль платформы для обучения. Например, платформой для обучения студентов Тольяттинского

государственного университета является система «Росдистант». Главная страница платформы продемонстрировано на рисунке 5.

Рисунок 5 – Образовательная платформа «Росдистант»

Это web-приложение, которое выполняет роль образовательной среды, где каждый студент может получить различного рода учебные материалы, также закрепить полученные знания, выполняя задания или прохождения теста прямо в образовательной платформе, которое представлено на рисунке 6.

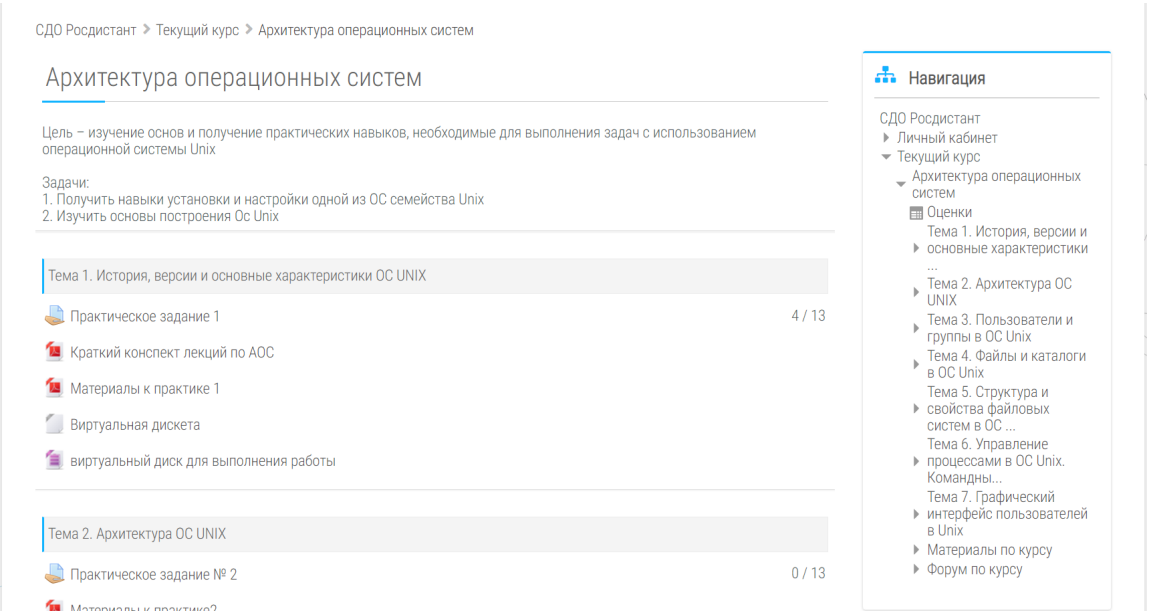

Рисунок 6 – Пример прохождения курса

Далее приведем пример наиболее похожих приложений из «Google Play» в сфере образования.

«Информатика. Теория» – Бесплатное русскоязычное приложение с простым и понятным интерфейсов, но со старым дизайном, которое содержит базу теоретической информации по основным темам информатики. Ниже на рисунке 7 представлен основное окно приложения.

Рисунок 7 – Основное окно приложения

«ЕГЭ Информатика» - Бесплатное русскоязычное мобильное приложение с современным дизайном. Данное приложение в первую очередь разработан для школьников, которые с помощью этого приложения получают всю необходимую информацию для успешной сдачи ЕГЭ по информатике. Имеет много теоретической информации, которое сопровождается показом различных графических материалов. Главное окно приложения представлено на рисунке 8.

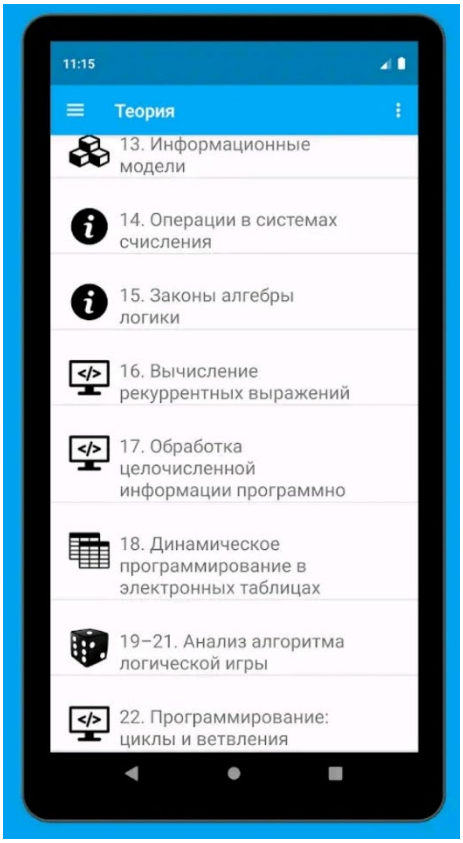

Рисунок 8 – Главное окно приложения

Данное приложение имеет высокий рейтинг 4.7 и является популярным среди подобных приложений, где количество скачиваний составляет порядка 10000. Разработчики данного приложения активно выпускают новую версию, что способствует к улучшению выпускаемого продукта.

«Learn Computer Science» - Бесплатное мобильное приложение, которое предназначение для изучения основ информатики, а также компонентов персонального компьютера. Содержит множество информаций по различным темам с соответствующими картинками. Материал, содержащий в данном приложении, является полностью на английским, что требует базовые знания по английскому языку. Главное окно приложения представлено на рисунке 9.

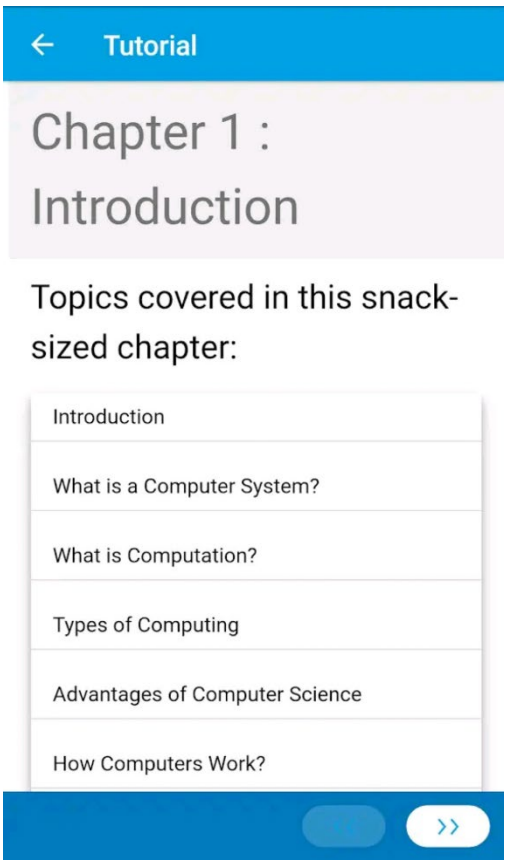

Рисунок 9 – Главное окно приложения

Данное приложение является популярным среди подобных приложений и количество его скачиваний составляет более 100000, что соответствует его рейтингу.

Для сравнения приложений и наглядного представления их преимуществ, ниже составим таблицу, в которой будет перечень требований. Если мобильное приложение будет соответствовать нашим требованиям, то ставим знак «+», в ином случае знак «-».

Таблица 1 – Сравнение приложений

Требования	**Название приложения**		
	Информатика. Теория	ЕГЭ Информатика	Learn Computer Science
Наличие русского языка	+	+	-

Продолжение таблицы 1

Современный дизайн	-	+	+
Малое потребление ресурсов	+	+	+
Работа в режиме офлайн	+	-	+
Техническая поддержка	-	+	+
Большая база информации	-	+	+
Возможность закрепления пройденного материала	-	+	+
Добавление теоретической информации	-	-	-
Добавление теста	-	-	-

На основе приведенного анализа в таблице 1 можно сделать вывод, что по вышеперечисленным критериям некоторые приложения не соответствуют нашим требованиям, что в свою очередь подтолкнуло меня к созданию мобильного приложения для изучения студентов основам информатики, которое будет соответствовать всем вышеперечисленным требованиям.

Выводы по первому разделу

В данной главе были сформированы требования к созданному приложению, а также был проведен анализ существующих приложений для изучения основ информатики, где в конечном итоге была составлена таблица.

2 Проектирование мобильного приложения для обучения основ информатики

2.1 Требования к мобильному приложению

Данная мобильная обучающая система должна иметь не только современный дизайн и красивые графические элементы приложения, но должно быть простым для использования пользователем.

Самые основные требования к разрабатываемому приложения должны соответствовать модели «FURPS». Основная информация о модели «FURPS» была раскрыта в подглаве 1.1. Но стоит добавить, что модель «FURPS» определяем требования, которые должны соблюдаться при разработке мобильного приложения для обучения студентов основ информатики.

На текущее время определяют два типа требования по вышеописанной модели «FURPS»:

– Нефункциональные требования – регламентируют методы функционирования разрабатываемой системы;

– Функциональные требования – регламентируют поведение разрабатываемой системы.

Ниже написана таблица 2, где описаны основные требования к мобильному приложения для обучения основ информатики.

Таблица 2 – требования к мобильному приложению

Требование	**Статус**	**Полезность**	**Риск**	**Стабильность**
Функциональные требования				
Просмотр контента (тем)	Одобренные	Важное	Средний	Средняя
Прохождение теста	Одобренные	Важное	Средний	Средняя

Продолжение таблицы 2

Вывод результата	Одобренные	Важное	Средний	Низкая
Требования к удобству пользования				
Удобство пользования интерфейсом	Одобренные	Критичное	Низкий	Низкая
Простота управления контентом	Одобренные	Важное	Низкий	Низкая
Требования к надежности				
Бесперебойность работы	Одобренные	Важное	Средний	Средняя
Отказоустойчивость	Одобренные	Важное	Средний	Средняя
Требования к производительности				
Скорость выполнения требований пользователя	Предложенные	Критичное	Средний	Низкая
Малое количество используемых ресурсов устройства	Предложенные	Важное	Средний	Средняя
Требования к поддержке				
Простая установка приложения	Одобренные	Важное	Низкий	Средняя

В таблице 2 были определены требования к разрабатываемому мобильному приложению для обучения основ информатики. Далее переходим к логическому моделированию мобильного приложения.

2.2 Логическое моделирование мобильного приложения

Принципы проектирования – это рекомендации по проектированию полезных и желанных продуктов, систем и услуг, а также рекомендации по успешному и этичному проектированию [3].

Цель логического моделирования мобильного приложения заключается в проверки функционирования логической схемы, то есть происходит процесс проверки функции разрабатываемого мобильного приложения без реализации на текущем этапе. Плюсом данной модели заключается в осуществлении проверки логических функций и ее временные соотношения.

Для построения данной логической модели, необходимо использовать язык моделирования «UML». Она связана с CASE-средствами проектированием информационных систем.

CASE-средства – служат для сокращения времени работы разработчиков и для повышения производительности труда. CASE-средства поддерживают множество технологий проектирования информационных систем.

После выбора технологии логического моделирования необходимо разработать диаграмму последовательности мобильного приложения для обучения основ информатики и диаграмму вариантов.

Диаграмма - является одним из популярных методов описания информационной системы в графическом виде. Для наглядного отображения процесса функционального назначения системы необходимо использовать диаграмму вариантов. В мобильном приложении для обучения основ информатики участвуют такие элементы:

– Пользователь, который изучает темы, проходит тесты и просматривает результат прохождения теста;

– Администратор, который добавляет, изменяет и удаляет теоретическую информацию и тесты из соответствующих разделов.

Далее в таблице 3 будут представлены прецеденты, которые будут реализованы в мобильном приложении.

Таблица 3 – Описание прецедентов мобильного приложения

Прецедент	**Краткое описание**
Добавление, изменение и удаление темы	Добавление, изменение и удаление теоретического материала администратором
Добавление, изменение и удаление теста	Добавление, изменение и удаление теста администратором
Список тем	Возможность просмотра и изучения теоретического материала
Список тестов	Возможность просмотра и прохождения тестов
Список результатов	Просмотр итогового результата после прохождения тестов

Вся необходимая информация для полного понимания диаграмм вариантов описана. Собственно, ниже на рисунке 10 будет реализована диаграмма для понимания вариантов использования мобильного приложения.

