Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
институт
Информационные системы
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой ИС
_____ Л.С. Троценко
подпись инициалы, фамилия
« 13 » июня 2018 г.

БАКАЛАВРСКАЯ РАБОТА

230400.62 «Информационные системы и технологии»
специальность

Разработка мобильного приложения онлайн-сервиса для музыкантов
тема

Руководитель _____ _____ Е.А. Мальцев
подпись, дата должность, ученая степень инициалы, фамилия

Выпускник _____ М.А. Королёв
подпись, дата инициалы, фамилия

Нормоконтролер _____ _____ Ю.В. Шмагрис
подпись, дата должность, ученая степень инициалы, фамилия

Красноярск 2018

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
институт
Информационные системы
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой ИС
_____ С.А. Виденин
подпись инициалы, фамилия
« 2 » марта 2018 г.

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
в форме бакалаврской работы

Студенту _____Королёву Михаилу Александровичу_____
фамилия, имя, отчество

Группа ЗКИ13-13б Направление (специальность) _____09.03.02_____
номер код

_____Информационные системы и технологии_____
наименование

Тема выпускной квалификационной работы Разработка мобильного приложения онлайн-сервиса для музыкантов_____

Утверждена приказом по университету № 3758/с от 14.03.2018_____

Руководитель ВКР _____Мальцев Е.А. консультант к.т.н., доцент кафедры систем искусственного интеллекта
инициалы, фамилия, должность, ученое звание и место работы

Исходные данные для ВКР: Теоретические сведения о способах проектирования информационных систем, о разработке программного обеспечения для мобильной платформы Android_____

Перечень разделов ВКР: Общие сведения; Описание проектных решений и реализация системы_____
_____
_____

Перечень графического материала Презентация, выполненная в Microsoft: PowerPoint 2016_____
_____
_____
_____

Руководитель ВКР _____ Е.А. Мальцев
подпись инициалы и фамилия

Задание принял к исполнению _____ М.А. Королев
подпись, инициалы и фамилия студента

« 2 » марта 2018 г.

РЕФЕРАТ

Выпускная квалификационная работа по теме «Разработка мобильного приложения онлайн-сервиса для музыкантов» содержит 42 страницы текстового документа, 17 использованных источников.

МОБИЛЬНЫЕ ПРИЛОЖЕНИЯ, РАЗРАБОТКА ПРИЛОЖЕНИЙ, МУЗЫКА, ОНЛАЙН-СЕРВИСЫ.

Объект исследования – работа онлайн-сервиса для музыкантов.

Предмет исследования – разработка мобильного приложения, позволяющего пользователям взаимодействовать с онлайн-сервисом.

Цели:

– сокращение времени доступа пользователей к онлайн-сервису;
– разграничение прав доступа к музыкальным произведениям;
– повышение мобильности и удобства использования онлайн-сервиса.

При исследовании онлайн-сервиса для музыкантов, была определена потребность разработки мобильного приложения, позволяющего эффективно взаимодействовать и предоставлять сервис пользователю быстрее и эффективнее, чем другие платформы для взаимодействия с онлайн-сервисом.

Разобранная по шагам работа реализованного приложения дает понимание функциональности разработанного мобильного приложения. Исходя из этого, главная цель выпускной квалификационной работы была выполнена.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В настоящее время существует множество онлайн-сервисов, деятельность которых ориентирована на взаимодействие с пользователями. Количество пользователей каждый день увеличивается, в связи с этим качество сервиса зависит не только от качества работы самого онлайн-сервиса, но также от возможностей и мобильности использования такого сервиса.

Количество пользователей перестает возрастать, когда большая часть из новых пользователей сталкивается с проблемой использования сервиса на своём мобильном устройстве. В связи с этим, для конкурентного и эффективного роста сервиса им требуется разработка мобильного приложения.

К примеру, онлайн-сервис для музыкантов накапливает много информации об авторах музыкальных произведений и самих музыкальных произведениях. Получить доступ к этой информации пользователи традиционно могут только при помощи веб-версии онлайн-сервиса. Предоставив пользователям альтернативу в качестве мобильного приложения, можно значительно увеличить количество новых пользователей и повысить лояльность к сервису уже зарегистрированных пользователей.

Передавать большие объемы данных между мобильным приложением и серверной частью является неэффективным подходом. В связи с этим, данные будут автоматически подгружаться с серверной части онлайн-сервиса, а уже загруженные данные предлагается хранить в собственном хранилище мобильного приложения.

С целью корректного использования больших объемов хранимых данных требуются программные средства, позволяющие как модификацию хранимых данных, ввод запросов, чтение файлов, добавление новой информации, а также принимающие решения исходя из имеющихся данных. С целью реализации данных требований созданы СУБД (система управления базами данных). В современном понимании СУБД – это системы, специализирующиеся на

управлении массивом информации одним пользователем, или множеством одновременно работающих пользователей.

Актуальность данной выпускной работы обуславливается тем, что современные онлайн-сервисы прогрессируют и для прироста новых пользователей стабильно требуется повышение качества сервиса, а также его мобильности.

Цель данной выпускной квалификационной работы – реализовать мобильное приложение онлайн-сервиса для музыкантов.

Главные требования, которые нужно учесть при реализации мобильного приложения онлайн-сервиса для музыкантов:

1. Уменьшение времени на доступ к информации;
2. Присутствие диалоговых программных средств;
3. Наличие собственного хранилища информации, полученной с серверной части онлайн-сервиса.

Установленная цель указывает на задачи работы:

1. Анализ предметной области;
2. Выявление процессов, требующих реализации в мобильном приложении;
3. Формирование технического задания на разработку мобильного приложения;
4. Обеспечение ведения и поддержки базы данных;
5. Создание удобного интерфейса, интуитивно-понятных связей между диалогами ввода информации.

Объектом исследования – функционирование онлайн-сервиса для музыкантов.

Предмет исследования – процесс взаимодействия пользователей с онлайн-сервисом для музыкантов.

Выпускная квалификационная работа содержит в себе введение, две главы основной части, выводы (заключения), и список использованных источников.

Первая глава предоставляет информацию об общих вопросах, связанных с теорией в мобильных приложениях, моделях информации, также в данной главе показана характеристика предприятия и разработано техническое задание на проектирование мобильного приложения.

Вторая глаза предоставляет информацию об описании среды разработки, также данная глава содержит описание реализуемого мобильного приложения.

1 Общие сведения

1.1 Мобильное приложение

В настоящее время операционная система Android является, по-видимому, самой популярной платформой для мобильных устройств.

Многообразие и широкое распространение смартфонов и планшетов различных производителей, функционирующих под управлением данной платформы, стимулирует рост рынка мобильных приложений, делая навыки разработки под Android весьма востребованными в современном мире.

1.1.1 Android SDK

Android SDK является набором инструментов, позволяющих реализовывать приложения для платформы Android. Он содержит библиотеки, которые позволяют разработчику API платформы Android, утилиты для создания и сборки приложений, менеджмента образов виртуальных устройств и реализации других различных команд. Android SDK можно получить бесплатно для всех основных платформ и может быть загружен с сайта Google.

Android SDK не включает компилятор языка программирования Java. В связи с этим, для осуществления компиляции Android-приложения нужен также набор инструментов Java Development Kit (JDK).

Для упрощения разработки может быть использована одна из интегрированных сред (Intergrated Development Environment, IDE). В настоящее время поддержка разработки под Android имеется во всех основных средах разработки для Java, в том числе IntelliJ IDEA, NetBeans и Eclipse. Следует отметить, что наличие IDE не является обязательным для разработки, поскольку все необходимые для сборки и развёртывания приложения операции могут быть выполнены средствами командной строки.

1.1.2 Менеджер пакетов Android SDK

Android SDK поддерживает разработку под все официальные версии платформы Android с использованием большого количества внешних библиотек, включая многие библиотеки сторонних разработчиков. Для управления пакетами, обеспечивающими разработку с использованием данных библиотек, используется утилита Android. Её можно найти в подкаталоге «tools» внутри каталога SDK. Утилита Android поддерживает как интерфейс командной строки, так и графический интерфейс пользователя.

С целью менеджмента библиотек, предлагается использовать графическую оболочку, которую можно открыть при помощи запуска утилиты Android без параметров. После того, как запуск оболочки произведен, на экране отобразится перечень пакетов, в который будут включены пакеты уже установленные в системе, а также пакеты, доступные для установки.

При желании установить требуемые пакеты, их можно указать в общем перечне, после чего воспользоваться кнопкой «Install packages». После того, как лицензия будет подтверждена пользователем, система начнет скачивать указанные пакеты из Интернет, после чего осуществит их установку. Пакеты в перечне сгруппированы по версиям платформы Android. Каждая из версий платформы обладает двойной нумерацией: «коммерческую» и «внутреннюю»: например, Android 4.0.3 соответствует API 15. Первый тип нумерации используется для обозначения поддержки возможностей платформы устройствами на рынке, тогда как в процессе разработки приложений повсеместно используемым является второй тип нумерации.

Каждая группа в перечне включает такие составляющие как: основной набор API для разработки приложений под данную платформу (SDK Platform), примеры приложений (Samples for SDK), документация на API (Documentation for Android SDK), исходные тексты библиотек платформы (Sources for Android SDK), образы виртуальных устройств для различных архитектур (ARM EABI v7a System Image, Intel x86 Atom System Image и другие).

Помимо этого, перечень может содержать библиотеки сторонних разработчиков, предназначенные для определённого класса устройств.

Для старта программирования нужно установить основной набор библиотек (SDK Platform), образы виртуальных устройств (System Images) для конкретной платформы, а также общий набор инструментов, не привязанный к версии API (Android SDK Tools, Android SDK Platform-tools в группе Tools). Остальные пакеты являются необязательными.

1.1.3 Сборка проекта

Сборка Android-проекта может производиться различными способами. В простейшем случае используется утилита «Ant», входящая в JDK и являющаяся стандартным средством сборки Java-проектов.

Сборочный файл «Ant» обычно называется «build.xml» и располагается в корневом каталоге проекта. Если создавать проект командой из п. 1.3, то данный файл также будет создан.

В сборочном файле определяются цели, каждая из которых соответствует некоторой операции (например, компиляция, сборка, развёртывание, тестирование проекта). Операции состоят из команд, выполнение которых приводит к достижению указанной цели.

Между целями могут быть установлены зависимости, при этом команды, необходимые для достижения некоторой цели, выполняются только после того, как выполнены команды зависимых целей.

При этом поддерживается достаточно гибкий механизм, позволяющий не выполнять некоторые команды, если они уже были выполнены ранее и их повторное выполнение даст тот же результат (например, компиляция модуля не запускается повторно, если этот модуль уже был скомпилирован ранее и его исходный текст не изменялся).

1.1.4 Компоненты Android-приложения

Типичное Android-приложение состоит из компонентов, которые могут относиться к следующим типам:

– сервис (service) — фоновый процесс;

– слушатель широковещательных сообщений (broadcast receiver) – обработчик некоторого глобального события в операционной системе (например, выключение экрана, низкий заряд батареи и т.д.).

– активность (activity) — компонент, осуществляющий взаимодействие с пользователем;

– провайдер контента (content provider) – компонент, осуществляющий предоставление доступа к данным, находящимся в некотором хранилище;

Данные компоненты являются достаточно независимыми друг от друга и имеют чётко определённые интерфейсы для взаимодействия с другими компонентами.

Особенность Android-приложений заключается в том, что компоненты различных приложений могут взаимодействовать между собой. Например, в случае, когда приложению необходимо отправить сообщение по электронной почте, оно может вызвать стандартную активность с функционалом почтового клиента, причём после завершения этой активности пользователь вернётся к работе с исходным приложением.

И наоборот: программист может разработать собственный почтовый клиент и зарегистрировать его в системе при установке приложения, тем самым разрешив другим приложениям его использование.

Эта особенность несколько размывает само понятие приложения и заставляет рассматривать всю платформу как открытую систему, компоненты которой способны к кооперативному поведению.

Взаимодействие компонентов осуществляется посредством отправки асинхронных сообщений. В приложении каждое из таких сообщений ассоциируется с сущностью, называемой «интент» (intent).

1.1.5 Интенты

В Android-приложениях интент — это объект, инкапсулирующий в себе запрос на выполнение некоторого действия. Интент может включать в себя следующие компоненты:

– URI, идентифицирующий данные, над которыми необходимо выполнить действие;

– набор категорий, позволяющих группировать действия;

– действие, которое необходимо выполнить (обязательный компонент);

– дополнительные параметры (extras), необходимые для выполнения действия.

Следует отметить, что интент, как правило, не содержит в явном виде указания адресата отправляемого сообщения. Вместо этого указывается действие, которое необходимо выполнить. Каждый компонент системы во время установки регистрирует некоторый набор интентов, которые он способен выполнять.

В момент активации соответствующего интента, система осуществляет поиск компонента, который способен выполнить данное действие. После этого система передает управление такому компоненту. В случае, если система найдет несколько таких компонентов, пользователю будет предоставлен выбор.

1.1.6 Жизненный цикл активности

Активности являются короткоживущими компонентами в архитектуре Android-приложения.

Это означает, что они имеют свойство автоматически создаваться и уничтожаться в разных случаях. К примеру, когда меняется ориентация экрана, либо происходит нехватка памяти для других приложений. Помимо этого, активность может переходить в фон или на передний план, принимать и терять

фокус. Для правильной обработки всех этих ситуаций вводится понятие жизненного цикла активности и предоставляются средства, позволяющие выполнять те или иные действия при переходе между различными фазами этого жизненного цикла.

Жизненный цикл активности включает в себя следующие основные состояния:

– paused – активность находится на переднем плане, но не в фокусе, т. е. перекрыта всплывающим окном или окном диалога; пользователь не может непосредственно взаимодействовать с такой активностью;

– running/resumed – активность находится на переднем плане и в фокусе; в этом состоянии пользователь может непосредственно взаимодействовать с приложением посредством графического интерфейса;

– stopped – активность находится в фоне и не отображается на экране; пользователь не может взаимодействовать с такой активностью.

Переходы между перечисленными состояниями осуществляются по событиям, инициированным пользователем (например, переключением на другую активность), или системным событиям (например, в связи с нехваткой памяти). Каждый переход сопровождается вызовом определённых методов в классе Activity, от которого обязаны наследоваться любые пользовательские активности. В свою очередь, в классах-наследниках указанные методы могут переопределяться для того, чтобы должным образом отреагировать на переход между состояниями жизненного цикла. Возможные переходы вместе с соответствующими callback-методами изображены на рисунке 1.

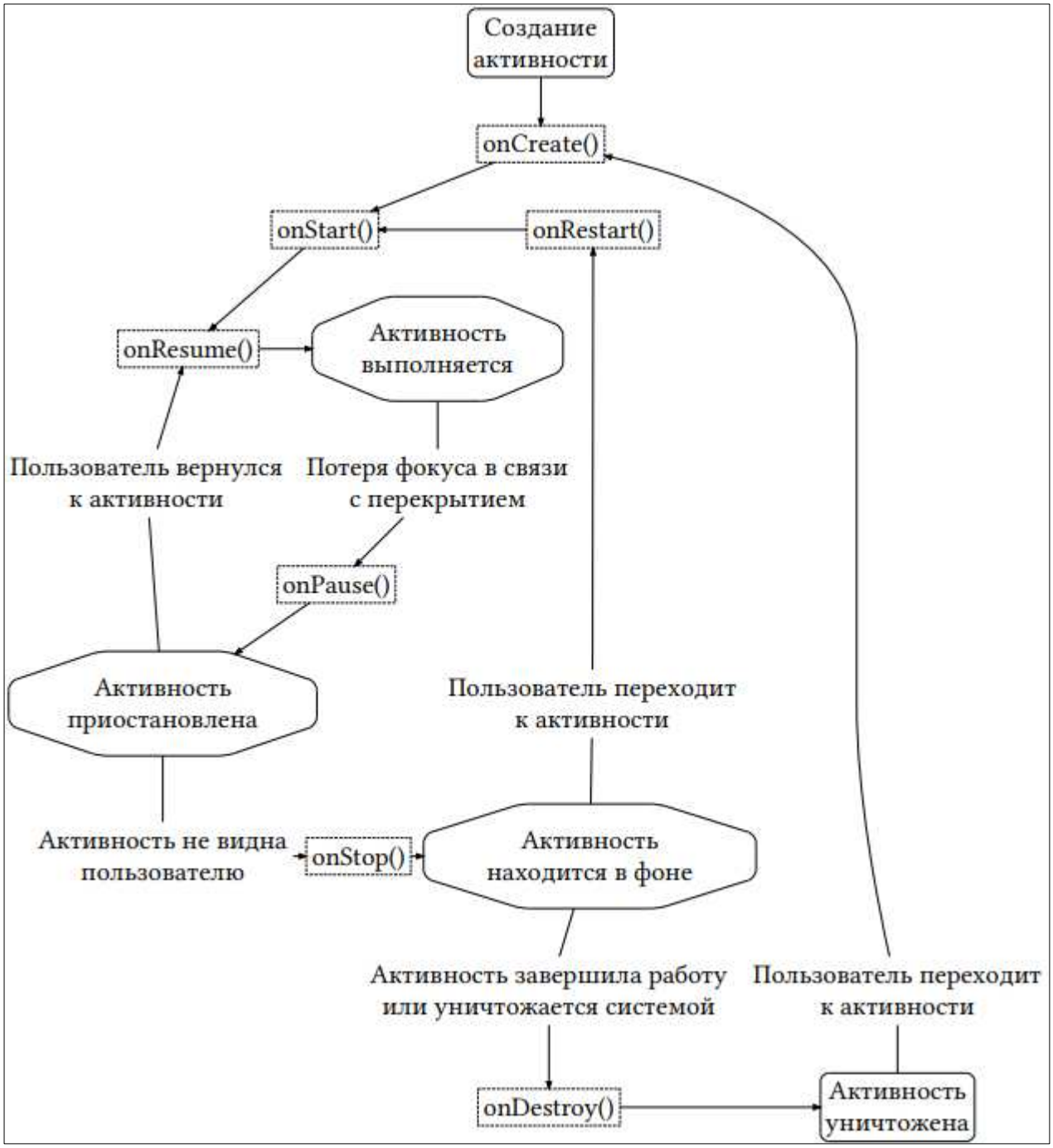

Рисунок 1 – Жизненный цикл активностей в Android

Метод onCreate() вызывается непосредственно после создания активности. Здесь осуществляется инициализация интерфейса пользователя, привязка данных к элементам интерфейса, создание потоков и т.д.

Парным к данному методу является метод onDestroy(), в котором необходимо освободить ресурсы, полученные при вызове onCreate().

Метод onPause() вызывается, когда активность теряет фокус в связи с перекрытием её экрана всплывающим окном, диалогом или другой

активностью. После возврата из данного метода активность перейдёт в состояние paused.

Активность, которая обладает данным состоянием, может быть уничтожена операционной системой когда угодно в том случае, если системе будет недостаточно оперативной памяти и имеются другие более высокоприоритетные приложения. Исходя из этого, метод onPause() должен предусматривать сохранение состояния активности для того, чтобы быть в состоянии восстановить его при перезапуске. Восстановление состояния осуществляется в callback-методе onResume().

В случае, если разработчик изменит системную конфигурацию приложения, которая содержит в себе смену локализации или поворот экрана, это приведет к уничтожению активности и в дальнейшем к её повторному созданию. При этом выполняются все callback-методы, включая onDestroy() для уничтожаемой и onStart() для вновь создаваемой активности. Для того чтобы отличить рассматриваемую ситуацию от ситуации, когда приложение закрыто пользователем, а затем вновь открыто, предусмотрена пара методов onSaveInstanceState() и onRestoreInstanceState(). Реализация этих методов в классе Activity автоматически сохраняет и восстанавливает состояние всех элементов пользовательского интерфейса. Однако в некоторых сложных случаях может потребоваться переопределение и этих методов для обеспечения корректного функционирования приложения при изменении конфигурации.

1.1.7 Задачи и стек активностей

Под задачей (task) в Android подразумевается набор активностей, вызываемых друг из друга и направленных на удовлетворение одной потребности пользователя. Список всех выполняемых на устройстве задач отображается, когда пользователь нажимает и удерживает кнопку «Home».

Когда пользователь запускает приложение, создаётся новая задача и первая открывшаяся активность запущенного приложения помещается в стек

активностей этой задачи. Относительно задачи эта активность называется корневой. Задача существует до тех пор, пока корневая активность не завершится.

Корневая активность может вызвать вторую активность, которая будет помещена в стек текущей задачи поверх корневой. В свою очередь, вторая активность может вызывать третью и т. д. Все эти активности помещаются в стек. При закрытии активности, находящейся на вершине стека (по нажатию кнопки «Back» на Android-устройстве или программно с помощью вызова соответствующего метода), она удаляется из стека, а управление передаётся той активности, которая находилась в стеке непосредственно под удалённой.

При нажатии кнопки «Home» текущая активность переходит в фоновый режим, однако весь стек активностей соответствующей задачи сохраняется. Если теперь нажать и удерживать кнопку «Home», то пользователь увидит список активных задач. При выборе любой из них активность, находящаяся на вершине стека, получит фокус, а все остальные активности будут сохраняться в стеке, пока находящиеся выше их активности не будут закрыты.

Следует отметить, что активности, входящие в стек некоторой задачи, могут входить в состав различных приложений. В этом выражается одна из важных концепций Android, декларирующая кооперацию нескольких различных приложений для удовлетворения потребностей пользователя и предоставляющая механизм интентов для осуществления этой кооперации.

1.1.8 Архитектура «модель-вид-контроллер»

Приложения для Android, как правило, разрабатываются в соответствии с архитектурным шаблоном «модель-вид-контроллер» (model-view-controller, MVC). Этот шаблон позволяет разделять отдельные слои ответственности приложения таким образом, чтобы упростить поддержку кода и облегчить внесение изменений.

Моделью в терминологии MVC является уровень бизнес-логики, ответственный за хранение и обработку информации. Данный слой не должен делать никаких предположений относительно представления информации или взаимодействия пользователя с приложением.

Потенциально это делает классы модели пригодными для повторного применения, в том числе на платформах, отличных от той, на которой они были разработаны.

Вид – это уровень отображения. Он ответствен за отображение данных модели. Классы, принадлежащие данному уровню, как правило, являются виджетами платформы либо унаследованы от них. Вид может обращаться к модели для получения данных, однако модель не должна непосредственно зависеть от вида.

Контроллер является уровнем, ответственным за взаимодействие с пользователем. К данному уровню относятся обработчики событий пользователя и часто код, соединяющий все компоненты системы в единое целое (так называемый glue code). В зависимости от конкретной реализации MVC уровни вида и контроллера могут быть отдельными классами или совмещены. Контроллер, как правило, зависит от модели, однако, как и в случае вида, модель не должна хранить ссылок на конкретные классы контроллера.

1.1.9 Назначение класса View

Класс View является суперклассом всех классов-виджетов в Android, включая TextView, ImageView, Button и т.д. Каждый экземпляр класса View ответствен за отрисовку некоторой прямоугольной области на экране, а также за обработку событий, связанных с этой областью.

При разработке приложений под Android, как правило, используются готовые библиотечные субклассы класса View. Однако в некоторых случаях бывают необходимы компоненты, имеющие специфический внешний вид и

поведение. Подобные компоненты можно легко реализовать, унаследовав собственный класс от класса View и переопределив методы, ответственные за отрисовку и/или обработку событий.

Из всех событий, обрабатываемых классом View, наиболее важными представляются события, возникающие при взаимодействии пользователя с областью, за которую ответствен View, события передачи фокуса, нажатия на клавиши и отрисовки.

1.1.10 Правила обработки событий вдоль иерархии виджетов

Каждый виджет имеет возможность содержать другие виджеты в рамках области, за которую он ответственен. Иерархия включения может быть сформирована непосредственно в программном коде или путём вложения элементов в XML-файле, описывающем пользовательский интерфейс. Исходя из практики, второй способ используют чаще.

Иерархия виджетов имеет существенное значение для обработки событий. Действуют специальные правила, определяющие обработку событий вдоль иерархии:

– первоочередно, событие делегируется самому вложенному (листовому в дереве вложения) виджету, который ответственен за область, в рамках которой произошло событие;

– в случае, если виджет не смог определить собственного обработчика произошедшего события, событие делегируется для обработки родительскому в терминах иерархии включения виджету; в противном случае происходит вызов обработчика дочернего виджета;

– в случае, если обработчик события возвращает «false», то виджет обработал событие, но требуется также продолжить обработку этого события родительским виджетом;

– в случае, если обработчик события возвращает «true», то считается, что виджет обработал событие, и дальнейшая обработка не требуется.

Описанные выше правила – гибкие, что означает что их можно применять не только для переопределения варианта обработки событий во вложенных виджетах, но также и для связывания в последовательность обработчиков разных виджетов из иерархии.