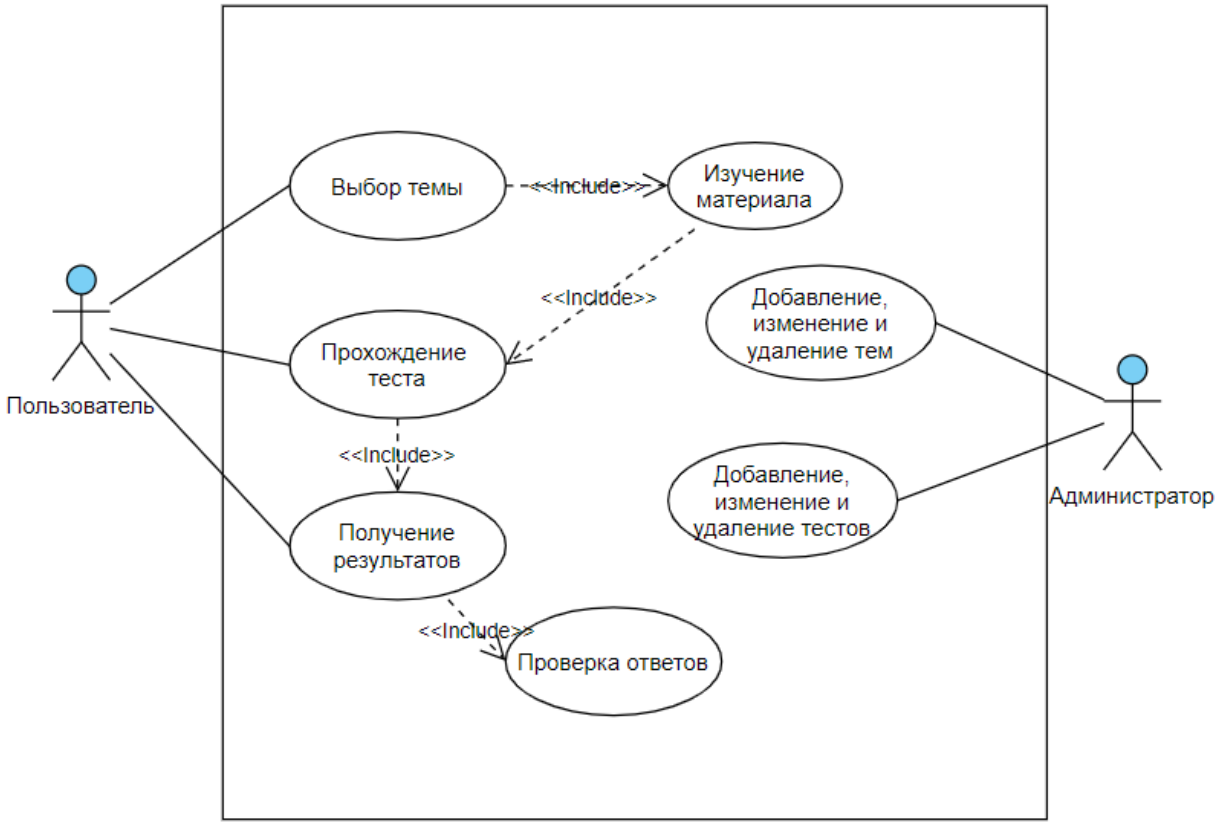

Рисунок 10 – Диаграмма вариантов использования

На диаграмме вариантов отображены основные моменты работы мобильного приложения, такие, как: добавление, изменение, удаление

теоретической информации и тестов; изучение теоретической информации, прохождение тестов, просмотр результатов.

Для более понятного представления взаимодействий между пользователем и мобильным приложением необходимо разработать диаграмму последовательности.

2.3 Разработка диаграммы последовательности мобильного приложения

В соответствии с функциями мобильного приложения для обучения основ информатики необходимо разработать диаграмму последовательности. Для этого мобильному приложению необходимо выполнить ряд действий, которые описаны ниже:

– Администратору необходимо добавить теоретическую информацию в разделе «Темы», а также прикрепить к каждой теме ряд тестов;

– Пользователю при запуске приложения выводится список доступных тем для изучения при нажатии на которой открывается новое окно, где будет продемонстрирована более подробная информацию о выбранной пользователем темы для изучения;

– После изучения выбранной темы пользователь открывает окно вывода тестов и нажимает на пройденную тему после которой открывается окно прохождения тестирования;

– Далее завершается тестирование пользователем и выводится приложением результат.

Ниже на рисунке 11 будет представлена диаграмма последовательности работы мобильного приложения для обучения основ информатики.

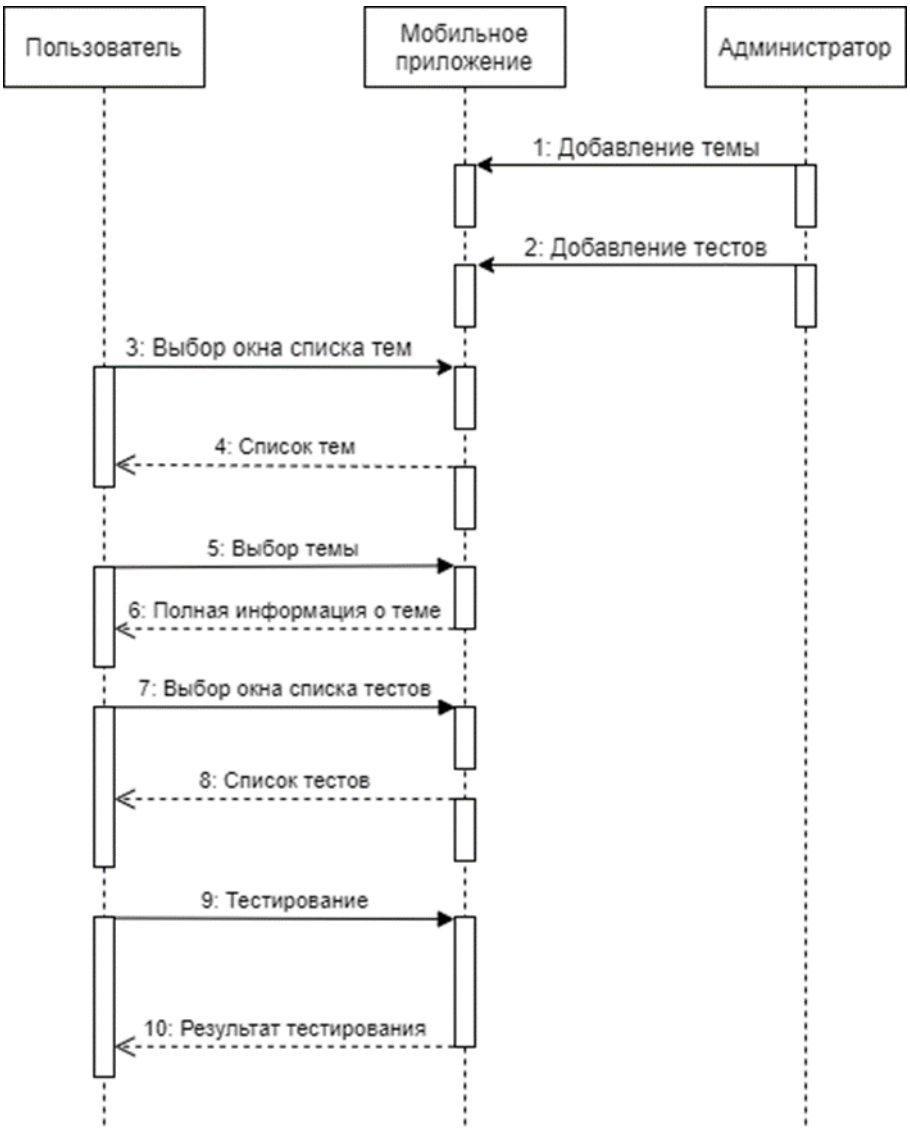

Рисунок 11 – Диаграмма последовательности работы мобильного приложения

На диаграмме последовательности описаны основные моменты работы приложения, которое определит порядок разработки необходимых методов, что в свою очередь упростит работу разработчику мобильного приложения.

2.4 Логическая модель мобильного приложения

Следующим не менее важным этапом моделирования является составление UML-диаграммы классов мобильного приложения для обучения основ информатики [8]. Данная диаграмма будет графически описывать

структуры классов, а также свойства и связи между ними, тем самым представляет логическую модель разрабатываемого мобильного приложения.

В UML диаграмме классов используются такие компоненты, как:

– название класса;

– атрибут класса;

– свойства класса.

Класс – представляет из себя контейнер с множество элементами, которые имеют одинаковые атрибуты и связи.

Модификаторы доступа в классе используются для представления атрибутов класса. Существует три типа модификаторов:

– «-» - private (закрытый);

– «#» - protected (защищенный);

– «+» - public (доступный).

В нашем мобильном приложении используются только два модификатора доступа, такие, как: private и public.

Разработанная UML-диаграмма классов мобильного приложения хорошо демонстрирует классы и его связи в разрабатываемом мобильном приложении. Мобильное приложение для обучения основ информатики состоит из 20 классов, 13 из которых необходимы для работы графических окон приложения.

Для представления логической связи мобильного приложения для обучения основ информатики будем использовать диаграмму классов, которая ниже на рисунке 12 представлена.

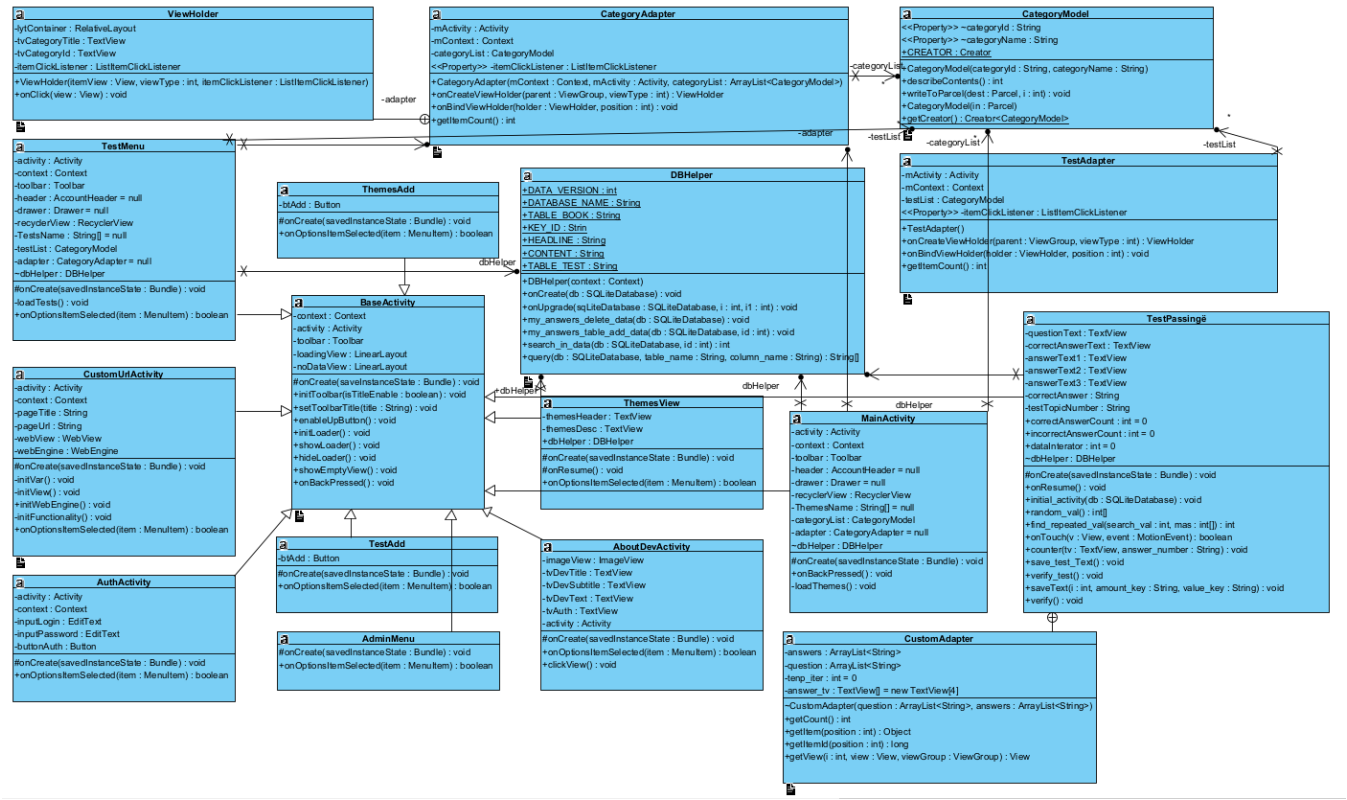

Рисунок 12 – Диаграмма классов мобильного приложения

Все наши классы наследуются от созданного мною родительского класса «BaseActivity», который в свою очередь наследуется от стандартного класса «AppCompatActivity». Во всех класса описаны поля для хранения данных и методы, которые нужны для выполнения каких-либо нужных функций.

Ниже будут более подробно описаны ряд самых основных классов мобильного приложения:

Класс «MainActivity» используется в качестве главного окна мобильного приложения, где описаны функции вывода бокового меню, вывода тем из базы данных и другие значимые функции приложения для корректной работы.

Класс «ThemesView» необходим для полного вывода информации о выбранной пользователем темы.

Класс «TestMenu» используется для вывода информации о тестах, которые отображаются в графическом виде. Все элементы данного класса являются кликабельными, то есть пользователь может выбрать подходящий

тест для проверки своих знаний и в конечном итоге увидеть результат прохождения теста.

Класс «TestPassing» вызывается обработчиком из класса «TestMenu» после выбора нужного теста. Данный класс необходим для отображения окна прохождения теста и последующим выводом всех правильных и неправильных ответов.

Класс «TestResult» нужен для вывода более подробного результата тестирования.

Класс «AboutDevActivity» содержит информацию о приложении и об разработчике. Также кнопку для изменения данных, при нажатии на которой открывается окно ввода логина и пароля для изменения данных в приложении.

Также существуют классы, которые срабатывают после авторизации администратором, а именно:

Класс «ThemesAdd» необходим для добавления, изменения или удаления определенной темы в мобильном приложении.

Класс «TestAdd» необходим для добавления, изменения или удаления определенного теста в мобильном приложении.

Фрагмент кода основного класса находится в Приложение А.

После описания UML диаграммы классов перейдем проектированию базы данных.