1.1.11 Работа с ресурсами

Ресурсы в Android являются статическими данными. Как пример можно привести описание пользовательского интерфейса, изображения, текст. Ресурсы располагаются программистом в проекте в виде файлов и транслируются в apk-пакет приложения автоматически во время сборки.

Использование ресурсов преследует две основные цели.

1. Реализация вариативности ресурсов, которые используются, исходя из конфигурации. Ресурсы позволяют реализовать поддержку различных типов ориентации экрана и различных языков интерфейса пользователя без дополнительных затрат со стороны программиста, так как ресурсы, подходящие для текущей конфигурации системы, загружаются автоматически.

2. Отделение данных от кода. При использовании ресурсов данные хранятся отдельно от кода в декларативном виде, поэтому их легко изменять, причём изменения могут вносить не программисты, а, например, дизайнеры пользовательского интерфейса или переводчики. Изменение ресурсов не требует перекомпиляции приложения – необходима лишь его пересборка.

В Android выделяются следующие основные типы ресурсов:

– Drawable – файлы изображений, используемых приложением. Сюда также входят графические файлы, используемые в пользовательском интерфейсе;

– Raw – произвольные данные, как правило бинарные;

– Layout – файлы в формате XML, описывающие расположение элементов интерфейса пользователя;

– Values – данные приложения, представленные в текстовом формате XML. В первую очередь в состав ресурсов данного типа входят все текстовые строки приложения, традиционно размещаемые в файле strings.xml;

– Menu – файлы в формате XML, описывающие компоновку элементов меню или панели действий.

Ресурсы каждого типа размещаются внутри каталога «res» проекта в подкаталогах, названия которых совпадают с типами ресурсов, приведёнными выше, но написаны строчными буквами.

Для использования ресурсов из приложения в ходе сборки проекта (а при использовании инструментальных сред также при любом изменении ресурсов приложения) создаётся специальный файл R.java, содержащий идентификаторы всех ресурсов проекта. Все они определены как статические константы внутри подклассов класса R, каждый из которых соответствует своему типу ресурсов:

– R.drawable — ресурсы-изображения из каталога res/drawables;

– R.raw — прочие ресурсы из каталога res/raw;

– R.id – компоненты пользовательского интерфейса, описанные в файлах ресурсов (файлы каталога res/layout); элементы идентифицируются по идентификаторам, задаваемым с помощью атрибута android:id;

– R.layout – ресурсы из каталога res/layout, описывающие компоновку пользовательского интерфейса;

– R.menu – ресурсы из каталога res/menu, описывающие состав меню или панели действий;

– R.string, R.integer, R.boolean, R.color, R.array и т. д. — ресурсы из файлов данных (файлы каталога res/values), сгруппированные по типам этих данных;

1.1.12 Хранение данных

Многим приложениям требуется сохранять данные в постоянной памяти и восстанавливать их при последующих запусках.

Платформа Android предоставляет три возможности решения данной задачи: настройки (prefereneces), файловая система и базы данных. В текущем приложении будет использована база данных.

1.2 Модель данных

Согласно классической теории БД, модель данных это теория визуализации и расчета информации в СУБД, содержащая следующие факторы:

1. Фактор структуры, определяющий способы определения типов и логических схем информации в БД;
2. Фактор манипуляции, определяющий способы манипулирования информацией;
3. Фактор целостности, определяющий способы определения и поддержки целостности БД.

Фактор структуры реализует собой логическое представление базы данных, фактор манипуляции реализует варианты транслирования между статусами БД и варианты получения информации из БД, фактор целостности реализует инструменты определений правильных статусов БД.

Модель данных представляет собой логическое, самодостаточное, абстрактное понятие операторов, объектов и прочих элементов, вместе представляющих механизм доступа к информации. Такие объекты предоставляют возможность проектирования структуры информации.

Базы данных и системы управления базами данных основываются на определенной явной или неявной структуре информации. Системы управления базами данных, реализованные по одинаковой структуре информации, называют однотипными.

Среди специализированных материалов часто можно увидеть понятие «модель данных», которое используется эквивалентно понятию «схема БД». Многие специалисты указывают на то, что это равенство проводить некорректно. Эквивалентность между данными терминами можно сравнить с

эквивалентностью программного языка и программой, реализованной при помощи этого языка.

1.3 Системы управления базами данных

Oracle

Oracle Database или Oracle DBMS является объектно-реляционной системой управления базами данных (СУБД). Oracle может хранить и выполнять хранимые процедуры и функции внутри себя. PL / SQL (собственные процедурные расширения корпорации Oracle на SQL), или объектно-ориентированного языка Java может ссылаться на такие объекты, коды и предоставлять возможность программирования структур.

Microsoft SQL Server

Microsoft SQL Server является системой управления реляционными базами данных (СУБД), разработанной корпорацией Microsoft. Используется для от небольших и средних по размеру баз данных до крупных баз данных масштаба предприятия, конкурирует с другими СУБД в этом сегменте рынка.

MySQL

MySQL является свободной системой управления базами данных (СУБД). Является решением для малых и средних приложений. Обычно используется в качестве сервера, к которому обращаются локальные или удалённые клиенты, однако в дистрибутив входит библиотека внутреннего сервера, позволяющая включать MySQL в автономные программы.

PostgreSQL

PostgreSQL (произносится «Пост-Грес-Кью-Эл» или просто «постгрес») является свободной объектно-реляционной системой управления базами данных (СУБД). Является свободной альтернативой коммерческим СУБД (таким как Oracle Database, Microsoft SQL Server, Informix и СУБД производства Sybase) вместе с другими свободными СУБД (такими как MySQL и Firebird).

Sybase ASA

Сервер управления базами данных "Sybase SQL Anywhere Studio" (сокращенно ASA) сейчас является одним из наиболее стремительно развивающихся продуктов компании iAnywhere Solution, дочерней компании холдинга Sybase.

FireBird

Firebird (FirebirdSQL) является компактной, кроссплатформенной, свободной системой управления базами данных (СУБД), работающей на GNU/Linux, Microsoft Windows и разнообразных Unix платформах. Это коммерчески независимый проект C и C++ программистов, технических советников и разработчиков мультиплатформенных систем управления базами данных, основанный на исходном коде, выпущенном корпорацией Borland 25 июля 2000 года в виде свободной версии Interbase 6.0.

InterBase

InterBase является системой управления базами данных, разработанной компанией Borland. Основными достоинствами последней версии InterBase являются низкие требования к системе, с одновременной масштабируемостью на несколько процессоров, плюс развитая система мониторинга, временные таблицы, встраиваемая аутентификация пользователей, журналирование.

1.4 Техническое задание на проектирование

1.4.1 Общие сведения

Полное наименование программного продукта: Мобильное приложение онлайн-сервиса для музыкантов;

Условное обозначение: Мобильное приложение для музыкантов;

Заказчик: ООО «Онлайн-Мьюзик»;

Исполнитель: Королёв М.А.;

Плановая дата начала выполнения работ: 16.02.2018;

Плановая дата окончания выполнения работ: 07.05.2018.

1.4.2 Назначение и цели создания программного продукта

Назначение программного продукта:

«Мобильное приложение онлайн-сервиса для музыкантов» реализовано для повышения скорости и эффективности взаимодействия пользователей онлайн-сервиса с данным сервисом. Основными пользователями онлайн-сервиса являются музыканты.

Целью данного мобильного приложения является:

1) обеспечение сбора и хранения данных об авторах;
2) увеличение скорости и производительности взаимодействия пользователей с сервисом;
3) возможность подбора нот музыкального произведения;
4) обеспечение возможности оценивания музыкальных произведений;
5) реализация возможности отправки музыкальных произведений.

1.4.3 Функциональные требования

Необходимо, чтобы разработанное мобильное приложение полностью отвечало заявленным пользовательским и функциональным требованиям.

Бизнес-требования:

Реализованное мобильное приложение должно предоставить повышение производительности и мобильности взаимодействия пользователей с онлайн-сервисов для музыкантов.

Пользовательские требования:

Составляющие мобильного приложения должны быть понятны для пользователей.

Для роли музыканта необходимо реализовать и сделать доступным модуль ведения базы авторов, модуль ведения базы музыкальных произведений (создание, загрузка, редактирование, удаление музыкальных произведений), подбор нот музыкального произведения, отправка музыкальных произведений, оценивание музыкальных произведений.

На рисунке 2 приведена реализованная диаграмма вариантов использования.

С помощью диаграммы вариантов использования отражено, какие действия мобильное приложение будет в состоянии произвести, т.е. функциональное назначение данного программного продукта.

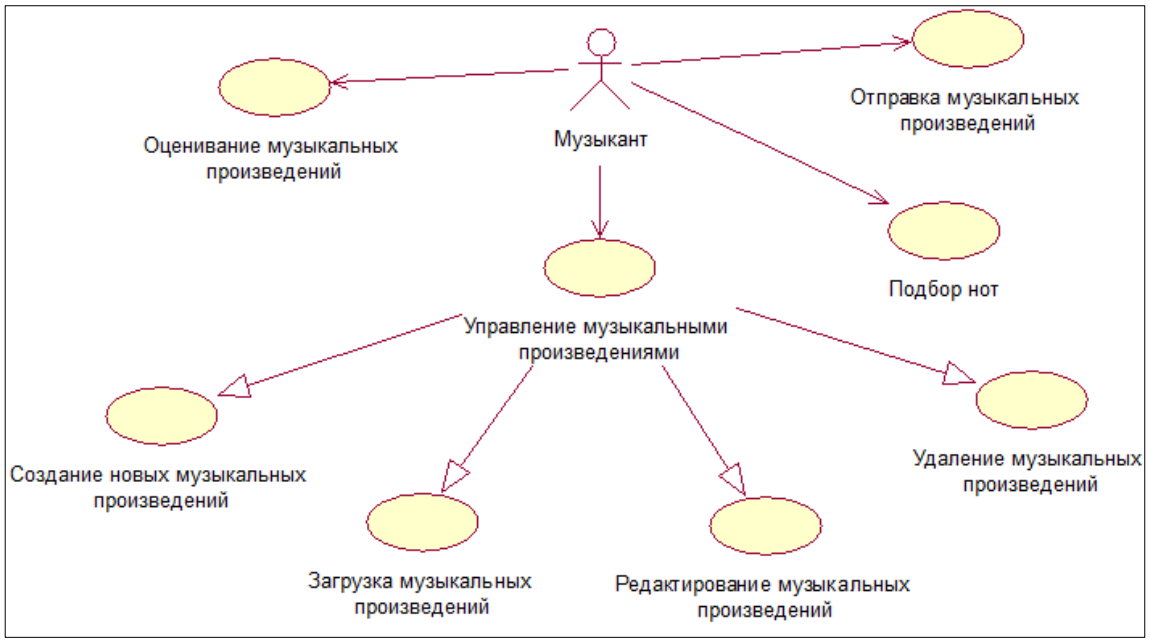

Рисунок 2 – Диаграмма вариантов использования

1.4.4 Нефункциональные требования

Интерфейс разработанного мобильного приложения, отвечающий за взаимодействие с пользователем должен быть интуитивно понятным и быстрым для обучения.

В реализованном мобильном приложении должны быть следующие основные модули: модуль ведения базы авторов, модуль работы с музыкальными произведениями.

Для работы в приложении предполагается организовать одну роль пользователя: музыкант. Данная роль будет взаимодействовать с музыкальными произведениями, и с самим онлайн-сервисом.

На рисунке 3 приведена разработанная диаграмма классов.

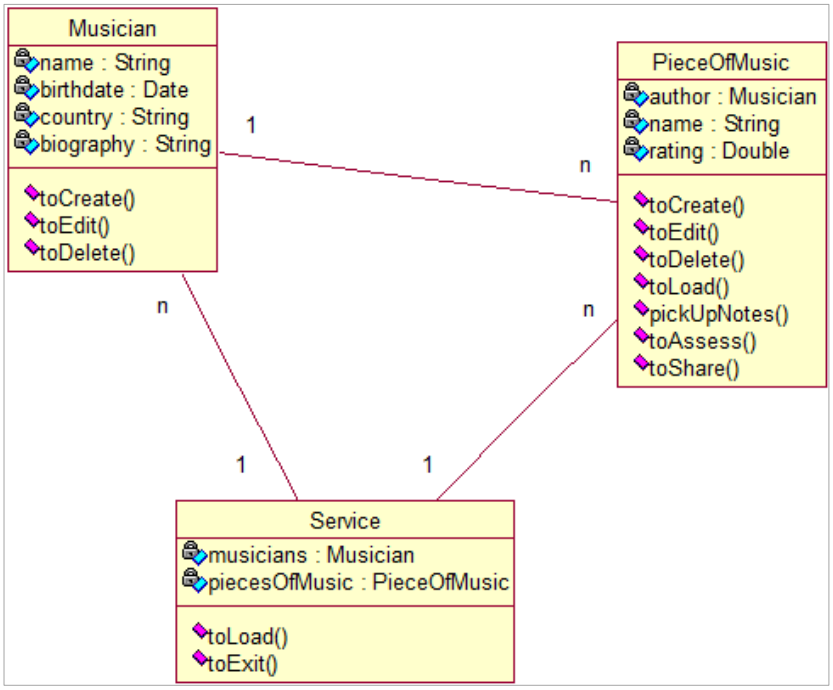

Рисунок 3 – Диаграмма классов

1.4.5 Системные ограничения

Необходимо, чтобы разработанное мобильное приложение могло работать на аппаратном обеспечении большинства пользователей, и для работы не нуждалась в высокой вычислительной мощности.

Системные требования:

– Операционная система Android 4.0

– Процессор Qualcomm Snapdragon 625 1 ГГц и выше;

– Оперативная память 512 Мбайт и выше (рекомендуется 768 Мбайт);

– Жесткий диск (при использовании необходимо около 50 Мбайт);

– TFT-дисплей (рекомендуется 4 дюймовый дисплей с разрешением 1024*768).

Требуется, чтобы в реализованном мобильном приложении не существовало несоответствий с другими приложениями, установленными на мобильном устройстве.

1.4.6 Атрибуты качества

В разработанном мобильном приложении должны присутствовать указанные атрибуты качества:

1. Отказоустойчивость, что означает способность приложения давать возможность дальнейшей работы с ним в случае возникновения нештатных ситуаций;

2. Масштабируемость, что означает способность приложения давать возможность увеличения эффективности и количества процессоров, объемов внешней и оперативной памяти, а также других параметров устройства;

3. Совместимость, что означает способность приложения не быть требовательным к варианту организации приложения и его архитектуре;

4. Производительность, что означает способность приложения исполнять поставленные задачи эффективно и без использования большого количества вычислительных ресурсов;

5. Доступность, что означает способность приложения демонстрировать длительное время непрерывной работы;

6. Надежность, что означает способность приложения быть устойчивым к появлению нештатных ситуаций при помощи уменьшения количества сбоев и отказов.

2 Описание проектных решений и реализация мобильного приложения

2.1 Обоснование выбора среды разработки

Стремительное появление инновационных технологий, связанных с обработкой информации, а также увеличение области применения таких технологий привели к бурному росту количества программных продуктов. Темпы роста качества и количества программных продуктов демонстрируют, что изменение трат на создание таких продуктов, или их покупку постепенно растет на сумму около 20% в год.

Термин «программное обеспечение программного продукта» означает комплекс документальных и программных инструментов для реализации и использования систем при помощи мобильных устройств.

В соответствии с функциями, которые реализует программное обеспечение, его подразделяют на 2 группы:

1) прикладное программное обеспечение (реализует решение определенных задач и целиком структурирует вычислительный процесс информационной системы.

2) базовое программное обеспечение (структурирует процесс обработки данных в вычислительной машине и реализует корректную среду для исполнения программ).

Базовое программное обеспечение содержит:

- операционные системы;
- сервисные программы;
- трансляторы языков программирования;
- программы технического обслуживания.

Взаимодействие между аппаратной частью мобильного устройства и пользователей реализуют операционные системы, которые управляют процессом взаимодействия с информацией. Автоматизация процессов ввода и

вывода данных является важной функций операционной системы, а также менеджмент исполнения внутренних задач. Операционная система помещает программный продукт в память мобильного устройства, после чего производит наблюдение за процессом исполнения данной программы. Она также производит анализ ситуаций, которые мешают корректным вычислениям, после чего решает, какие действия необходимо предпринять.

В связи с этим, операционные системы подразделяют на:

1) сетевые;
2) однозадачные (для одного пользователя);
3) многозадачные (для многих пользователей).

В качестве основной операционной системы определена операционная система Android, разработанная компанией Google. Данная операционная система позволит реализовать корректное исполнение мобильного приложения онлайн-сервиса для музыкантов.

2.2 Среда программирования 1С

Для реализации программного продукта выбрана мобильная версия платформы «1С:Предприятие 8». Выбранная платформа «1С:Предприятие 8» - это многозадачная система, предназначенная для накопления и анализа информации. В связи с тем, что процессы сервиса могут динамически изменяться, платформа «1С:Предприятие 8» может адаптироваться с учетом индивидуальных требований онлайн-сервиса. Эту способность также называют «конфигурируемость», что означает способность доработки программного продукта под изменившиеся процессы сервиса.

Такая возможность реализуется благодаря тому, что мобильная версия платформы «1С:Предприятие 8» является не только программным продуктом, но также комплексом разных компонентов, которые можно использовать для индивидуальной настройки.

На текущий момент более миллиона компаний используют для работы компоненты платформы «1С:Предприятие 8».

Внутри мобильной версии платформы «1С:Предприятие 8» существует удобный и эффективный интерфейс, позволяющий пользователям комфортно и длительно выполнять работу.

Мобильная версия системы «1С:Предприятие 8» реализует множество способов исполнения программного решения. Среди таких способов как использования в режиме приложения на мобильном устройстве, так и персональное использование в качестве информационной системы на персональном компьютере.

Платформа «1С:Предприятие 8» определяется как открытая. Это означает, что система способна интегрироваться с любым программным и аппаратным обеспечением.

Модуль прав доступа реализует разрешение или запрещение работы пользователей с определенным блоком или модулем программного продукта.

Мобильная платформа «1С:Предприятие 8» обладает следующими преимуществами:

1. Предусмотрено использование различных систем хранения данных;
2. Многоплатформенность: приложение может исполняться под любыми операционными системами, для которых разработана платформа, такими как Android или iOS;
3. Интеграция с любыми программами и оборудованиями, которые поддерживают общепризнанные протоколы передачи данных и стандарты;
4. Удобное администрирование;
5. Отказоустойчивость, обеспечивается бесперебойность работы пользователей во время возникновения сбоев аппаратного или программного характера за счёт резервирования кластера, резервирования рабочих процессов и устойчивости к разрыву связи.

2.3 Описание реализованного мобильного приложения

При запуске мобильного приложения пользователь видит окно следующего вида (рисунок 4):

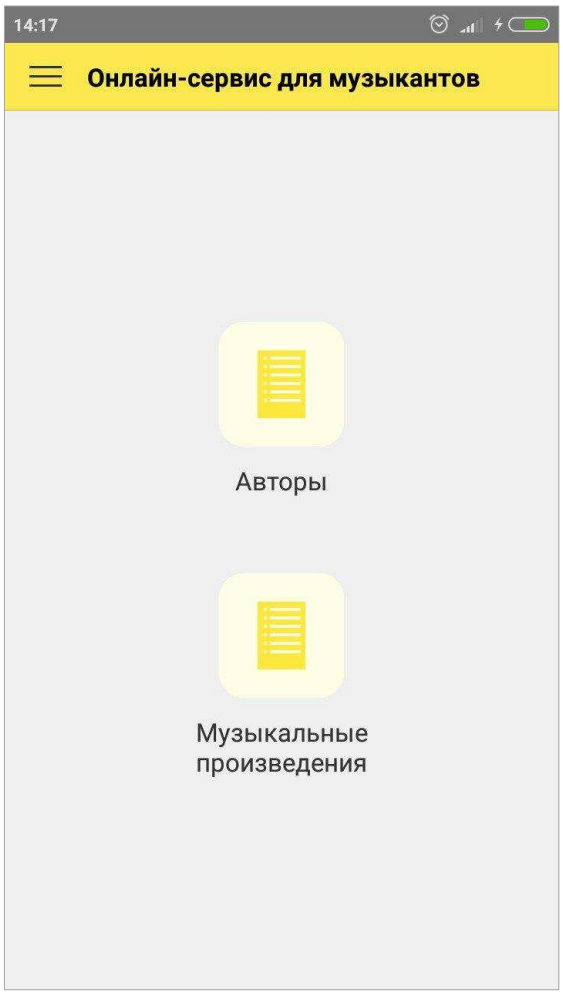

Рисунок 4 – Главное окно приложения

При работе с приложением, пользователь может перейти в режим работы с авторами, либо в режим работы с музыкальными произведениями. В данном окне пользователю предлагается сделать соответствующий выбор.

При нажатии на кнопку «Авторы» пользователь переходит в режим работы с авторами. В данном режиме пользователю открывается окно с базой авторов, из которых он может выбрать уже существующего автора или создать нового (рисунок 5).

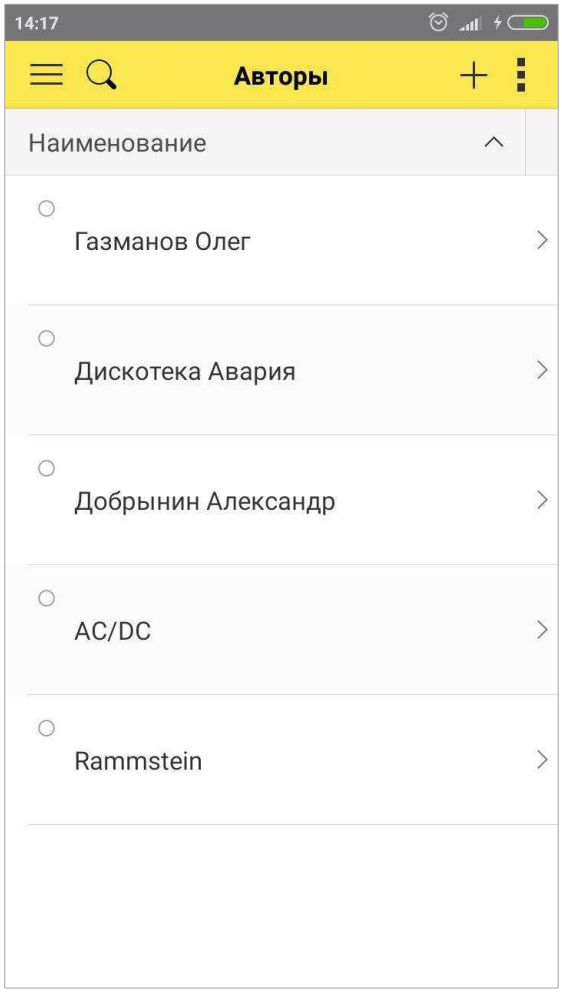

Рисунок 5 – Режим работы с авторами

При нажатии на существующего автора пользователь перейдет в окно карточки автора, в котором он увидит все поля автора с заполненными данными в них (рисунок 6). Также при желании пользователь может осуществить поиск среди существующих авторов по одному из полей.

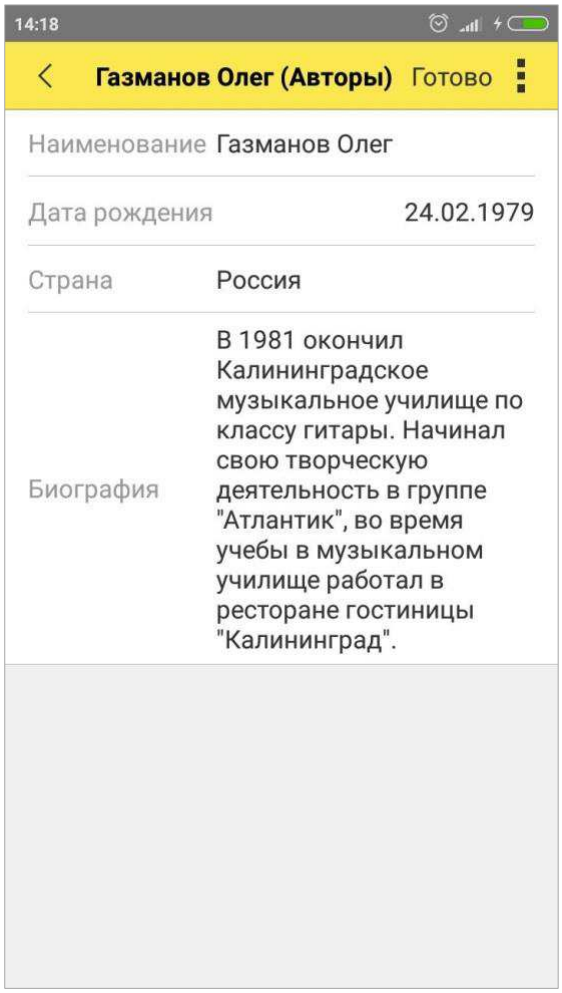

Рисунок 6 – Карточка существующего автора

При нажатии на кнопку «Музыкальные произведения» в главном окне приложения, пользователь перейдет в режим работы с музыкальными произведениями. В данном режиме пользователь увидит список музыкальных произведений в виде авторов и названий каждого из них (рисунок 7).