2.5 Разработка логической модели данных мобильного приложения

Логическая модель данных используют в качестве начальной точки проектирования будущей базы данных программного продукта. Данная модель описывает объекты предметной области, их атрибуты и связи. Для создания логической модели не нужная какая-либо система управления базой данных (СУБД), можно использовать методологию IDEF1X модель данных, которая описывается как на логическом уровне, так и на физическом уровне.

Методология IDEF1X хорошо показывает концептуальную модель разрабатываемой базы данных приложения.

Логическая модель данных мобильного приложения для обучения основ информатики представлена на рисунке 13.

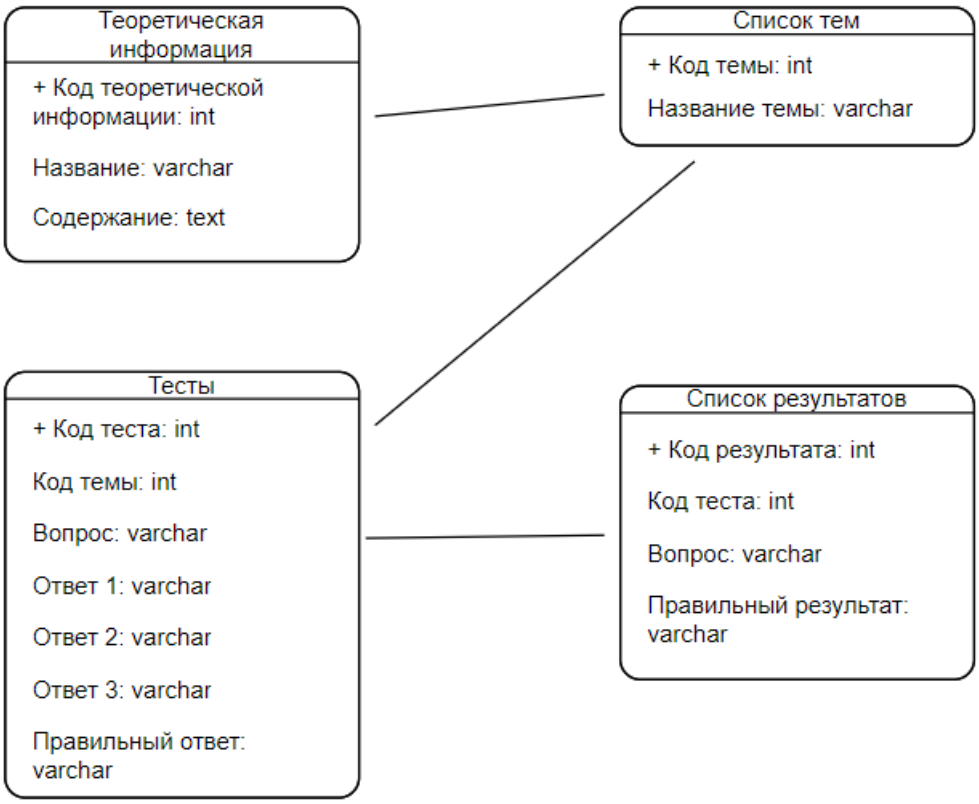

Рисунок 13 – Логическая модель данных приложения

На рисунке 13 отображена логическая модель данных на которой изображены такие сущности, как: «Список тем», «Теоретическая информация», «Список результатов», «Тесты».

Сущность «Список тем» содержит атрибуты «Код темы», который необходим в качестве первичного ключа (Primary key) и «Название темы».

Сущность «Теоретическая информация» содержит атрибуты «Код теоретической информации», который необходим в качестве первичного ключа (Primary key), «Название» и «Содержание».

Сущность «Список результатов» содержит атрибуты «Код результата», который необходим в качестве первичного ключа (Primary key), «Код теста»,

который выступает в роли внешнего ключа (Foreign key), «Вопрос» и атрибут «Правильный ответ».

Сущность «Тесты» содержит атрибуты «Код теста», который необходим в качестве первичного ключа (Primary key), «Код темы», который выступает в роли внешнего ключа (Foreign key), «Вопрос», «Ответ 1», «Ответ 2», «Ответ 3» и атрибут «Правильный ответ».

Можно сделать вывод, что были описаны все необходимые объекты для реализации базы данных мобильного приложения.

2.6 Выбор архитектуры мобильного приложения

Архитектура «клиент-сервер» имеет несколько видов представления:

– двухзвенная архитектура, графическое представление архитектуры которого показано на рисунке 14.

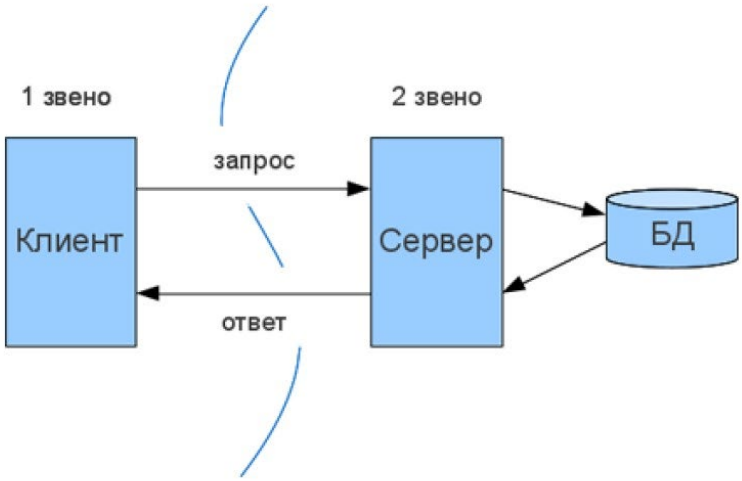

Рисунок 14 – Двухзвенная архитектура

Первое звено – мобильные устройства и используемые приложения пользователей, которые взаимодействуют через сеть с базой данных.

Второе звено – сервер БД которое обрабатывает запросы мобильного приложения пользователя.

Данная архитектура подходит для локальных приложения в виду уменьшенной нагрузки на сеть с помощью которой отправляются запросы и принимаются ответы.

– трехзвенная архитектура, графическое представление архитектуры которого показано на рисунке 15.

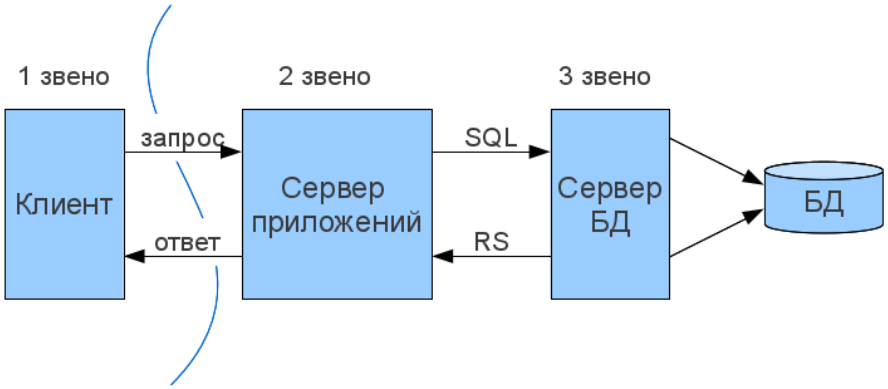

Рисунок 15 – Второй вид архитектуры

В данной архитектура первичным звеном являет работающее приложение пользователя.

Вторым звеном является СУБД приложения, которое представляет собой мост с помощью чего происходит синхронизация и связь между компонентами системы.

Третье звено – сервер БД, которое имеет защиту от прямого подключения пользователя, что в свою очередь является хорошим тоном для повышения безопасности и целостности базы данных и приложения.

После описания архитектур «клиент-серверного» мобильного приложения проведем сравнительный анализ в таблице 4.

Таблица 4 – Сравнение архитектур систем

Характеристика	**Двухзвенная архитектура**	**Трехзвенная архитектура**
Поддержка современных технологий	-	+
Масштабируемость	-	+

Продолжение таблицы 4

Минимальные требования	+	-
Легкая реализация	-	+

Исходя из результатов таблицы 4 можно сделать вывод, что наиболее подходящей архитектурой для мобильного приложения будет трехзвенная архитектура «клиент-сервера» [8].

2.7 Выбор системы управления базой данных мобильного приложения

Не менее важным шагом при разработке мобильного приложения является выбор наиболее подходящей системы управления базой данных (СУБД), так как современные мобильные приложения используют архитектуру «клиент-сервер».

В данный момент существуют много систем управления базой данных, такие, как: Firebase, PostgreSQL, SQLite, MySQL, и другие. Но стоит выбрать именно ту, которая будет наиболее совместимой с мобильным приложением под управлением Android.

Далее, чтобы выбрать наиболее подходящую СУБД, необходимо провести анализ.

MySQL – популярная и бесплатная система управления базой данных, которая разрабатывается корпорацией Oracle. Данную систему используют в качестве сервера в малых и средних приложениях, так как она предлагает большой функционал и малые требования для развертки системы. Логотип системы продемонстрирован на рисунке 16.

Рисунок 16 – Логотип СУБД MySQL

PostgreSQL – объектно-реляционная СУБД, которая популярна своей гибкостью и надежностью. Данная система включает в себя типы данных, операции, функции индексы, домены и т.д. Ее используют в больших проектах, где необходима серьезная СУБД для хранения десятки терабайт данных с чем данная система справляется с легкостью. Логотип системы продемонстрирован на рисунке 17.

Рисунок 17 – Логотип СУБД PostgreSQL

Firebase – облачная база данных разрабатываемой компанией Google. Данная база обновляется в режиме реального времени, то есть при изменении данных она сразу же синхронизируется с устройством. Данные внутри СУБД представлены в виде JSON файла. Единственный минус данной СУБД заключается в необходимости подключаться к сети интернет, чтобы получить актуальную информацию для обновления базы данных приложения, что в свою очередь может создать нагрузку на сеть интернета и замедлит скорость работы мобильного приложения. Логотип системы продемонстрирован на рисунке 18.

Рисунок 18 – Логотип СУБД Firebase

SQLite – встроенная система управления базой данных. СУБД легко подключается с помощью стандартной библиотеки, которая использует вызовы функций SQLite. SQLite доступен на любом Android-устройстве, его не нужно устанавливать отдельно [6]. Исходный код библиотеки передан в общественное достояние [20]. СУБД хранит данные базы в файле, который будет находится внутри приложения, что положительно влияет на скорость обмена информацией между приложением и базой данных. Логотип системы продемонстрирован на рисунке 19.

Рисунок 19 – Логотип СУБД SQLite

После подробного описания систем управления базой данный перейдем к созданию таблицы сравнительного анализа.

Таблица 5 – Сравнительный анализ выбранных систем управления базой данных

Критерии	**MySQL**	**PostgreSQL**	**Firebase**	**SQLite**
Бесплатность	+	+	+-	+
Скорость работы	+	-	-	+
Поддержка реляционных БД	+	+	-	+
Популярность	+	-	-	+
Простота использования	+	-	-	+
Графический интерфейс	+	+	+	-

Исходя из данных таблицы 5 можно сделать вывод, что наиболее подходящей для разработки мобильного приложения для обучения основ информатики подойдет система управления базой данных SQLite, в виде того, что она встраиваемая, легко подключаемая и быстрая в работе.

2.8 Разработка физической модели данных мобильного приложения

Физическая модель данных используют в качестве конечной точки проектирования будущей базы данных программного продукта. Данная модель отображает физическое представление логической модели, то есть на данном этапе происходит создание базы данных.

В данной модели физической сущности являются реляционные таблицы и ее экземплярами являются строки таблицы, которое представлено на рисунке 20.

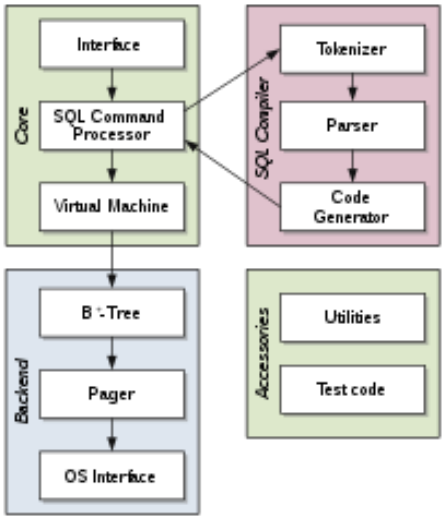

Рисунок 20 – Архитектура СУБД SQLite

Ниже на рисунке 21 будет представлена физическая модель базы данных мобильного приложения для обучения основ информатики используя систему управления базой данных SQLite [7].

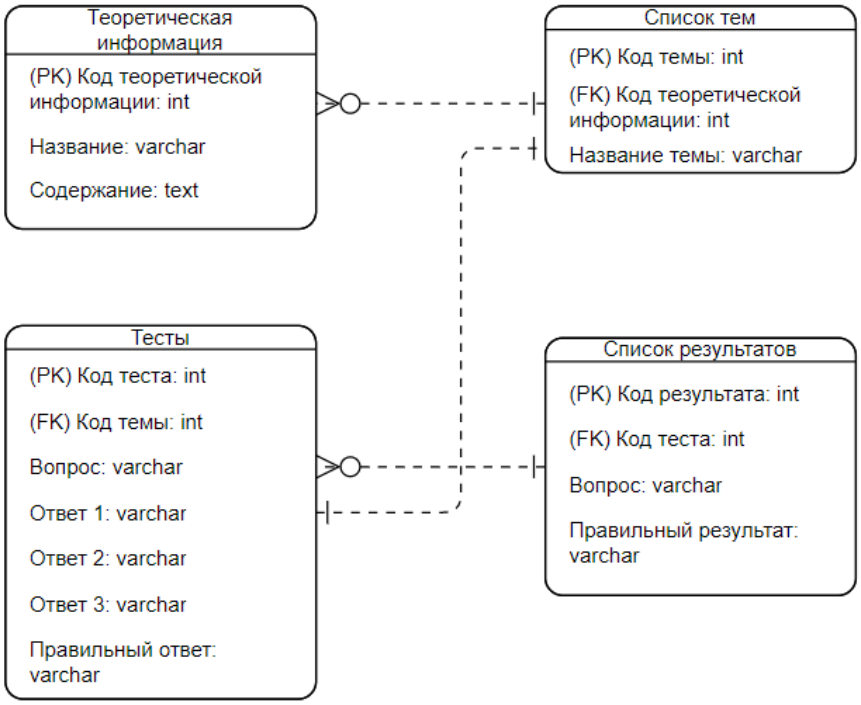

Рисунок 21 – Физическая модель базы данных

Можно сделать вывод, что физическая модель базы данных для мобильного приложения была успешно разработана, опираясь на логическую модель данных [7]. Данный этап можно считать конечным для проектирования базы данных мобильного приложения для обучения основ информатики.

Выводы по второму разделу

Во второй главе были сформированы требования к разрабатываемому мобильному приложению. Было выполнено функциональное моделирование для выявления основных функций мобильного приложения. Далее была построена логическая модель данных, которая необходима для понимания структуры созданной базы данных. Также была разработана физическая модель базы данных.

3 Реализация мобильного приложения для обучения основ информатики

3.1 Выбор языка программирования

Первым делом для реализации мобильного приложения для обучения основ информатики нужно определиться с выбором языка программирования, на котором будет написана программа.

В настоящее время существует множество популярных и различных языков программирования. При выборе языка программирования нужно будет учитывать такие критерии, как: понимание кода, множество справочной информации, скорость работы мобильного приложения и возможность добавления различного рода функционала посредством подключения внешних функций.

Исходя и вышеположенного, мною были выделены такие языки программирования, как:

– C#;

– Java.

C# – объектно-ориентированный язык программирования, синтаксис которого во многом похож на синтаксис языка C++. Данный язык программирования был разработан в 2001 году корпорацией «Microsoft» для платформы .NET. C# является типизированным языком, как и Java.

Java – объектно-ориентированный и высокоуровневый язык программирования, который является наиболее востребованным и популярным среди разработок мобильного приложения. Данный язык программирования был разработан и представлен компанией «Oracle». Язык Java был задуман в 1991 году сотрудниками компании Sun Microsystems Джеймсом Гослингом, Патриком Нотоном, Крисом Уортом, Эдом Фрэнком и Майком Шериданом [17].

Основную сущность языка Java составляют библиотеки файлов, называемые классами, каждый из которых содержит небольшие фрагменты проверенного, готового к выполнению кода [4]. Java имеет многоуровневую структуру, которая будет представлена на рисунке 22.

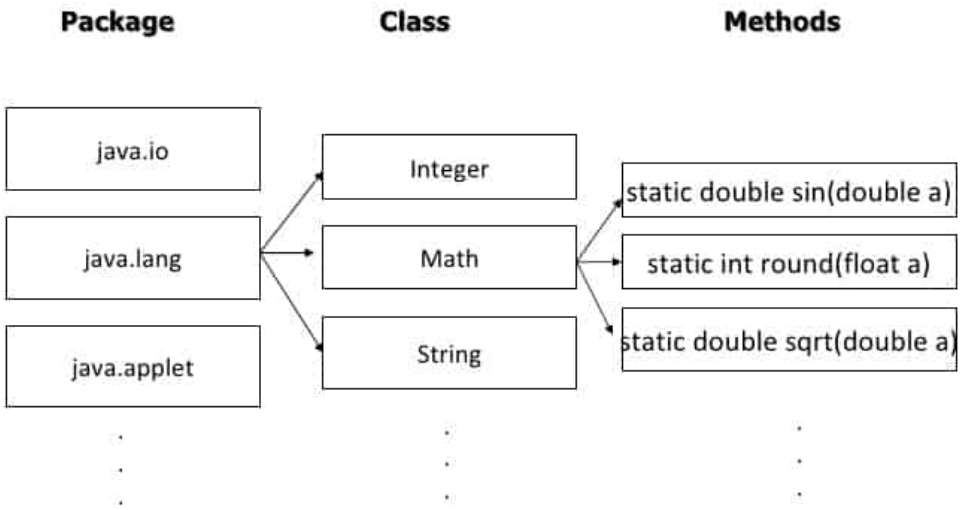

Рисунок 22 – Многоуровневая структура Java

Ниже будет представлена таблица, где было проведено сравнение языков программирования.

Таблица 6 – Сравнительный анализ языков программирования

Критерии	**C#**	**Java**
Популярность в использовании	-	+
Читаемость кода	+	+
Требовательность к ресурсам	+	+
Наличие библиотек	-	+
Удобство сборки	+	+
Опыт работы	-	+

Таким образом, по результатам сравнительного анализа мною был выбран язык программирования Java для дальнейшей разработки мобильного приложения для обучения студентов основам информатики.

3.2 Выбор среды разработки для реализации мобильного приложения

Среда разработки – важный инструмент для разработчиков, где происходит реализация программного обеспечения, поэтому следует выбрать ту среду, которая будет наиболее удобной в использовании, которая будет включать в себя все важные средства для разработки программы.

Существует множество сред для разработки мобильного приложения, каждый из которых имеет свои преимущества и недостатки. Ниже разберем такие среды, как:

– IntelliJ IDEA;

– Eclipse;

– Android Studio.

IntelliJ IDEA – бесплатная и свободная среда разработки, разработанная компанией «JetBrains». Используется для написания кода с использованием наиболее популярных языков программирования Java, JavaScript и Python. Главный экран среды продемонстрирован на рисунке 23.

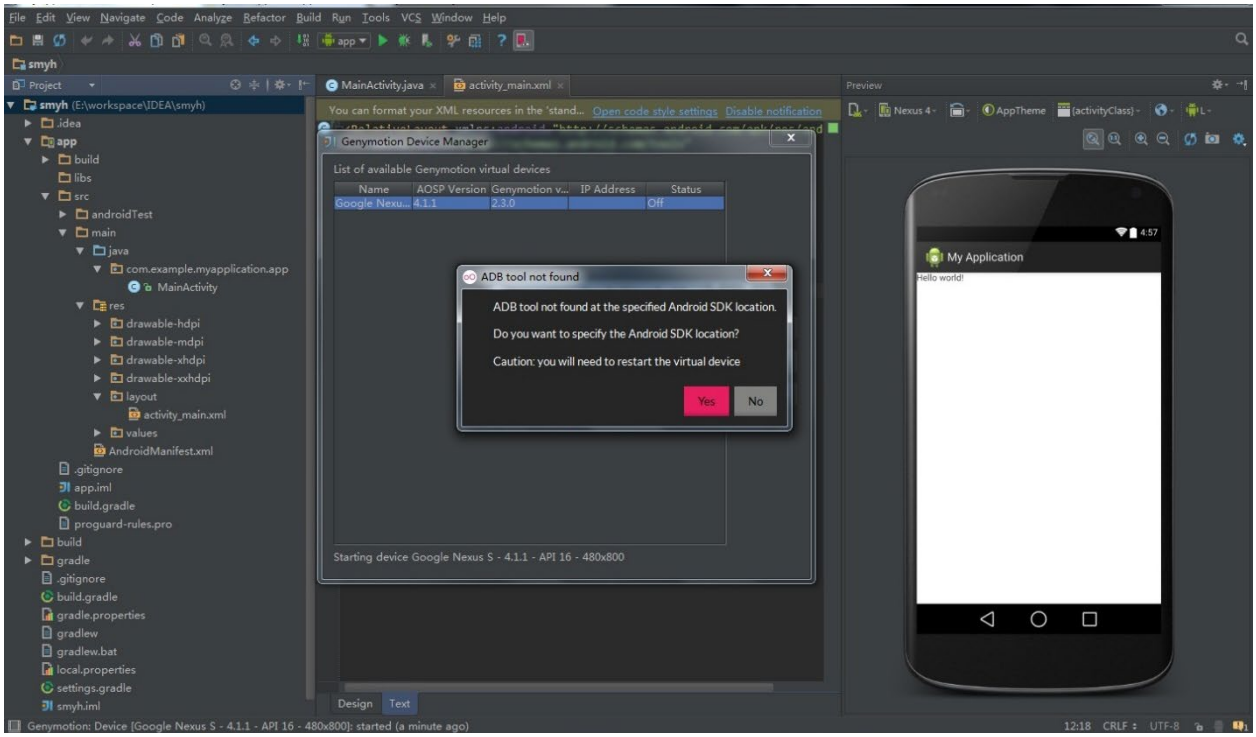

Рисунок 23 – Среда разработки IntelliJ IDEA

Eclipse – бесплатная и свободная среда разработки, разработанная компанией «Eclipse Foundation». Данная среда популярна среди разработок модульных кроссплатформенных приложений. Основное окно среды разработки продемонстрировано на рисунке 24.

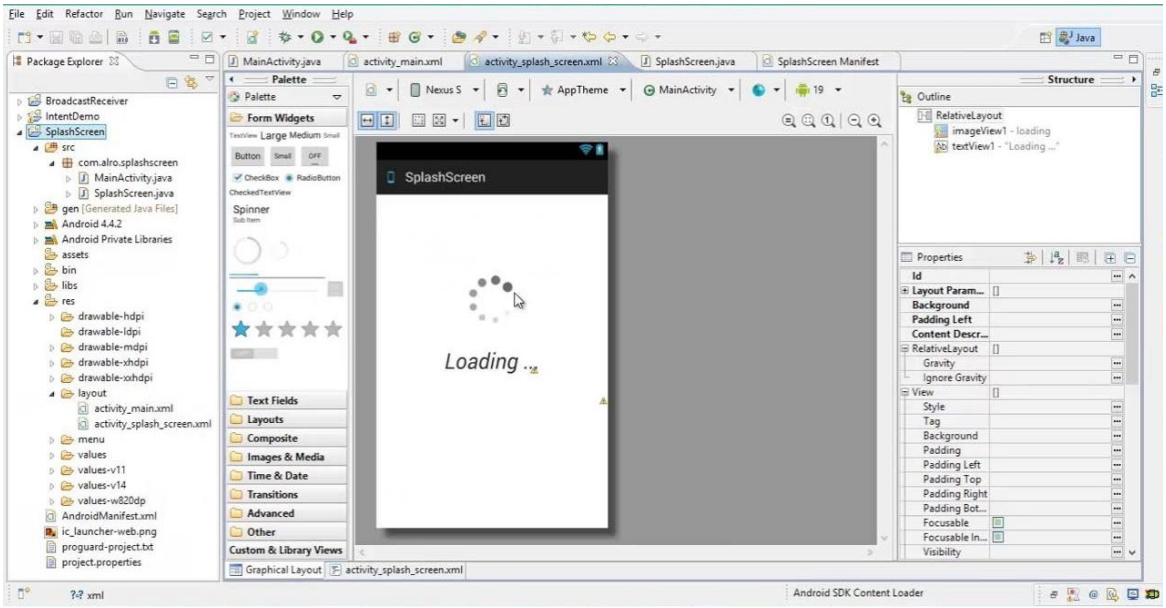

Рисунок 24 – Среда разработки Eclipse

Android Studio – бесплатная и наиболее популярная среда разработки для создания мобильных приложений на языке Java и Kotlin. Android Studio была представлена в 2013 компанией Google. Android Studio, основанная на программном обеспечении IntelliJ IDEA от компании JetBrains [14]. Данную среду можно считать официальной среды для разработки мобильного приложения. Основное окно среды разработки продемонстрировано на рисунке 25.

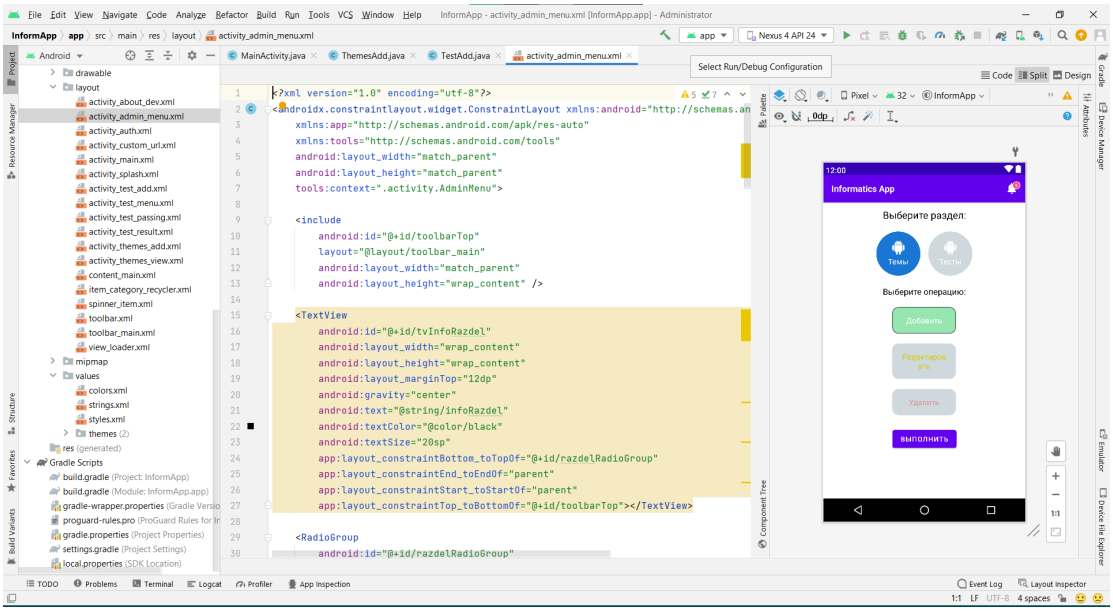

Рисунок 25 – Среда разработки Android Studio

В качестве среды разработки под операционную систему Android мною была выбрана среда разработки Android Studio, т.к. она оказалась наиболее удобной для разработки в виду того, что предназначена именно для разработки мобильных приложений.