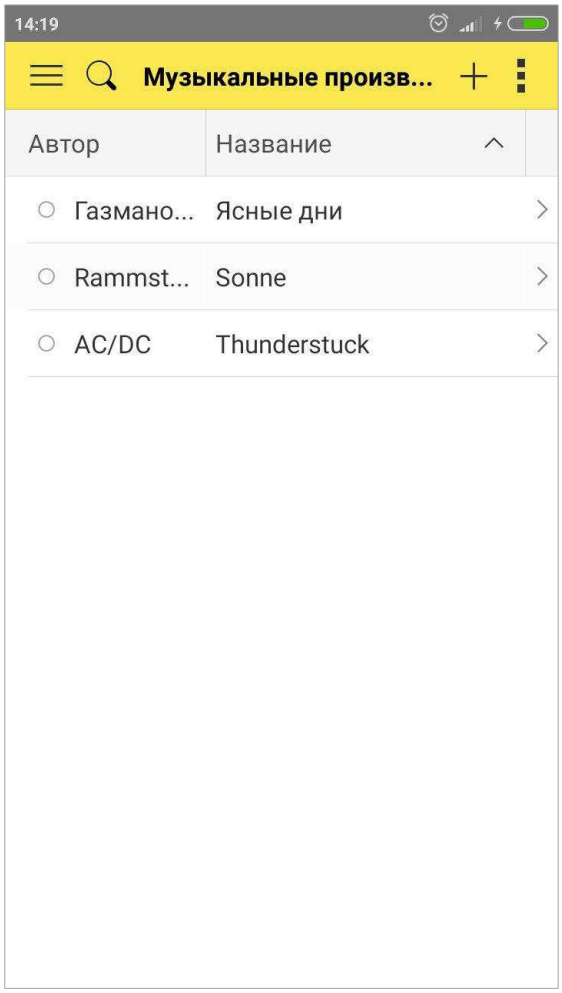

Рисунок 7 – Режим работы с музыкальными произведениями

При выборе какого-либо из музыкальных произведений, пользователю откроется карточка музыкального произведения. В данной карточке пользователь сможет увидеть автора, название и среднюю оценку по данному музыкальному произведению (рисунок 8).

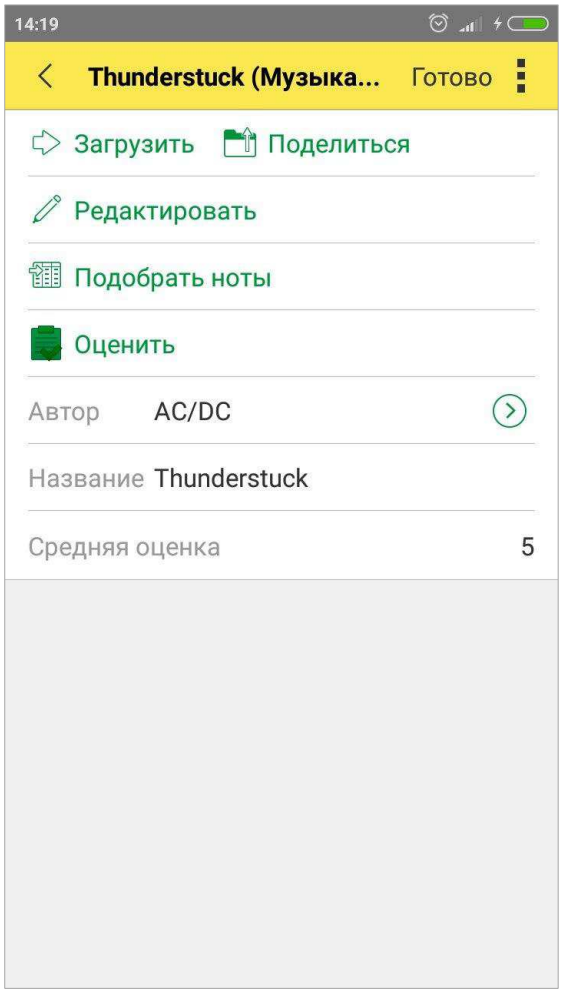

Рисунок 8 – Работа с определенным музыкальным произведением

Воспользовавшись данной формой, пользователь может произвести следующие операции с музыкальным произведением с помощью соответствующих кнопок:

– Загрузить музыкальное произведение в текущую карточку;

– Поделиться музыкальным произведением с другими пользователями;

– Отредактировать музыкальное произведение в текущей карточке;

– Подобрать ноты для музыкального произведения;

– Оценить музыкальное произведение.

При выполнении любой из этих операций, мобильное приложение будет обращаться к серверной части онлайн-сервиса. Таким образом, для исполнения

данных функций, приложению будет необходимо, чтобы мобильное устройство было подключено к Интернет.

ЗАКЛЮЧЕНИЕ

Целью данной выпускной квалификационной работы являлась разработка мобильного приложения для повышения скорости и эффективности взаимодействия пользователей с онлайн-сервисом для музыкантов. В процессе реализации выпускной квалификационной работы разработан программный продукт с использованием современного инструмента разработки мобильных приложений на базе платформы «1С: Предприятие 8».

В процессе разработки программного продукта проанализированы процессы взаимодействия пользователей с онлайн-сервисом, приобретен навык проектирования БД и работы с ними, изучен язык программирования для разработки мобильных приложений на базе платформы «1С:Предприятие 8».

С целью выполнения поставленной цели были решены следующие задачи:

– анализ предметной области;

– выявление процессов онлайн-сервиса, требующих реализации в мобильном приложении;

– формирование технического задания на разработку мобильного приложения;

– обеспечение ведения и поддержки базы данных;

– создание удобного интерфейса, интуитивно-понятных связей между диалогами ввода информации.

С помощью реализованного продукта пользователи онлайн-сервиса смогут использовать сервис на своих мобильных устройствах, что увеличит величину охвата аудитории потенциальных пользователей.

Это приведет к увеличению потока регистраций новых пользователей, усилению активности уже зарегистрированных пользователей, а также в целом к повышению лояльности пользователей к онлайн-сервису.

По сути, использование онлайн-сервиса теперь не ограничивается только полноценными компьютерами или ноутбуками, пользователи имеют возможность получать все услуги более мобильно и не менее полноценно, чем пользователи веб-версии на компьютере.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Тельнов, Ю.Ф. Информационные системы и технологии: Научное издание. / Ю.Ф. Тельнов. – Москва: ЮНИТИ, 2016. – 303 с.
2. Парамонов, И.В. Разработка мобильных приложений для платформы Android: учебное пособие / И.В. Парамонов; Ярославский государственный университет им. П.Г. Демидова. – Ярославль: ЯрГУ, 2013. – 88 с.
3. Федотова, Е.Л. Информационные технологии и системы: Учебное пособие / Е.Л. Федотова. – Москва: ИД ФОРУМ, НИЦ ИНФРА, 2013. – 352 с.
4. Ульман, Д. Основы реляционных баз данных / Д. Ульман, Д. Уид: Издательство Лори, 2006. – 384 с.
5. Рубичев, Н.А. Измерительные информационные системы / Н.А. Рубичев. – Москва: Дрофа, 2010. – 334 с.
6. Кашаев, С. Программирование в «1С:Предприятие 8.3». / С. Кашаев – Санкт-Петербург: Питер, 2014. – 451 с.
7. Бородакий, Ю.В. Информационные технологии. Методы, процессы, системы / Ю.В. Бородакий, Ю.Г. Лободинский. – Москва: ГЛТ, 2004. – 456 с.
8. Радченко, М. Архитектура и работа с данными «1С:Предприятия 8.2» / М. Радченко, Е. Хрусталева – Москва: 1С-Паблишинг, 2011. – 268 с.
9. Миков, А.И. Информационные процессы и нормативные системы в IT: Математические модели. Проблемы проектирования. Новые подходы / А.И. Миков. – Москва: КД Либроком, 2013. – 256 с.
10. Иванова, Г.С. Технология программирования: учеб. для вузов / Г.С. Иванова. - Москва: Издательство МГТУ им. Н. Э. Баумана, 2002. – 319 с.
11. Хрусталева, Е. Разработка сложных отчетов в «1С:Предприятии 8». Система компоновки данных / Е. Хрусталева. – Москва: 1С-Паблишинг, 2012. – 458 с.

12. Федорова, Г. Информационные системы. / Г. Федорова – Москва: Academia, 2013. – 208 с.

13. Норенков, И.П. Автоматизированные информационные системы: Учебное пособие / И.П. Норенков. - Москва: МГТУ им.Баумана, 2011. - 342 с.

14. Пирогов, В. Информационные системы и базы данных: организация и проектирование. / В. Пирогов. – Санкт-Петербург: БХВ-Петербург, 2005. – 492 с.

15. Голицина, О. Информационные системы и технологии. / О. Голицина, И. Попов, Н. Максимов – Москва: Инфра-М, 2014. – 400 с.

16. Чаадаев, В.К. Информационные системы компаний связи. Создание и внедрение / В.К. Чаадаев. – Москва: Эко-Трендз, 2004. - 256 с.

17. СТО 4.2–07–2014 Система менеджмента качества. Общие требования к построению, изложению и оформлению документов учебной деятельности. – Введ. 30.12.2013. – Красноярск: ИПК СФУ, 2014. – 60 с.

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
институт
Информационные системы
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой ИС
_____ Л.С. Троценко
подпись инициалы, фамилия
« 13 » июня 2018 г.

БАКАЛАВРСКАЯ РАБОТА

230400.62 «Информационные системы и технологии»
специальность

Разработка мобильного приложения онлайн-сервиса для музыкантов
тема

Руководитель _____ _____ Е.А. Мальцев
подпись, дата должность, ученая степень инициалы, фамилия

Выпускник _____ М.А. Королёв
подпись, дата инициалы, фамилия

Нормоконтролер _____ _____ Ю.В. Шмагрис
подпись, дата должность, ученая степень инициалы, фамилия

Красноярск 2018

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
институт
Информационные системы
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой ИС
_____ С.А. Виденин
подпись инициалы, фамилия
« 2 » марта 2018 г.

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
в форме бакалаврской работы

Студенту _____Королёву Михаилу Александровичу_____
фамилия, имя, отчество

Группа ЗКИ13-13б Направление (специальность) _____09.03.02_____
номер код

_____Информационные системы и технологии_____
наименование

Тема выпускной квалификационной работы Разработка мобильного приложения онлайн-сервиса для музыкантов_____

Утверждена приказом по университету № 3758/с от 14.03.2018_____

Руководитель ВКР _____Мальцев Е.А. консультант к.т.н., доцент кафедры систем искусственного интеллекта
инициалы, фамилия, должность, ученое звание и место работы

Исходные данные для ВКР: Теоретические сведения о способах проектирования информационных систем, о разработке программного обеспечения для мобильной платформы Android_____

Перечень разделов ВКР: Общие сведения; Описание проектных решений и реализация системы_____
_____
_____

Перечень графического материала Презентация, выполненная в Microsoft: PowerPoint 2016_____
_____
_____
_____

Руководитель ВКР _____ Е.А. Мальцев
подпись инициалы и фамилия

Задание принял к исполнению _____ М.А. Королев
подпись, инициалы и фамилия студента

« 2 » марта 2018 г.

РЕФЕРАТ

Выпускная квалификационная работа по теме «Разработка мобильного приложения онлайн-сервиса для музыкантов» содержит 42 страницы текстового документа, 17 использованных источников.

МОБИЛЬНЫЕ ПРИЛОЖЕНИЯ, РАЗРАБОТКА ПРИЛОЖЕНИЙ, МУЗЫКА, ОНЛАЙН-СЕРВИСЫ.

Объект исследования – работа онлайн-сервиса для музыкантов.

Предмет исследования – разработка мобильного приложения, позволяющего пользователям взаимодействовать с онлайн-сервисом.

Цели:

– сокращение времени доступа пользователей к онлайн-сервису;
– разграничение прав доступа к музыкальным произведениям;
– повышение мобильности и удобства использования онлайн-сервиса.

При исследовании онлайн-сервиса для музыкантов, была определена потребность разработки мобильного приложения, позволяющего эффективно взаимодействовать и предоставлять сервис пользователю быстрее и эффективнее, чем другие платформы для взаимодействия с онлайн-сервисом.

Разобранная по шагам работа реализованного приложения дает понимание функциональности разработанного мобильного приложения. Исходя из этого, главная цель выпускной квалификационной работы была выполнена.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В настоящее время существует множество онлайн-сервисов, деятельность которых ориентирована на взаимодействие с пользователями. Количество пользователей каждый день увеличивается, в связи с этим качество сервиса зависит не только от качества работы самого онлайн-сервиса, но также от возможностей и мобильности использования такого сервиса.

Количество пользователей перестает возрастать, когда большая часть из новых пользователей сталкивается с проблемой использования сервиса на своём мобильном устройстве. В связи с этим, для конкурентного и эффективного роста сервиса им требуется разработка мобильного приложения.

К примеру, онлайн-сервис для музыкантов накапливает много информации об авторах музыкальных произведений и самих музыкальных произведениях. Получить доступ к этой информации пользователи традиционно могут только при помощи веб-версии онлайн-сервиса. Предоставив пользователям альтернативу в качестве мобильного приложения, можно значительно увеличить количество новых пользователей и повысить лояльность к сервису уже зарегистрированных пользователей.

Передавать большие объемы данных между мобильным приложением и серверной частью является неэффективным подходом. В связи с этим, данные будут автоматически подгружаться с серверной части онлайн-сервиса, а уже загруженные данные предлагается хранить в собственном хранилище мобильного приложения.

С целью корректного использования больших объемов хранимых данных требуются программные средства, позволяющие как модификацию хранимых данных, ввод запросов, чтение файлов, добавление новой информации, а также принимающие решения исходя из имеющихся данных. С целью реализации данных требований созданы СУБД (система управления базами данных). В современном понимании СУБД – это системы, специализирующиеся на

управлении массивом информации одним пользователем, или множеством одновременно работающих пользователей.

Актуальность данной выпускной работы обуславливается тем, что современные онлайн-сервисы прогрессируют и для прироста новых пользователей стабильно требуется повышение качества сервиса, а также его мобильности.

Цель данной выпускной квалификационной работы – реализовать мобильное приложение онлайн-сервиса для музыкантов.

Главные требования, которые нужно учесть при реализации мобильного приложения онлайн-сервиса для музыкантов:

1. Уменьшение времени на доступ к информации;
2. Присутствие диалоговых программных средств;
3. Наличие собственного хранилища информации, полученной с серверной части онлайн-сервиса.

Установленная цель указывает на задачи работы:

1. Анализ предметной области;
2. Выявление процессов, требующих реализации в мобильном приложении;
3. Формирование технического задания на разработку мобильного приложения;
4. Обеспечение ведения и поддержки базы данных;
5. Создание удобного интерфейса, интуитивно-понятных связей между диалогами ввода информации.

Объектом исследования – функционирование онлайн-сервиса для музыкантов.

Предмет исследования – процесс взаимодействия пользователей с онлайн-сервисом для музыкантов.

Выпускная квалификационная работа содержит в себе введение, две главы основной части, выводы (заключения), и список использованных источников.

Первая глава предоставляет информацию об общих вопросах, связанных с теорией в мобильных приложениях, моделях информации, также в данной главе показана характеристика предприятия и разработано техническое задание на проектирование мобильного приложения.

Вторая глаза предоставляет информацию об описании среды разработки, также данная глава содержит описание реализуемого мобильного приложения.

1 Общие сведения

1.1 Мобильное приложение

В настоящее время операционная система Android является, по-видимому, самой популярной платформой для мобильных устройств.

Многообразие и широкое распространение смартфонов и планшетов различных производителей, функционирующих под управлением данной платформы, стимулирует рост рынка мобильных приложений, делая навыки разработки под Android весьма востребованными в современном мире.

1.1.1 Android SDK

Android SDK является набором инструментов, позволяющих реализовывать приложения для платформы Android. Он содержит библиотеки, которые позволяют разработчику API платформы Android, утилиты для создания и сборки приложений, менеджмента образов виртуальных устройств и реализации других различных команд. Android SDK можно получить бесплатно для всех основных платформ и может быть загружен с сайта Google.

Android SDK не включает компилятор языка программирования Java. В связи с этим, для осуществления компиляции Android-приложения нужен также набор инструментов Java Development Kit (JDK).

Для упрощения разработки может быть использована одна из интегрированных сред (Intergrated Development Environment, IDE). В настоящее время поддержка разработки под Android имеется во всех основных средах разработки для Java, в том числе IntelliJ IDEA, NetBeans и Eclipse. Следует отметить, что наличие IDE не является обязательным для разработки, поскольку все необходимые для сборки и развёртывания приложения операции могут быть выполнены средствами командной строки.

1.1.2 Менеджер пакетов Android SDK

Android SDK поддерживает разработку под все официальные версии платформы Android с использованием большого количества внешних библиотек, включая многие библиотеки сторонних разработчиков. Для управления пакетами, обеспечивающими разработку с использованием данных библиотек, используется утилита Android. Её можно найти в подкаталоге «tools» внутри каталога SDK. Утилита Android поддерживает как интерфейс командной строки, так и графический интерфейс пользователя.

С целью менеджмента библиотек, предлагается использовать графическую оболочку, которую можно открыть при помощи запуска утилиты Android без параметров. После того, как запуск оболочки произведен, на экране отобразится перечень пакетов, в который будут включены пакеты уже установленные в системе, а также пакеты, доступные для установки.

При желании установить требуемые пакеты, их можно указать в общем перечне, после чего воспользоваться кнопкой «Install packages». После того, как лицензия будет подтверждена пользователем, система начнет скачивать указанные пакеты из Интернет, после чего осуществит их установку. Пакеты в перечне сгруппированы по версиям платформы Android. Каждая из версий платформы обладает двойной нумерацией: «коммерческую» и «внутреннюю»: например, Android 4.0.3 соответствует API 15. Первый тип нумерации используется для обозначения поддержки возможностей платформы устройствами на рынке, тогда как в процессе разработки приложений повсеместно используемым является второй тип нумерации.

Каждая группа в перечне включает такие составляющие как: основной набор API для разработки приложений под данную платформу (SDK Platform), примеры приложений (Samples for SDK), документация на API (Documentation for Android SDK), исходные тексты библиотек платформы (Sources for Android SDK), образы виртуальных устройств для различных архитектур (ARM EABI v7a System Image, Intel x86 Atom System Image и другие).

Помимо этого, перечень может содержать библиотеки сторонних разработчиков, предназначенные для определённого класса устройств.

Для старта программирования нужно установить основной набор библиотек (SDK Platform), образы виртуальных устройств (System Images) для конкретной платформы, а также общий набор инструментов, не привязанный к версии API (Android SDK Tools, Android SDK Platform-tools в группе Tools). Остальные пакеты являются необязательными.

1.1.3 Сборка проекта

Сборка Android-проекта может производиться различными способами. В простейшем случае используется утилита «Ant», входящая в JDK и являющаяся стандартным средством сборки Java-проектов.

Сборочный файл «Ant» обычно называется «build.xml» и располагается в корневом каталоге проекта. Если создавать проект командой из п. 1.3, то данный файл также будет создан.

В сборочном файле определяются цели, каждая из которых соответствует некоторой операции (например, компиляция, сборка, развёртывание, тестирование проекта). Операции состоят из команд, выполнение которых приводит к достижению указанной цели.

Между целями могут быть установлены зависимости, при этом команды, необходимые для достижения некоторой цели, выполняются только после того, как выполнены команды зависимых целей.

При этом поддерживается достаточно гибкий механизм, позволяющий не выполнять некоторые команды, если они уже были выполнены ранее и их повторное выполнение даст тот же результат (например, компиляция модуля не запускается повторно, если этот модуль уже был скомпилирован ранее и его исходный текст не изменялся).

1.1.4 Компоненты Android-приложения

Типичное Android-приложение состоит из компонентов, которые могут относиться к следующим типам:

– сервис (service) — фоновый процесс;

– слушатель широковещательных сообщений (broadcast receiver) – обработчик некоторого глобального события в операционной системе (например, выключение экрана, низкий заряд батареи и т.д.).

– активность (activity) — компонент, осуществляющий взаимодействие с пользователем;

– провайдер контента (content provider) – компонент, осуществляющий предоставление доступа к данным, находящимся в некотором хранилище;

Данные компоненты являются достаточно независимыми друг от друга и имеют чётко определённые интерфейсы для взаимодействия с другими компонентами.

Особенность Android-приложений заключается в том, что компоненты различных приложений могут взаимодействовать между собой. Например, в случае, когда приложению необходимо отправить сообщение по электронной почте, оно может вызвать стандартную активность с функционалом почтового клиента, причём после завершения этой активности пользователь вернётся к работе с исходным приложением.

И наоборот: программист может разработать собственный почтовый клиент и зарегистрировать его в системе при установке приложения, тем самым разрешив другим приложениям его использование.

Эта особенность несколько размывает само понятие приложения и заставляет рассматривать всю платформу как открытую систему, компоненты которой способны к кооперативному поведению.

Взаимодействие компонентов осуществляется посредством отправки асинхронных сообщений. В приложении каждое из таких сообщений ассоциируется с сущностью, называемой «интент» (intent).

1.1.5 Интенты

В Android-приложениях интент — это объект, инкапсулирующий в себе запрос на выполнение некоторого действия. Интент может включать в себя следующие компоненты:

– URI, идентифицирующий данные, над которыми необходимо выполнить действие;

– набор категорий, позволяющих группировать действия;

– действие, которое необходимо выполнить (обязательный компонент);

– дополнительные параметры (extras), необходимые для выполнения действия.

Следует отметить, что интент, как правило, не содержит в явном виде указания адресата отправляемого сообщения. Вместо этого указывается действие, которое необходимо выполнить. Каждый компонент системы во время установки регистрирует некоторый набор интентов, которые он способен выполнять.

В момент активации соответствующего интента, система осуществляет поиск компонента, который способен выполнить данное действие. После этого система передает управление такому компоненту. В случае, если система найдет несколько таких компонентов, пользователю будет предоставлен выбор.

1.1.6 Жизненный цикл активности

Активности являются короткоживущими компонентами в архитектуре Android-приложения.

Это означает, что они имеют свойство автоматически создаваться и уничтожаться в разных случаях. К примеру, когда меняется ориентация экрана, либо происходит нехватка памяти для других приложений. Помимо этого, активность может переходить в фон или на передний план, принимать и терять

фокус. Для правильной обработки всех этих ситуаций вводится понятие жизненного цикла активности и предоставляются средства, позволяющие выполнять те или иные действия при переходе между различными фазами этого жизненного цикла.

Жизненный цикл активности включает в себя следующие основные состояния:

– paused – активность находится на переднем плане, но не в фокусе, т. е. перекрыта всплывающим окном или окном диалога; пользователь не может непосредственно взаимодействовать с такой активностью;

– running/resumed – активность находится на переднем плане и в фокусе; в этом состоянии пользователь может непосредственно взаимодействовать с приложением посредством графического интерфейса;

– stopped – активность находится в фоне и не отображается на экране; пользователь не может взаимодействовать с такой активностью.

Переходы между перечисленными состояниями осуществляются по событиям, инициированным пользователем (например, переключением на другую активность), или системным событиям (например, в связи с нехваткой памяти). Каждый переход сопровождается вызовом определённых методов в классе Activity, от которого обязаны наследоваться любые пользовательские активности. В свою очередь, в классах-наследниках указанные методы могут переопределяться для того, чтобы должным образом отреагировать на переход между состояниями жизненного цикла. Возможные переходы вместе с соответствующими callback-методами изображены на рисунке 1.