3.3 Архитектура и компоненты мобильного приложения

Для начала нам нужно определиться с архитектурой мобильного приложения [9]. Разработанное мобильное приложение для обучения основ информатики будет следовать шаблону архитектуры «MVP».

MVP – расшифровывается, как:

– M – Model (Модель);

– V – View (Вид);

– P – Present (Представитель).

Основная задача — получение обратной связи для формирования гипотез дальнейшего развития продукта [5]. MVP, возник как альтернатива MVC [18].

Графическое представление модели шаблона можно увидеть на рисунке 26.

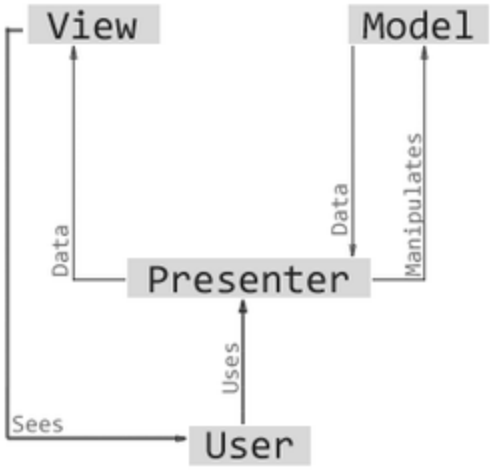

Рисунок 26 – Пример работы архитектуры

После выбора модели архитектуры следует описать ряд каталогов, которые содержат файлы кода, в которой происходит управление приложением, элементы экрана и поведение программы на действия пользователя.

3.4 Реализация основных компонентов мобильного приложения

Работа начинается с создания проекта Android. Проект Android содержит файлы, из которых состоит приложение [10].

Для этого открываем Android Studio и нажимаем кнопку «New» и выбираем платформу, где у нас будет работать приложение и шаблон мобильного приложения. В нашем случае это будет пустой проект с минимальными добавленными компонентами, то есть «Empty Activity». Окно создания проекта в среде разработки Android Studio представлен на рисунке 27.

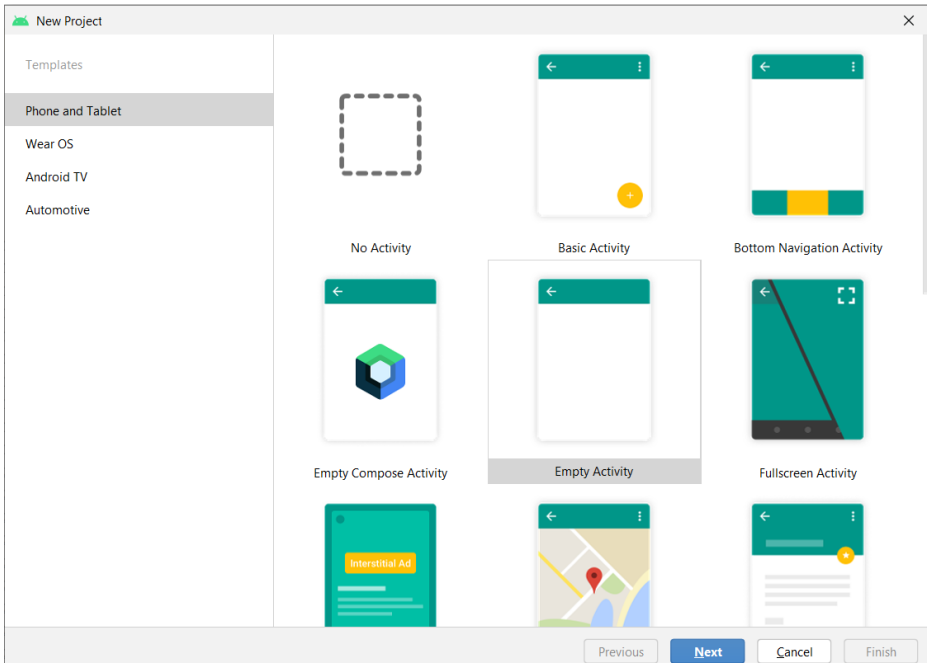

Рисунок 27 – Создание проекта

После создания проекта можно увидеть в папке «activity» файл MainActivity, который является основным классом при загрузке приложения. Все приложения включают в себя компоненты «Activity», где будет описана функциональная часть каждого окна. Каждое «Activity» имеет свой .xml файл, где описывается графическая часть окна, представленное на рисунке 28.

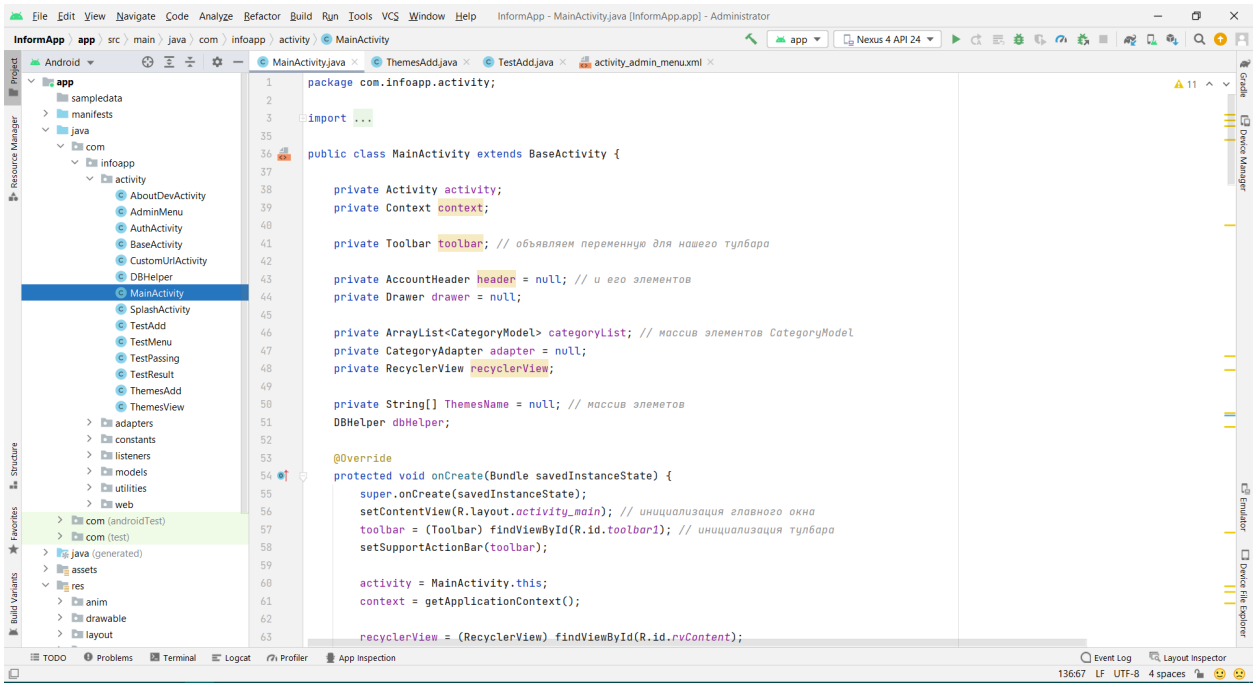

Рисунок 28 – Главное окно проекта

Весь проект в итоге будет выглядеть следующим образом ниже на рисунке 29 [19]:

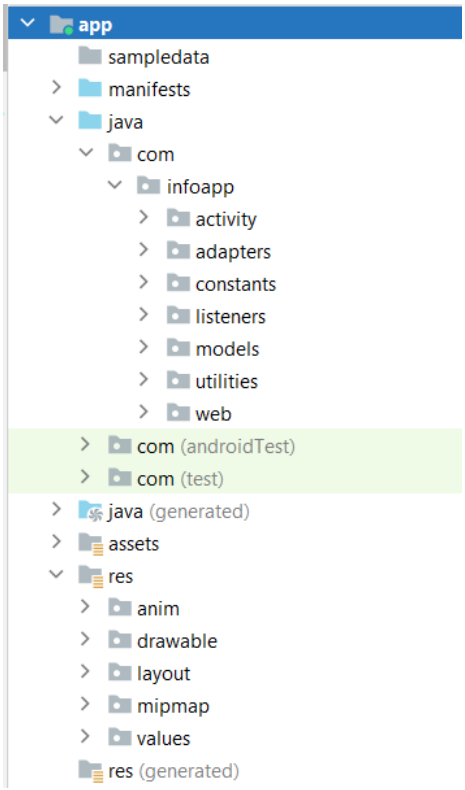

Рисунок 29 – Структура мобильного приложения

Разработанное мобильное приложение включает в себя такие основные папки, как:

– Необходимые приложению полномочия (как и многое другое) указываются в файле Манифеста приложения (AndroidManifest.xml) [2]. AndroidManifest.xml представляет собой файл, где хранится вся информация о приложении: имя пакета, компоненты приложения, разграничение доступа, список библиотек и другие важные описания;

Псевдокод манифеста представлен на рисунке 30:

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.infoapp">
    <!-- Даем права доступа приложению для подключения к интернету -->
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.INTERNET" />
```

Рисунок 30 – Манифест приложения

– java – папка, где хранятся классы, объекты, модели, адаптеры и другие важные функциональные компоненты, то есть в данной папке хранятся файлы исходного кода мобильного приложения. Содержит файлы исходного кода Java, включая тестовый код JUnit [15];

– res – папка, которая хранит графические ресурсы мобильного приложения, а именно: графические компоненты окон, цвет шрифта, иконки и т.д.

Далее после описания основных компонентов мобильного приложения следует перейти к создания классов для написания функциональной части мобильного приложения.

Первым делом создаем графическую часть главной страницы, где будут выводится темы для обучения. Дизайн страниц были созданы с помощью графического модуля среды разработки, которое продемонстрировано на рисунке 31.

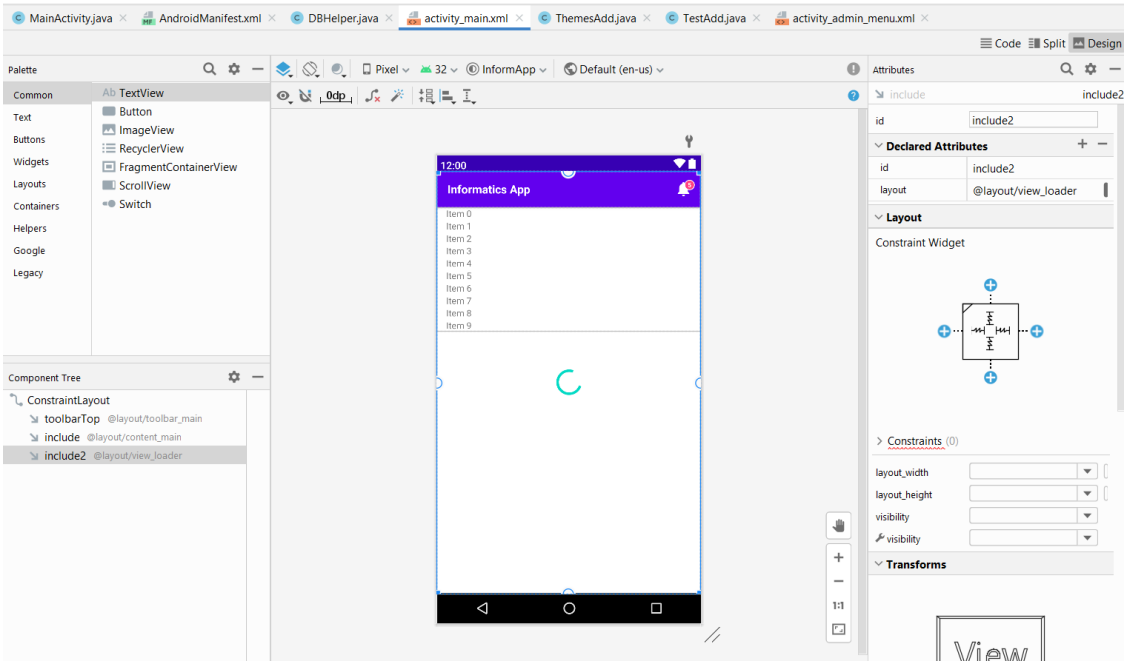

Рисунок 31 – Графическая часть главного окна

Далее разберем наиболее основные функциональные части мобильного приложения для понимая. Ниже на рисунке 32 будет представлен фрагмент кода главной страницы.