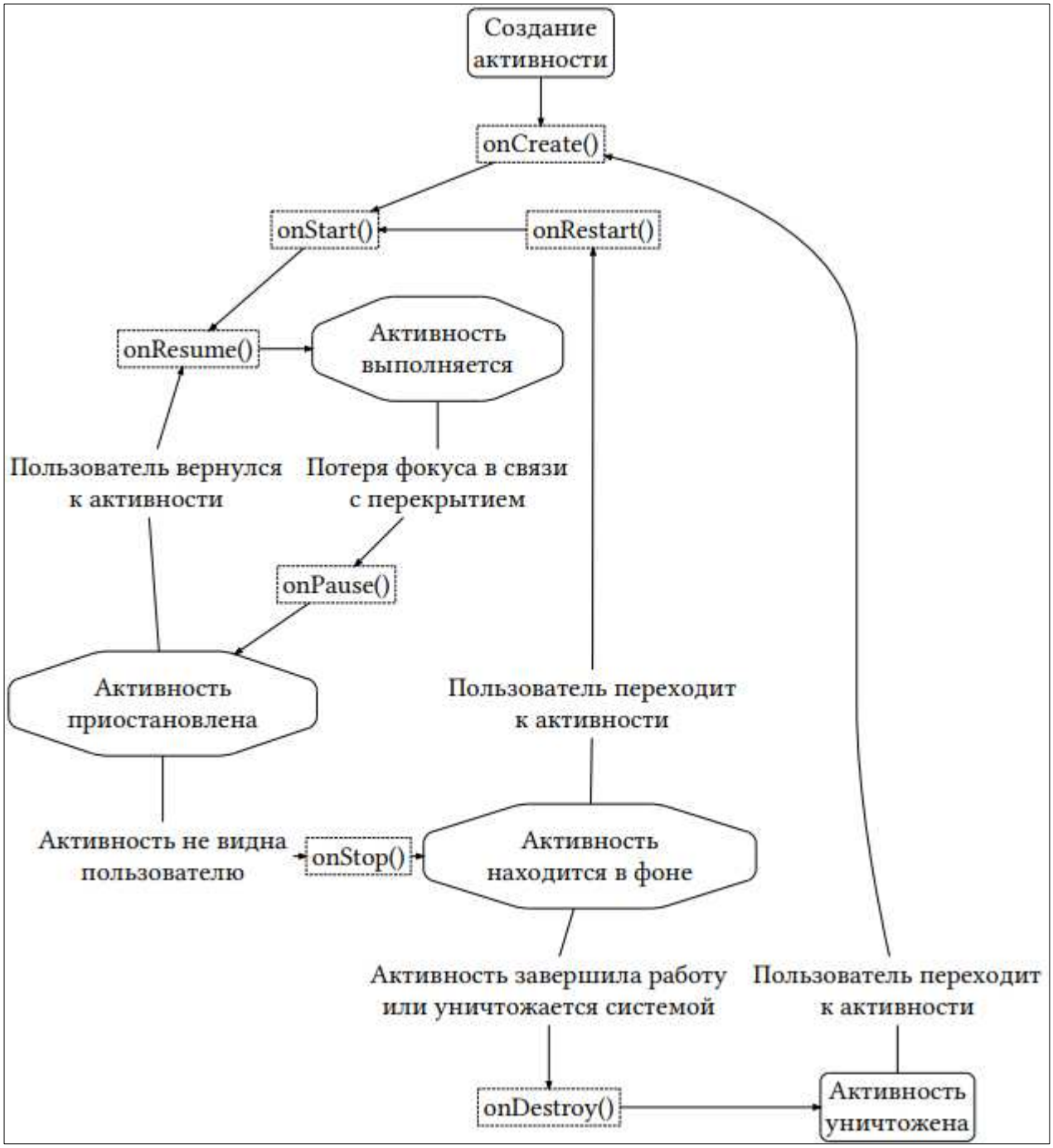

Рисунок 1 – Жизненный цикл активностей в Android

Метод onCreate() вызывается непосредственно после создания активности. Здесь осуществляется инициализация интерфейса пользователя, привязка данных к элементам интерфейса, создание потоков и т.д.

Парным к данному методу является метод onDestroy(), в котором необходимо освободить ресурсы, полученные при вызове onCreate().

Метод onPause() вызывается, когда активность теряет фокус в связи с перекрытием её экрана всплывающим окном, диалогом или другой

активностью. После возврата из данного метода активность перейдёт в состояние paused.

Активность, которая обладает данным состоянием, может быть уничтожена операционной системой когда угодно в том случае, если системе будет недостаточно оперативной памяти и имеются другие более высокоприоритетные приложения. Исходя из этого, метод onPause() должен предусматривать сохранение состояния активности для того, чтобы быть в состоянии восстановить его при перезапуске. Восстановление состояния осуществляется в callback-методе onResume().

В случае, если разработчик изменит системную конфигурацию приложения, которая содержит в себе смену локализации или поворот экрана, это приведет к уничтожению активности и в дальнейшем к её повторному созданию. При этом выполняются все callback-методы, включая onDestroy() для уничтожаемой и onStart() для вновь создаваемой активности. Для того чтобы отличить рассматриваемую ситуацию от ситуации, когда приложение закрыто пользователем, а затем вновь открыто, предусмотрена пара методов onSaveInstanceState() и onRestoreInstanceState(). Реализация этих методов в классе Activity автоматически сохраняет и восстанавливает состояние всех элементов пользовательского интерфейса. Однако в некоторых сложных случаях может потребоваться переопределение и этих методов для обеспечения корректного функционирования приложения при изменении конфигурации.

1.1.7 Задачи и стек активностей

Под задачей (task) в Android подразумевается набор активностей, вызываемых друг из друга и направленных на удовлетворение одной потребности пользователя. Список всех выполняемых на устройстве задач отображается, когда пользователь нажимает и удерживает кнопку «Home».

Когда пользователь запускает приложение, создаётся новая задача и первая открывшаяся активность запущенного приложения помещается в стек

активностей этой задачи. Относительно задачи эта активность называется корневой. Задача существует до тех пор, пока корневая активность не завершится.

Корневая активность может вызвать вторую активность, которая будет помещена в стек текущей задачи поверх корневой. В свою очередь, вторая активность может вызывать третью и т. д. Все эти активности помещаются в стек. При закрытии активности, находящейся на вершине стека (по нажатию кнопки «Back» на Android-устройстве или программно с помощью вызова соответствующего метода), она удаляется из стека, а управление передаётся той активности, которая находилась в стеке непосредственно под удалённой.

При нажатии кнопки «Home» текущая активность переходит в фоновый режим, однако весь стек активностей соответствующей задачи сохраняется. Если теперь нажать и удерживать кнопку «Home», то пользователь увидит список активных задач. При выборе любой из них активность, находящаяся на вершине стека, получит фокус, а все остальные активности будут сохраняться в стеке, пока находящиеся выше их активности не будут закрыты.

Следует отметить, что активности, входящие в стек некоторой задачи, могут входить в состав различных приложений. В этом выражается одна из важных концепций Android, декларирующая кооперацию нескольких различных приложений для удовлетворения потребностей пользователя и предоставляющая механизм интентов для осуществления этой кооперации.

1.1.8 Архитектура «модель-вид-контроллер»

Приложения для Android, как правило, разрабатываются в соответствии с архитектурным шаблоном «модель-вид-контроллер» (model-view-controller, MVC). Этот шаблон позволяет разделять отдельные слои ответственности приложения таким образом, чтобы упростить поддержку кода и облегчить внесение изменений.

Моделью в терминологии MVC является уровень бизнес-логики, ответственный за хранение и обработку информации. Данный слой не должен делать никаких предположений относительно представления информации или взаимодействия пользователя с приложением.

Потенциально это делает классы модели пригодными для повторного применения, в том числе на платформах, отличных от той, на которой они были разработаны.

Вид – это уровень отображения. Он ответствен за отображение данных модели. Классы, принадлежащие данному уровню, как правило, являются виджетами платформы либо унаследованы от них. Вид может обращаться к модели для получения данных, однако модель не должна непосредственно зависеть от вида.

Контроллер является уровнем, ответственным за взаимодействие с пользователем. К данному уровню относятся обработчики событий пользователя и часто код, соединяющий все компоненты системы в единое целое (так называемый glue code). В зависимости от конкретной реализации MVC уровни вида и контроллера могут быть отдельными классами или совмещены. Контроллер, как правило, зависит от модели, однако, как и в случае вида, модель не должна хранить ссылок на конкретные классы контроллера.

1.1.9 Назначение класса View

Класс View является суперклассом всех классов-виджетов в Android, включая TextView, ImageView, Button и т.д. Каждый экземпляр класса View ответствен за отрисовку некоторой прямоугольной области на экране, а также за обработку событий, связанных с этой областью.

При разработке приложений под Android, как правило, используются готовые библиотечные субклассы класса View. Однако в некоторых случаях бывают необходимы компоненты, имеющие специфический внешний вид и

поведение. Подобные компоненты можно легко реализовать, унаследовав собственный класс от класса View и переопределив методы, ответственные за отрисовку и/или обработку событий.

Из всех событий, обрабатываемых классом View, наиболее важными представляются события, возникающие при взаимодействии пользователя с областью, за которую ответствен View, события передачи фокуса, нажатия на клавиши и отрисовки.

1.1.10 Правила обработки событий вдоль иерархии виджетов

Каждый виджет имеет возможность содержать другие виджеты в рамках области, за которую он ответственен. Иерархия включения может быть сформирована непосредственно в программном коде или путём вложения элементов в XML-файле, описывающем пользовательский интерфейс. Исходя из практики, второй способ используют чаще.

Иерархия виджетов имеет существенное значение для обработки событий. Действуют специальные правила, определяющие обработку событий вдоль иерархии:

– первоочередно, событие делегируется самому вложенному (листовому в дереве вложения) виджету, который ответственен за область, в рамках которой произошло событие;

– в случае, если виджет не смог определить собственного обработчика произошедшего события, событие делегируется для обработки родительскому в терминах иерархии включения виджету; в противном случае происходит вызов обработчика дочернего виджета;

– в случае, если обработчик события возвращает «false», то виджет обработал событие, но требуется также продолжить обработку этого события родительским виджетом;

– в случае, если обработчик события возвращает «true», то считается, что виджет обработал событие, и дальнейшая обработка не требуется.

Описанные выше правила – гибкие, что означает что их можно применять не только для переопределения варианта обработки событий во вложенных виджетах, но также и для связывания в последовательность обработчиков разных виджетов из иерархии.

1.1.11 Работа с ресурсами

Ресурсы в Android являются статическими данными. Как пример можно привести описание пользовательского интерфейса, изображения, текст. Ресурсы располагаются программистом в проекте в виде файлов и транслируются в apk-пакет приложения автоматически во время сборки.

Использование ресурсов преследует две основные цели.

1. Реализация вариативности ресурсов, которые используются, исходя из конфигурации. Ресурсы позволяют реализовать поддержку различных типов ориентации экрана и различных языков интерфейса пользователя без дополнительных затрат со стороны программиста, так как ресурсы, подходящие для текущей конфигурации системы, загружаются автоматически.

2. Отделение данных от кода. При использовании ресурсов данные хранятся отдельно от кода в декларативном виде, поэтому их легко изменять, причём изменения могут вносить не программисты, а, например, дизайнеры пользовательского интерфейса или переводчики. Изменение ресурсов не требует перекомпиляции приложения – необходима лишь его пересборка.

В Android выделяются следующие основные типы ресурсов:

– Drawable – файлы изображений, используемых приложением. Сюда также входят графические файлы, используемые в пользовательском интерфейсе;

– Raw – произвольные данные, как правило бинарные;

– Layout – файлы в формате XML, описывающие расположение элементов интерфейса пользователя;

– Values – данные приложения, представленные в текстовом формате XML. В первую очередь в состав ресурсов данного типа входят все текстовые строки приложения, традиционно размещаемые в файле strings.xml;

– Menu – файлы в формате XML, описывающие компоновку элементов меню или панели действий.

Ресурсы каждого типа размещаются внутри каталога «res» проекта в подкаталогах, названия которых совпадают с типами ресурсов, приведёнными выше, но написаны строчными буквами.

Для использования ресурсов из приложения в ходе сборки проекта (а при использовании инструментальных сред также при любом изменении ресурсов приложения) создаётся специальный файл R.java, содержащий идентификаторы всех ресурсов проекта. Все они определены как статические константы внутри подклассов класса R, каждый из которых соответствует своему типу ресурсов:

– R.drawable — ресурсы-изображения из каталога res/drawables;

– R.raw — прочие ресурсы из каталога res/raw;

– R.id – компоненты пользовательского интерфейса, описанные в файлах ресурсов (файлы каталога res/layout); элементы идентифицируются по идентификаторам, задаваемым с помощью атрибута android:id;

– R.layout – ресурсы из каталога res/layout, описывающие компоновку пользовательского интерфейса;

– R.menu – ресурсы из каталога res/menu, описывающие состав меню или панели действий;

– R.string, R.integer, R.boolean, R.color, R.array и т. д. — ресурсы из файлов данных (файлы каталога res/values), сгруппированные по типам этих данных;

1.1.12 Хранение данных

Многим приложениям требуется сохранять данные в постоянной памяти и восстанавливать их при последующих запусках.

Платформа Android предоставляет три возможности решения данной задачи: настройки (prefereneces), файловая система и базы данных. В текущем приложении будет использована база данных.