```
private ArrayList<CategoryModel> categoryList; // массив элементов CategoryModel
private CategoryAdapter adapter = null;
private RecyclerView recyclerView;

private String[] ThemesName = null; // массив элеметов
DBHelper dbHelper;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main); // инициализация главного окна
    toolbar = (Toolbar) findViewById(R.id.toolbar1); // инициализация тулбара
    setSupportActionBar(toolbar);

    activity = MainActivity.this;
    context = getApplicationContext();

    recyclerView = (RecyclerView) findViewById(R.id.rvContent);
    // корневому макету списка присваевам GridLayout, который нужен для вывода элементов в виде сетки (2)
    recyclerView.setLayoutManager(new GridLayoutManager(activity,  spanCount: 2, GridLayoutManager.VERTICAL,  reverseLayout: false));

    categoryList = new ArrayList<>();
    adapter = new CategoryAdapter(context, activity, categoryList);
    recyclerView.setAdapter(adapter);

    initLoader();

    dbHelper = new DBHelper( context: this);
    //final SQLiteDatabase database = dbHelper.getWritableDatabase();

    loadThemes();
```

Рисунок 32 – Класс «MainActivity»

В данном классе происходит инициализация переменных, компонентов главного экрана, компонентов «toolbar» и слайдбар, а также происходит вызов функции подключения к базы данных и вызов loadThemes().

loadThemes() служит для выгрузки списка тем из базы данных. Работа данной функции происходит следующим образом:

– Открываем подключение к базе данных для получения информации из нужной таблицы;

– Создаем курсор, который будет хранить всю выгруженную из базы информацию. Курсор предоставляет доступ к наборам записей базы данных [16];

– Создаем вспомогательный строковый массив, где отправим из курсора информацию заранее переведя в строковый тип из массива объектов;

– Далее циклом будет происходить выгрузка данных из базы в строковый массив ThemesName, который в свою очередь будет добавлять

данный в наш categoryList. categoryList служит для вывода данных на главном окне;

– После выгрузки в categoryList скрываем анимацию загрузки и информируем адаптер об изменение данных;

– Если пользователь выберет любой блок на главном экране, то срабатывает метод setItemClickListener, где мы передаем данные categoryList в categoryModel;

– Дальше создаем интент для перехода в окно, где будет происходить полный вывод информации, выбранной пользователем, а также передаем ему нужные параметры.

Интент (intent) — объект, который может использоваться компонентом для взаимодействия с ОС [13].

Фрагмент кода функции загрузки темы будет представлен на рисунке 33.

```
private void loadThemes () {
    // открываем подключение
    final SQLiteDatabase database = dbHelper.getReadableDatabase();

    //Создаем курсор для вывода данных
    Cursor cursor=database.rawQuery( sql: "select *from " + DBHelper.TABLE_BOOK + " where id >= 0",  selectionArgs: null);
    cursor.moveToFirst();

    ThemesName = new String [cursor.getCount()];

    Log.d( tag: "dd", msg: "records = " + cursor.getCount());
    int i=0;
    do{
        try {
            ThemesName[i] = cursor.getString(cursor.getColumnIndex(DBHelper.HEADLINE));
            categoryList.add(new CategoryModel(Integer.toString(i), ThemesName[i])); // добавляем в категори лист данные
            i++;
        }catch (Exception e){
              e.printStackTrace();
        }


    }while (cursor.moveToNext());
    hideLoader(); // после выгрузки информации скрываем анимацию
    adapter.notifyDataSetChanged(); // информируем апатер об изменении данных
    // recycler list item click listener
    adapter.setItemClickListener(new ListItemClickListener() {
        @Override
        public void onItemClick(int position, View view) {

            CategoryModel model = categoryList.get(position);
            Intent intent = new Intent(activity, ThemesView.class);
```

Рисунок 33 – Функция loadThemes()

Для работы с данным нам необходимо создать базу данных используя физическую модель, которая описана в подглаве 2.8. Далее создаем класс

«DBHelper», где у нас будут описаны основные свойства и методы для подключения к базе данных. Ниже на рисунке 34 будет представлен псевдокод подключения к базе данных.

```
public static final String TOPIC_NAME="topic";
public static final String QUESTION = "question";
public static final String TRUE_ANSWER = "true_answer";
public static final String ANSWER1 = "answer1";
public static final String ANSWER2 = "answer2";
public static final String ANSWER3 = "answer3";

public static final String TABLE_MyANSWERS = "my_answers";
public static final String ANSWERS_ID="id";
public static final String ANSWERS_QUESTION = "question";
public static final String ANSWERS_TRUE_ANSWER = "true_answer";
public static final String ANSWERS_ANSWER1 = "answer1";
public static final String ANSWERS_ANSWER2 = "answer2";
public static final String ANSWERS_ANSWER3 = "answer3";

public DBHelper(Context context) { super(context, DATABASE_NAME, factory: null, DATA_VERSION); }

// метод для создания таблиц при первом запуске
@Override
public void onCreate(SQLiteDatabase db) {
    db.execSQL("create table " + TABLE_BOOK + "(" + KEY_ID
            + " integer primary key autoincrement," + HEADLINE + " text," + CONTENT + " text" + ")");
    db.execSQL("create table " + TABLE_TEST + "(" + TEST_ID
            + " integer primary key autoincrement," + TOPIC_NAME + " text references " + TABLE_BOOK + "(" + HEADLINE +")," +
            QUESTION + " text," + TRUE_ANSWER + " text," + ANSWER1 + " text," + ANSWER2 + " text," + ANSWER3 + " text" + ")")


}
```

Рисунок 34 – Класс «DBHelper»

Метод «onCreate()» служит для подключения существующей базы данных, однако, в случае отсутствия базы может создавать новую.

Также существует метод «onUpgrade()», который служит для записи в таблицу при изменении данных, «my_answer_delete_table()» для удаления таблицы и «my_answer_table_add_data()», который нужен для добавления данных в таблицу my_answer, представленное на рисунке 35.

```
@Override
public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
    db.execSQL("drop table if exists " + TABLE_BOOK);//для удаления таблицы
    db.execSQL("drop table if exists " + TABLE_TEST);
    /*db.execSQL("create table " + TABLE_BOOK + "(" + KEY_ID
            + " integer primary key," + KEY_NAME + "text," + KEY_CONTENT + "text" + ")");*/
    onCreate(db);
}
/*создание таблицы my_answers */
public void my_answers_table(SQLiteDatabase db){
    /*db.execSQL("create table " + TABLE_MyANSWERS + "(" + ANSWERS_ID + " integer primary key autoincrement," +
            ANSWERS_QUESTION + " text," + ANSWERS_TRUE_ANSWER + " text," + ANSWERS_ANSWER1 + " text," + ANSWERS_ANSWER2 + " text,"
}
/*функция для удаления таблицы my_answers*/
public void my_answers_delete_data(SQLiteDatabase db){
    db.execSQL("delete from " + TABLE_MyANSWERS );
}
/*функция для добавления данных в таблицу my answers*/
public void my_answers_table_add_data(SQLiteDatabase db,int id){
    if(search_in_data(db,id)==0) {
        Cursor cursor = db.rawQuery( sql: "select *from " + DBHelper.TABLE_TEST + " where id = " + id, selectionArgs: null);
        ContentValues contentValues = new ContentValues();
        contentValues.put(ANSWERS_QUESTION, cursor.getString(cursor.getColumnIndex(QUESTION)));
        contentValues.put(ANSWERS_TRUE_ANSWER, cursor.getString(cursor.getColumnIndex(TRUE_ANSWER)));
        contentValues.put(ANSWERS_ANSWER1, cursor.getString(cursor.getColumnIndex(ANSWER1)));
        contentValues.put(ANSWERS_ANSWER2, cursor.getString(cursor.getColumnIndex(ANSWER2)));
        contentValues.put(ANSWERS_ANSWER3, cursor.getString(cursor.getColumnIndex(ANSWER3)));
        db.insert(TABLE_MyANSWERS, nullColumnHack: null, contentValues);
    }
}
```

Рисунок 35 – Фрагмент класса «DBHelper»

3.5 Описание основных принципов работы мобильного приложения

Основная задача разработанного мобильного приложения – это предоставление теоретической информации и закрепление путем прохождения тестов, а также просмотр результатов.

При запуске пользователем приложение перед ним открывается окно загрузки, которое состоит из логотипа приложения, краткое описание и анимации загрузки, который представлен на рисунке 36.

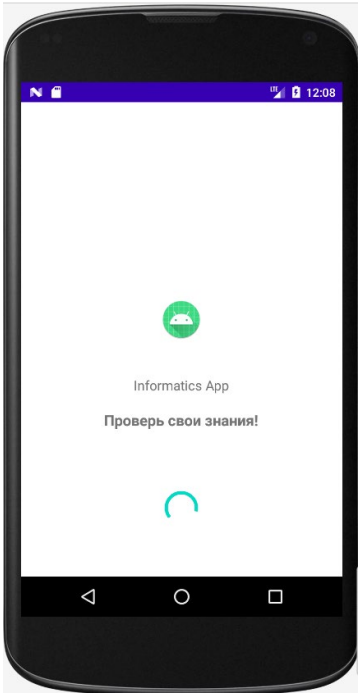

Рисунок 36 – Окно загрузки приложения

Далее после загрузки открывается главное окно, где мы видим пронумерованные, разноцветные колонки, где отображаются название тем. Каждый блок является «кликабельным». Ниже на рисунке 37 можно увидеть графическое представление главного окна.

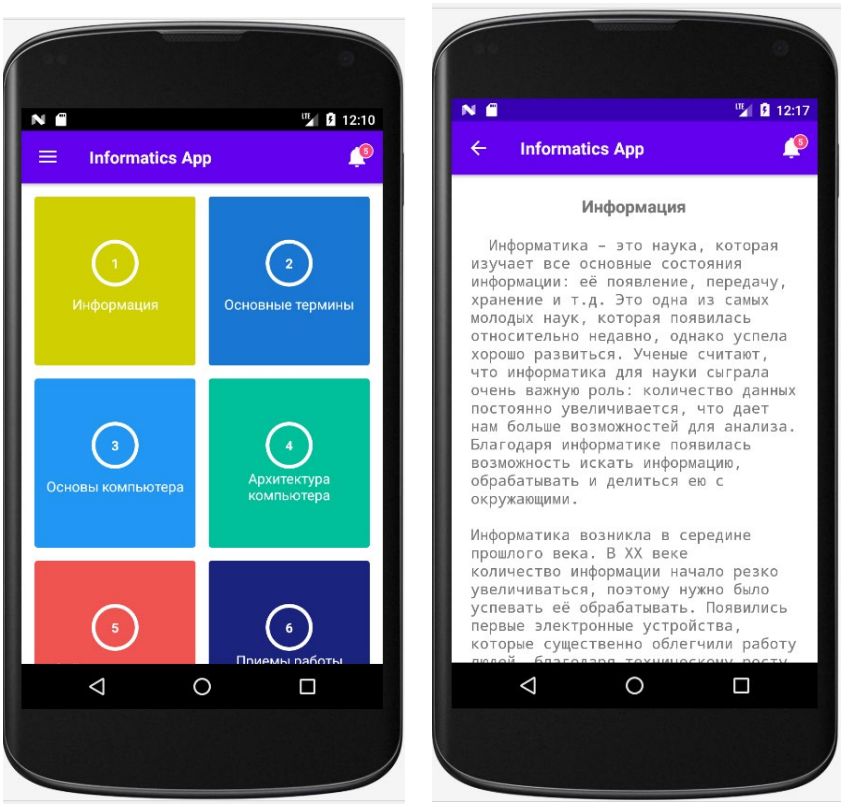

Рисунок 37 – Главное окно приложения

При нажатии на любой блок из вышепоказанного, будет открываться окно, в котором выводится полная информация выбранной пользователем темы.

После изучения темы пользователю доступно возврат домой с помощью иконки на левом верхнем углу, либо нажатием кнопки «Назад» на мобильном устройстве, где открывается главное окно приложения. Далее для закрепления пройденного материала пользователю доступно прохождения теста. В данном случае пользователь может нажать на иконку, которая расположена на левом верхнем углу, после которого открывается боковое меню, где доступны такие функции мобильного приложения, как:

– «Темы» – главный раздел приложения, где находятся темы;

– «Тесты» – раздел, где расположен список тестов;

– «Мои результаты» – раздел в котором пользователю доступны список пройденных тестов;

– «О приложении» – раздел об общей информации о приложении;

– После нажатия в боковом меню на пункт «Тесты» открывается похожее меню, как в разделе «Темы», которое ниже представлен на рисунке 38.

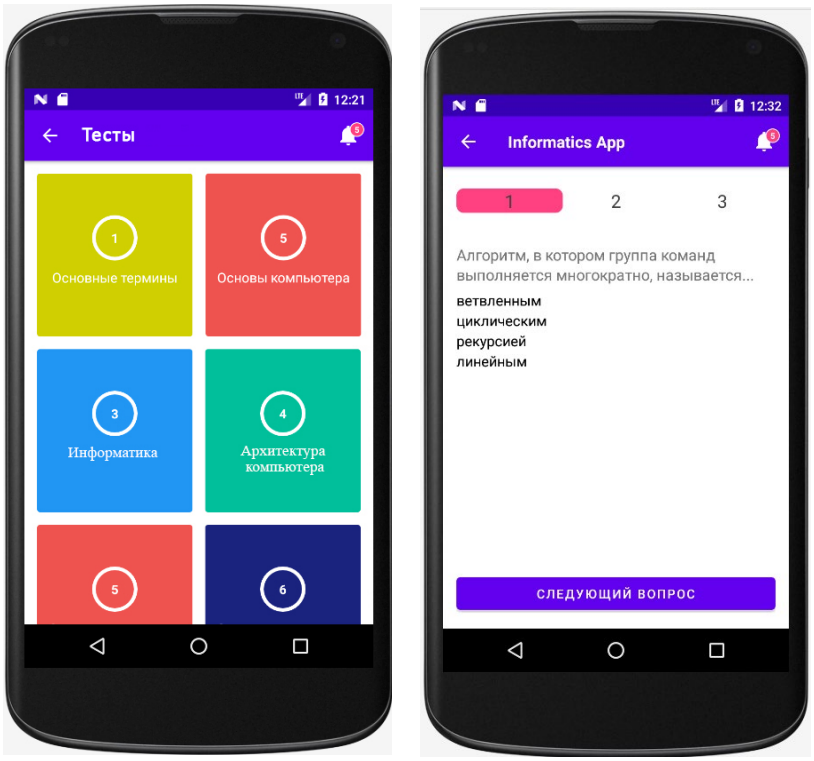

Рисунок 38 – Раздел «Тесты»

После выбора пользователем тему для тестирования открывается окно прохождения теста.

Тестовые задания содержат вопросы из пройденной темы пользователем. Далее после прохождения тестирования, пользователю выводится результат.

При нажатии на боковом меню пункта «О приложении» пользователю открывается окно, на котором выводится общая информация о приложении и об авторе. На рисунке 39 будет продемонстрировано графическое представление страницы «О приложении».

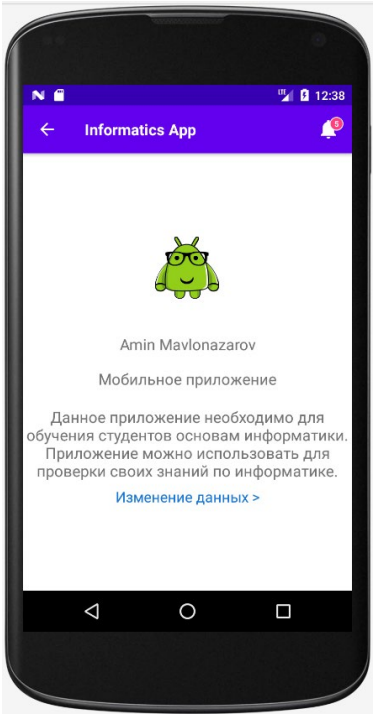

Рисунок 39 – Окно информации о приложении

На данном экране можно увидеть текст голубым цветом, который намекает на то, что у него существует «кликабельное» свойство. При нажатии на этот текст открывается окно авторизации. Данное окно существует для того, чтобы ввести логин и пароль администратора для прохода в панель администратора.

Ниже на рисунке 40 будет представлено окно авторизации.

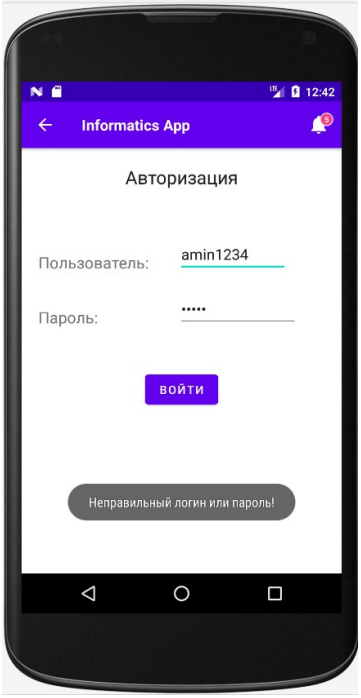

Рисунок 40 – Окно авторизации

Если пользователь вводит неправильный логин или пароль, то система его уведомляет об этом. Уведомление реализовано с помощью класса «Toast». После ввода правильного логина и пароля будет выведено уведомление и откроется окно администратора. На рисунке 41 будет представлена страница администратора.

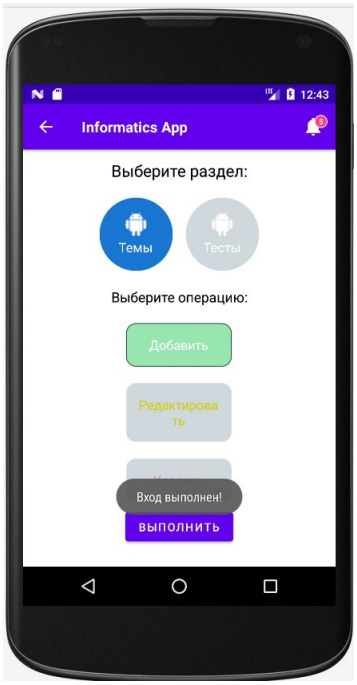

Рисунок 41 – Окно администратора

В данном окно администратору доступны такие функции для работы с темами: добавление, изменение и удаление темы. При выборе администратором функции добавлении открывается следующее окно:

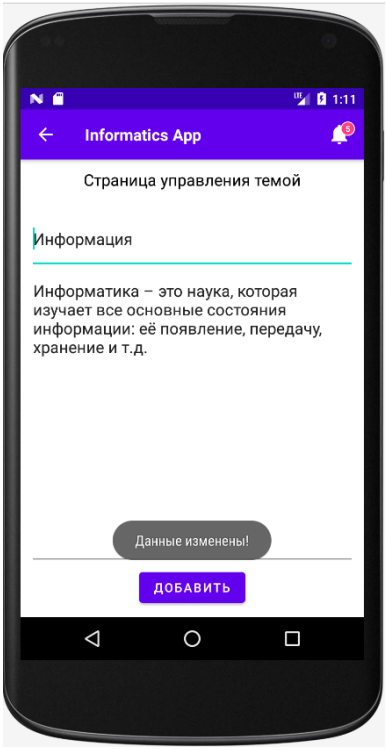

Рисунок 42 – Окно добавление темы

В данном окно реализованы вывод полей ввода, где пользователю необходимо заполнить название темы и описание, далее нужно нажать на кнопку «Добавить». После успешного добавления данных в базу система проинформирует администратора уведомлением, которое можно увидеть на рисунке 42.

Также администратору доступна функция изменения темы. Для этого в панели администратора необходимо выбрать опцию «Редактировать» и нажать кнопку «Выполнить». После этого открывается окно редактирования.

В данном окно администратору представлен список после нажатия на него раскрывается, где будут доступны существующие темы для изменения.

Список показывает лишь выбранный элемент в одну строку, но, если нажать на стрелку, открываются другие варианты выбора [1]. Далее необходимо выбрать тему и поля ввода будут заполнены информацией о

выбранной темы. Ниже на рисунке 43 представлено окно редактирования темы.

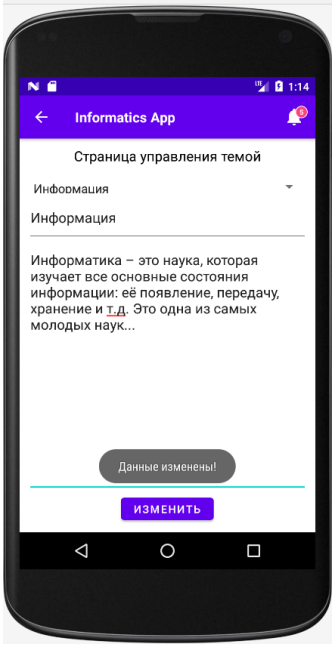

Рисунок 43 – Окно изменения темы

Также администратору доступно окно добавления теста. Для этого в панели администратора необходимо переключить раздел с «Темы» на «Тесты» и выбрать вариант «Добавить» после которого нажать кнопку «Выполнить». Далее открывается окно добавления представленное на рисунке 44, где нужно выбрать тему, в которую нужно добавить тест.

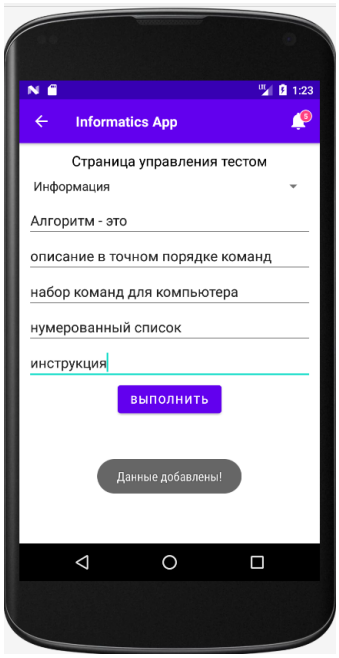

Рисунок 44 – Окно добавление теста

В данном окне необходимо нажать на список и выбрать доступную тему. После этого заполнить всю необходимую информацию и нажать кнопку «Выполнить».

Существует окно редактирования теста. Для этого нужно выбрать в панели администратора опцию «Редактировать» и нажать «Выполнить», после чего открывается окно, которое продемонстрировано на рисунке 45.

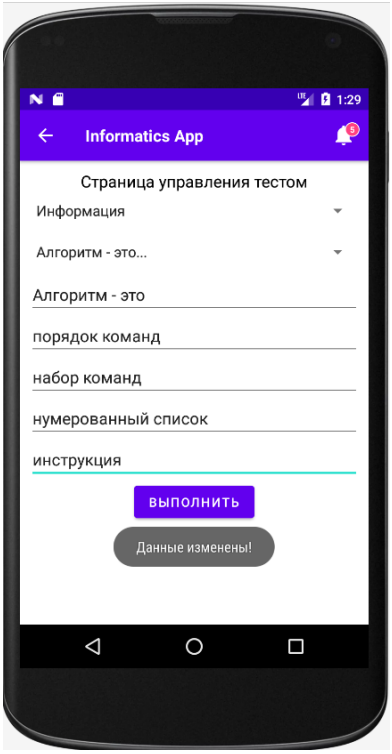

Рисунок 45 – Окно редактирования теста

В данном окне администратору необходимо выбрать тему и название теста в раскрывающем списке, после которого заполняются поля ввода, которые можно потом изменить и нажать на кнопку «Выполнить».

Также администратора доступна функция удаления теста. Необходимо выбрать опцию «Удалить», где открывается окно, в котором необходимо выбрать название теста и нажать кнопку «Удалить».

3.6 Тестирование мобильного приложения

Целью тестирования является поиск наличия разного рода ошибок в работе мобильного приложения. Данный процесс дает понять пользователю качество работы приложения.

Функциональное тестирования представляет собой процесс тестирования программного обеспечения в целях проверки разработанных функций, то есть проверка программы на выполнения задачи в разных условиях, которые необходимы пользователю. Функциональные требования дает понять, что делает мобильное приложение, какие функции выполняет.

Ниже проведем тестирование некоторых главных функций приложения.

Тест № 1

Цель: провести тестирование функции авторизации пользователя в панели авторизации.

Ожидаемый результат: успешная авторизация пользователя и открытие панели администратора.

Входные данные: пользователь находится в окне авторизации администратора.

Процедура тестирования: пользователь вводит логин и пароль в соответствующие поля ввода и нажимает кнопку «Войти», в случае правильности ввода открывается панель администратора, иначе, мобильное алгоритм уведомляется пользователя о неправильности введенных данных.

Полученный результат: совпадает с ожидаемым результат.

Вывод: тест № 1 успешно пройден.

Тест № 2

Цель: провести тестирование функции добавления темы.

Ожидаемый результат: введенные данные пользователем успешно добавились в базу данных.

Входные данные: пользователь находится в окне добавления темы.

Процедура тестирования: пользователь выбирает в панели администратора раздел «Темы» и вариант «Добавить», после чего открывается окно добавления, где нужно заполнить соответствующие поля ввода и нажать кнопку «Выполнить». В случае успешного выполнения алгоритма, пользователю выводится уведомление «Успешно добавлено», в ином случае «Ошибка!».

Полученный результат: совпадает с ожидаемым результат.

Вывод: тест № 2 успешно пройден.

Тест № 3

Цель: провести тестирование функции изменение темы.

Ожидаемый результат: выбранное пользователем в окне изменения темы появляется информация в соответствующие поля, где редактируются и добавляются в базу.

Входные данные: пользователь находится в окне изменения темы.

Процедура тестирования: пользователь выбирает в панели администратора раздел «Темы» и вариант «Редактировать», после чего открывается окно изменения темы, где в соответствующие поля появляется вся данные темы, после чего пользователь может их отредактировать и нажать кнопку «Выполнить». В случае успешного выполнения алгоритма, пользователю выводится уведомление «Успешно отредактировано», в ином случае «Ошибка!».

Полученный результат: совпадает с ожидаемым результат.

Вывод: тест № 3 успешно пройден.

Выводы по третьему разделу

В данной главе был выбран язык программирования и среда разработки для создания мобильного приложения. Также описана архитектура и компоненты приложения. Были реализованы и описаны основные компоненты приложения, а также описаны основные принципы работы с мобильным приложения для обучения основ информатики.

Заключение

Во время выполнения бакалаврской работы было реализовано мобильное приложение для обучения студентов основам информатики на операционной системе Android. В мобильном приложении были реализованы функции главного экрана, окна вывода тестов, прохождения тестов, а также окно авторизации, панель администратора, где осуществлялись функции добавления, изменения и удаления тем и тестов.

В данной работе была изучена предметная область, определены цели и задачи. Также проведен сравнительный анализ похожих мобильных приложений из магазина Google Play, где были выделены преимущества и недостатки каждого приложения. Благодаря сравнительному анализу были определены функциональные требования к разрабатываемому мобильному приложению для обучения студентов основам информатики.

Для того, чтобы проверить правильности логического и функционального моделирования, были построены: UML-диаграмма вариантов, диаграмма последовательности, а также UML-диаграмма классов.

Для реализации мобильного приложения был выбран язык программирования Java, среда разработки Android Studio и для хранения информации в базы данных была выбрана система управления базой данных (СУБД) SQLite.

Разработанное мобильное приложение имеет большой функционал для обучения, а также для управления контентом приложения с помощью панели администратора. Также данное мобильное приложение имеет перспективы дальнейшего развития.

Список используемой литературы и используемых источников

1. Березовская Ю.В., Юфрякова О.А., Вологдина В.Г. и др. Введение в разработку приложений для ОС Android. - М.: НОУ "ИНТУИТ", 2016.-434 с.
2. Варакин М.В. Разработка мобильных приложений под Android. УЦ «Специалист» при МГТУ им. Н. Э. Баумана, 2012.
3. Кронин Д, Купер А., Рейман Р. Алан Купер об интерфейсе. Основы проектирования взаимодействия СПб.: Символ'Плюс, 2009. – 688 с., ил.
4. МакГрат, М. Программирование на Java для начинающих / М. МакГрат. - М.: Эксмо, 2016. - 192 с. ISBN 978-5-699-85743-2
5. Минимально жизнеспособный продукт. [Электронный ресурс] / Режим доступа: URL: https://ru.wikipedia.org/wiki/Минимально_жизнеспособный_продукт (дата обращения 11.05.2022)
6. Работа с базами данных SQLite в Android. [Электронный ресурс]: / Режим доступа: URL: http://developer.alexanderklimov.ru/android/sqlite/android-sqlite.php (дата обращения 28.04.2022).
7. Разработка мобильного приложения на платформе Android для ООО “Первая типография». [Электронный ресурс] / Режим доступа: URL: https://dspace.tltsu.ru/bitstream/123456789/4109/1/Тырсенко%20С.А._ПИб-1301.pdf (дата обращения 14.04.2022).
8. Разработка мобильного приложения для волонтерского центра [Электронный ресурс]: / Режим доступа: URL: https://dspace.tltsu.ru/bitstream/123456789/4120/1/Янин%20Р.О._ПИб-1301.pdf (дата обращения 15.04.2022).
9. Создаём развивающее приложение при помощи Android Studio. [Электронный ресурс] / Режим доступа: URL: https://habr.com/ru/post/322008/ (дата обращения 16.04.2022).
10. Android. Программирование для профессионалов. Билл Филлипс, К. Стюарт, Кристин Марсикано. Питер, 2017, ISBN: 978-5-4461-0413-0.

11. Android для разработчиков, Дейтел П., Дейтел Х., Уолд А., СПб.: Питер. 2016. ISBN: 978-5-496-02371-9, 978-0134289366.

12. Android. Сборник рецептов. Задачи и решения для разработчиков приложений. Дарвин Ян Ф. 2018, ISBN: 9785990944602, 9781449374433.

13. Android Programming: The Big Nerd Ranch Guide. By Bill Phillips, Chris Stewart & Kristin Marsicano. Big Nerd Ranch Guides; 3 edition February 9, 2017; 624 pages.

14. Android Studio. [Электронный ресурс] / Режим доступа: URL: https://ru.wikipedia.org/wiki/Android_Studio (дата обращения 15.04.2022)

15. Download Android Studio and SDK tools. [Электронный ресурс] / Режим доступа: URL: https://developer.android.com/studio/intro (дата обращения 10.04.2022).

16. Head First Android Development 2e: A Brain-Friendly Guide. Dawn Griffiths, David Griffiths. ISBN 9781449362188.

17. Java 8: руководство для начинающих, 6-е изд.: Пер. с англ. - М. ООО "И.Д. Вильяме", 2015. - 720 с.: ил. - Парал. тит. Анг.

18. Model-View-Presenter. [Электронный ресурс] / Режим доступа: URL: https://ru.wikipedia.org/wiki/Model-View-Presenter (дата обращения 15.05.2022)

19. RecyclerView. [Электронный ресурс] / Режим доступа: URL: https://metanit.com/java/android/5.11.php (дата обращения 02.05.2022).

20. SQLite. Общая информация [Электронный ресурс] / Режим доступа: URL: https://ru.wikipedia.org/wiki/SQLite (дата обращения 18.04.2022)

Приложение А

Фрагмент кода класса MainActivity

```
package com.infoapp.activity;

import android.annotation.SuppressLint;
import android.app.Activity;
import android.content.Context;
import android.content.Intent;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.util.Log;
import android.view.View;

import androidx.appcompat.widget.Toolbar;
import androidx.recyclerview.widget.GridLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import com.infoapp.R;
import com.infoapp.adapters.CategoryAdapter;
import com.infoapp.listeners.ListItemClickListener;
import com.infoapp.models.test.CategoryModel;
import com.infoapp.utilities.ActivityUtilities;
import com.infoapp.utilities.AppUtilities;
import com.mikepenz.materialdrawer.AccountHeader;
import com.mikepenz.materialdrawer.AccountHeaderBuilder;
import com.mikepenz.materialdrawer.Drawer;
import com.mikepenz.materialdrawer.DrawerBuilder;
import com.mikepenz.materialdrawer.model.DividerDrawerItem;
import com.mikepenz.materialdrawer.model.PrimaryDrawerItem;
import com.mikepenz.materialdrawer.model.ProfileDrawerItem;
import com.mikepenz.materialdrawer.model.SecondaryDrawerItem;
import com.mikepenz.materialdrawer.model.interfaces.IDrawerItem;
import com.mikepenz.materialdrawer.model.interfaces.IProfile;

import java.util.ArrayList;

public class MainActivity extends BaseActivity {

    private Activity activity;
    private Context context;

    private Toolbar toolbar; // объявляем переменную для нашего тулбара

    private AccountHeader header = null; // и его элементов
    private Drawer drawer = null;

    private ArrayList<CategoryModel> categoryList; // массив элементов
CategoryModel
    private CategoryAdapter adapter = null;
    private RecyclerView recyclerView;

    private String[] ThemesName = null; // массив элеметов
    DBHelper dbHelper;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main); // инициализация главного
```

```
окна
        toolbar = (Toolbar) findViewById(R.id.toolbar1); // инициализация
тулбара
        setSupportActionBar(toolbar);

        activity = MainActivity.this;
        context = getApplicationContext();

        recyclerView = (RecyclerView) findViewById(R.id.rvContent);
        // корневому макету списка присваевам GridLayout, который нужен для
вывода элементов в виде сетки (2)
        recyclerView.setLayoutManager(new GridLayoutManager(activity, 2,
GridLayoutManager.VERTICAL, false));

        categoryList = new ArrayList<>();
        adapter = new CategoryAdapter(context, activity, categoryList);
        recyclerView.setAdapter(adapter);

        initLoader();

        dbHelper = new DBHelper(this);
        //final SQLiteDatabase database = dbHelper.getWritableDatabase();

        loadThemes();

        final IProfile profile = new
ProfileDrawerItem().withIcon(R.drawable.ic_dev); // профиль пользователя

        header = new AccountHeaderBuilder()
                .withActivity(this) // привязываем к нашему активити
                .withTranslucentStatusBar(true) // устанавливаем прозрачность
                .withHeaderBackground(R.drawable.header) // фон
                .withOnAccountHeaderProfileImageListener(new
AccountHeader.OnAccountHeaderProfileImageListener() { // слушатель клика по
профилю пользователя
                    @Override
                    public boolean onProfileImageClick(View view, IProfile
profile, boolean current) {
                        // кастом url открытие сайта

ActivityUtilities.getInstance().invokeCustomUrlActivity(activity,
CustomUrlActivity.class,
                                getResources().getString(R.string.site),
getResources().getString(R.string.site_url), false);
                        return false;
                    }

                    @Override
                    public boolean onProfileImageLongClick(View view,
IProfile profile, boolean current) {
                        return false;
                    }
                })
                .addProfiles(profile) // добавляем профиль пользователя
                .build();
        DrawerBuilder drawerBuilder = new DrawerBuilder();
        drawerBuilder.withActivity(this);
        drawerBuilder.withToolbar(toolbar);
        drawerBuilder.withHasStableIds(true);
        drawerBuilder.withAccountHeader(header);
        drawerBuilder.addDrawerItems(
```

```
                new
PrimaryDrawerItem().withName(R.string.infoThemes).withIcon(R.drawable.ic_dev)
.withIdentifier(1).withSelectable(false),
                new
PrimaryDrawerItem().withName(R.string.infoTest).withIcon(R.drawable.ic_dev).w
ithIdentifier(10).withSelectable(false),
                new
PrimaryDrawerItem().withName(R.string.infoResult).withIcon(R.drawable.ic_dev)
.withIdentifier(11).withSelectable(false),
                new
PrimaryDrawerItem().withName(R.string.about_dev).withIcon(R.drawable.ic_dev).
withIdentifier(12).withSelectable(false),

                new DividerDrawerItem(), // разделительная линия
                new
SecondaryDrawerItem().withName(R.string.di_youtube).withIcon(R.drawable.ic_yo
utube).withIdentifier(20).withSelectable(false),
                new
SecondaryDrawerItem().withName(R.string.di_twitter).withIcon(R.drawable.ic_tw
itter).withIdentifier(22).withSelectable(false),

                new DividerDrawerItem(), // разделительная линия
                new
SecondaryDrawerItem().withName("Настройки").withIcon(R.drawable.ic_settings).
withIdentifier(30).withSelectable(false),
                new           SecondaryDrawerItem().withName("Пользовательское
соглашение").withIcon(R.drawable.ic_privacy_policy).withIdentifier(31).withSe
lectable(false),

                new DividerDrawerItem(), // разделительная линия
                new
SecondaryDrawerItem().withName("Выход").withIcon(R.drawable.ic_exit).withIden
tifier(40).withSelectable(false)

        );
        drawerBuilder.withOnDrawerItemClickListener(new
Drawer.OnDrawerItemClickListener() { // слушатель нажатия пунктов меню
            @Override
            public boolean onItemClick(View view, int position, IDrawerItem
drawerItem) { // вызываем соответствующие окна
                // обработка нажатии пуктов меню в панели навигации
                if (drawerItem != null) {
                    Intent intent = null;
                    if (drawerItem.getIdentifier() == 10) {

ActivityUtilities.getInstance().invokeNewActivity(activity,    TestMenu.class,
false);
                    } else if (drawerItem.getIdentifier() == 11) {

ActivityUtilities.getInstance().invokeNewActivity(activity,     TestAdd.class,
false);
                    } else if (drawerItem.getIdentifier() == 12) {

ActivityUtilities.getInstance().invokeNewActivity(activity,
AboutDevActivity.class, false);
                    } else if (drawerItem.getIdentifier() == 20) {
                        AppUtilities.youtubeLink(activity);
                    } else if (drawerItem.getIdentifier() == 21) {
                        AppUtilities.youtubeLink(activity);
                    } else if (drawerItem.getIdentifier() == 22) {
                        AppUtilities.youtubeLink(activity);
```

```
                    } else if (drawerItem.getIdentifier() == 23) {
                        AppUtilities.youtubeLink(activity);
                    } else if (drawerItem.getIdentifier() == 30) {
                        // TODO : добавить окно настройки приложения
                    } else if (drawerItem.getIdentifier() == 31) {

ActivityUtilities.getInstance().invokeCustomUrlActivity(activity,
CustomUrlActivity.class,
                                getResources().getString(R.string.privacy),
getResources().getString(R.string.privacy_url), false);
                    } else if (drawerItem.getIdentifier() == 40) {

                    }
                }

                return false;
            }
        });
        drawerBuilder.withSavedInstance(savedInstanceState);
        drawerBuilder.withShowDrawerOnFirstLaunch(true);// привязываем панель
к активити
        // header
        // сохраняем состояние drawer
        // принудительно открываем при запуске
        drawer = drawerBuilder
                // определяет отображение drawer при свайпе
                //.withShowDrawerUntilDraggedOpened(true)
                .build();


    }
    // метод обработки нажатия кнопки назад
    public void onBackPressed() {
        if (drawer != null && drawer.isDrawerOpen()) {
            drawer.closeDrawer();
        } else {
            AppUtilities.tapPromtToExit(this);
        }
    }
```

```
    // метод для выгрузки данных из бд
    @SuppressLint("Range")
    private void loadThemes () {
        // открываем подключение
        final SQLiteDatabase database = dbHelper.getReadableDatabase();

        //Создаем курсор для вывода данных
        Cursor cursor=database.rawQuery("select *from " + DBHelper.TABLE_BOOK
+ " where id >= 0", null);
        cursor.moveToFirst();

        ThemesName = new String [cursor.getCount()];

        Log.d("dd","records = " + cursor.getCount());
        int i=0;
        do{
            try {
                ThemesName[i]                                                =
cursor.getString(cursor.getColumnIndex(DBHelper.HEADLINE));
                categoryList.add(new        CategoryModel(Integer.toString(i),
ThemesName[i])); // добавляем в категори лист данные
                i++;
            }catch (Exception e){
                e.printStackTrace();
            }


        }while (cursor.moveToNext());
        hideLoader(); // после выгрузки информации скрываем анимацию
        adapter.notifyDataSetChanged();  // информируем апатер об изменении
данных
        // recycler list item click listener
        adapter.setItemClickListener(new ListItemClickListener() {
            @Override
            public void onItemClick(int position, View view) {

                CategoryModel model = categoryList.get(position);
                Intent intent = new Intent(activity, ThemesView.class);
                intent.putExtra("topic_name", ThemesName[position]);
                activity.startActivity(intent);
            }});}}
```