1.2 Модель данных

Согласно классической теории БД, модель данных это теория визуализации и расчета информации в СУБД, содержащая следующие факторы:

1. Фактор структуры, определяющий способы определения типов и логических схем информации в БД;
2. Фактор манипуляции, определяющий способы манипулирования информацией;
3. Фактор целостности, определяющий способы определения и поддержки целостности БД.

Фактор структуры реализует собой логическое представление базы данных, фактор манипуляции реализует варианты транслирования между статусами БД и варианты получения информации из БД, фактор целостности реализует инструменты определений правильных статусов БД.

Модель данных представляет собой логическое, самодостаточное, абстрактное понятие операторов, объектов и прочих элементов, вместе представляющих механизм доступа к информации. Такие объекты предоставляют возможность проектирования структуры информации.

Базы данных и системы управления базами данных основываются на определенной явной или неявной структуре информации. Системы управления базами данных, реализованные по одинаковой структуре информации, называют однотипными.

Среди специализированных материалов часто можно увидеть понятие «модель данных», которое используется эквивалентно понятию «схема БД». Многие специалисты указывают на то, что это равенство проводить некорректно. Эквивалентность между данными терминами можно сравнить с

эквивалентностью программного языка и программой, реализованной при помощи этого языка.

1.3 Системы управления базами данных

Oracle

Oracle Database или Oracle DBMS является объектно-реляционной системой управления базами данных (СУБД). Oracle может хранить и выполнять хранимые процедуры и функции внутри себя. PL / SQL (собственные процедурные расширения корпорации Oracle на SQL), или объектно-ориентированного языка Java может ссылаться на такие объекты, коды и предоставлять возможность программирования структур.

Microsoft SQL Server

Microsoft SQL Server является системой управления реляционными базами данных (СУБД), разработанной корпорацией Microsoft. Используется для от небольших и средних по размеру баз данных до крупных баз данных масштаба предприятия, конкурирует с другими СУБД в этом сегменте рынка.

MySQL

MySQL является свободной системой управления базами данных (СУБД). Является решением для малых и средних приложений. Обычно используется в качестве сервера, к которому обращаются локальные или удалённые клиенты, однако в дистрибутив входит библиотека внутреннего сервера, позволяющая включать MySQL в автономные программы.

PostgreSQL

PostgreSQL (произносится «Пост-Грес-Кью-Эл» или просто «постгрес») является свободной объектно-реляционной системой управления базами данных (СУБД). Является свободной альтернативой коммерческим СУБД (таким как Oracle Database, Microsoft SQL Server, Informix и СУБД производства Sybase) вместе с другими свободными СУБД (такими как MySQL и Firebird).

Sybase ASA

Сервер управления базами данных "Sybase SQL Anywhere Studio" (сокращенно ASA) сейчас является одним из наиболее стремительно развивающихся продуктов компании iAnywhere Solution, дочерней компании холдинга Sybase.

FireBird

Firebird (FirebirdSQL) является компактной, кроссплатформенной, свободной системой управления базами данных (СУБД), работающей на GNU/Linux, Microsoft Windows и разнообразных Unix платформах. Это коммерчески независимый проект C и C++ программистов, технических советников и разработчиков мультиплатформенных систем управления базами данных, основанный на исходном коде, выпущенном корпорацией Borland 25 июля 2000 года в виде свободной версии Interbase 6.0.

InterBase

InterBase является системой управления базами данных, разработанной компанией Borland. Основными достоинствами последней версии InterBase являются низкие требования к системе, с одновременной масштабируемостью на несколько процессоров, плюс развитая система мониторинга, временные таблицы, встраиваемая аутентификация пользователей, журналирование.

1.4 Техническое задание на проектирование

1.4.1 Общие сведения

Полное наименование программного продукта: Мобильное приложение онлайн-сервиса для музыкантов;

Условное обозначение: Мобильное приложение для музыкантов;

Заказчик: ООО «Онлайн-Мьюзик»;

Исполнитель: Королёв М.А.;

Плановая дата начала выполнения работ: 16.02.2018;

Плановая дата окончания выполнения работ: 07.05.2018.

1.4.2 Назначение и цели создания программного продукта

Назначение программного продукта:

«Мобильное приложение онлайн-сервиса для музыкантов» реализовано для повышения скорости и эффективности взаимодействия пользователей онлайн-сервиса с данным сервисом. Основными пользователями онлайн-сервиса являются музыканты.

Целью данного мобильного приложения является:

1) обеспечение сбора и хранения данных об авторах;
2) увеличение скорости и производительности взаимодействия пользователей с сервисом;
3) возможность подбора нот музыкального произведения;
4) обеспечение возможности оценивания музыкальных произведений;
5) реализация возможности отправки музыкальных произведений.

1.4.3 Функциональные требования

Необходимо, чтобы разработанное мобильное приложение полностью отвечало заявленным пользовательским и функциональным требованиям.

Бизнес-требования:

Реализованное мобильное приложение должно предоставить повышение производительности и мобильности взаимодействия пользователей с онлайн-сервисов для музыкантов.

Пользовательские требования:

Составляющие мобильного приложения должны быть понятны для пользователей.

Для роли музыканта необходимо реализовать и сделать доступным модуль ведения базы авторов, модуль ведения базы музыкальных произведений (создание, загрузка, редактирование, удаление музыкальных произведений), подбор нот музыкального произведения, отправка музыкальных произведений, оценивание музыкальных произведений.

На рисунке 2 приведена реализованная диаграмма вариантов использования.

С помощью диаграммы вариантов использования отражено, какие действия мобильное приложение будет в состоянии произвести, т.е. функциональное назначение данного программного продукта.

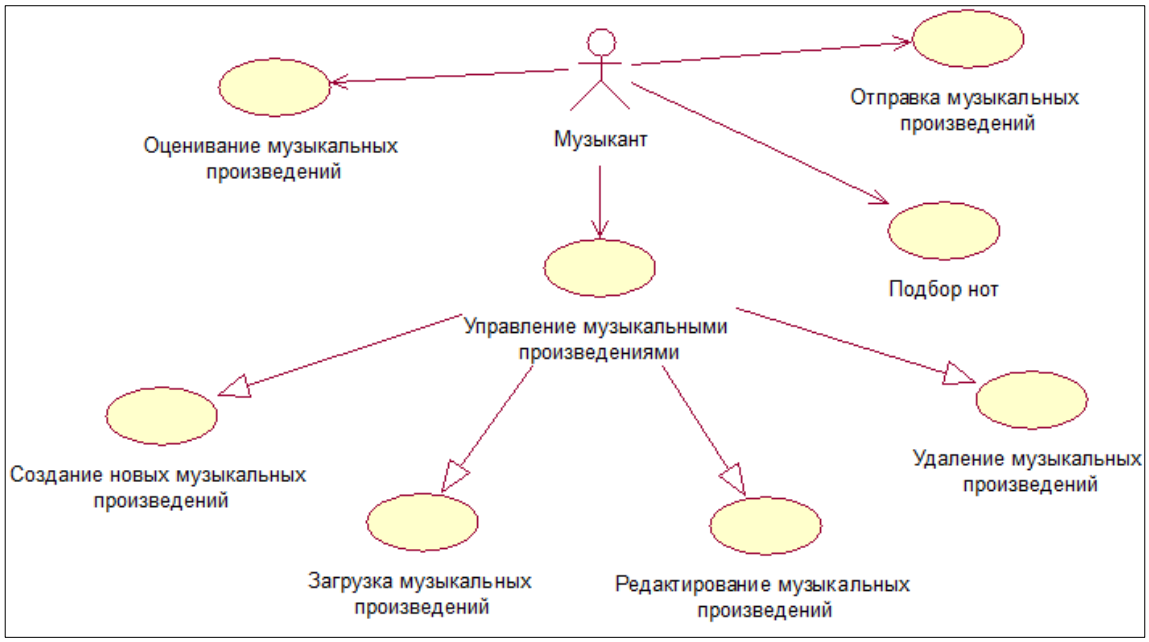

Рисунок 2 – Диаграмма вариантов использования

1.4.4 Нефункциональные требования

Интерфейс разработанного мобильного приложения, отвечающий за взаимодействие с пользователем должен быть интуитивно понятным и быстрым для обучения.

В реализованном мобильном приложении должны быть следующие основные модули: модуль ведения базы авторов, модуль работы с музыкальными произведениями.

Для работы в приложении предполагается организовать одну роль пользователя: музыкант. Данная роль будет взаимодействовать с музыкальными произведениями, и с самим онлайн-сервисом.

На рисунке 3 приведена разработанная диаграмма классов.

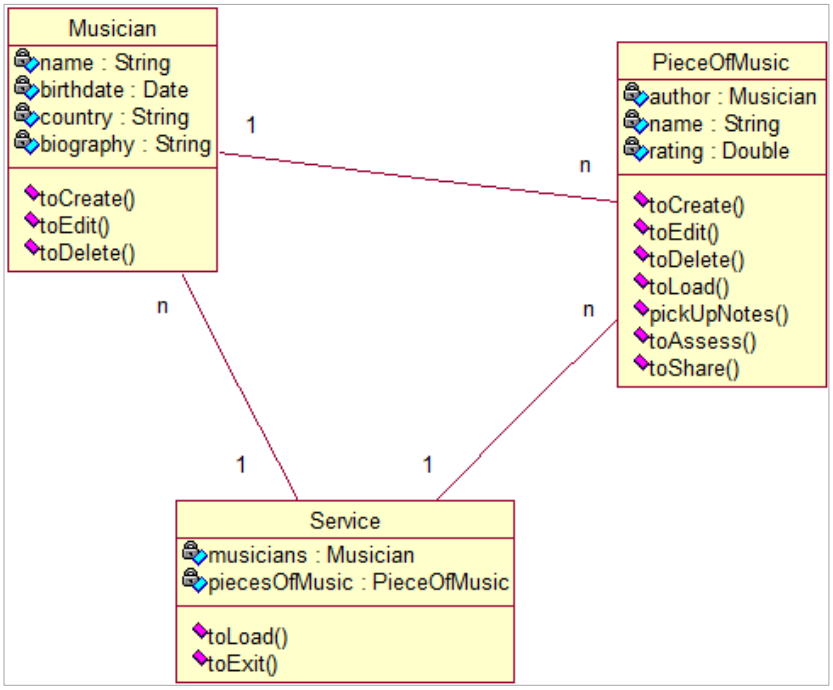

Рисунок 3 – Диаграмма классов

1.4.5 Системные ограничения

Необходимо, чтобы разработанное мобильное приложение могло работать на аппаратном обеспечении большинства пользователей, и для работы не нуждалась в высокой вычислительной мощности.

Системные требования:

– Операционная система Android 4.0

– Процессор Qualcomm Snapdragon 625 1 ГГц и выше;

– Оперативная память 512 Мбайт и выше (рекомендуется 768 Мбайт);

– Жесткий диск (при использовании необходимо около 50 Мбайт);

– TFT-дисплей (рекомендуется 4 дюймовый дисплей с разрешением 1024*768).

Требуется, чтобы в реализованном мобильном приложении не существовало несоответствий с другими приложениями, установленными на мобильном устройстве.

1.4.6 Атрибуты качества

В разработанном мобильном приложении должны присутствовать указанные атрибуты качества:

1. Отказоустойчивость, что означает способность приложения давать возможность дальнейшей работы с ним в случае возникновения нештатных ситуаций;

2. Масштабируемость, что означает способность приложения давать возможность увеличения эффективности и количества процессоров, объемов внешней и оперативной памяти, а также других параметров устройства;

3. Совместимость, что означает способность приложения не быть требовательным к варианту организации приложения и его архитектуре;

4. Производительность, что означает способность приложения исполнять поставленные задачи эффективно и без использования большого количества вычислительных ресурсов;

5. Доступность, что означает способность приложения демонстрировать длительное время непрерывной работы;

6. Надежность, что означает способность приложения быть устойчивым к появлению нештатных ситуаций при помощи уменьшения количества сбоев и отказов.

2 Описание проектных решений и реализация мобильного приложения

2.1 Обоснование выбора среды разработки

Стремительное появление инновационных технологий, связанных с обработкой информации, а также увеличение области применения таких технологий привели к бурному росту количества программных продуктов. Темпы роста качества и количества программных продуктов демонстрируют, что изменение трат на создание таких продуктов, или их покупку постепенно растет на сумму около 20% в год.

Термин «программное обеспечение программного продукта» означает комплекс документальных и программных инструментов для реализации и использования систем при помощи мобильных устройств.

В соответствии с функциями, которые реализует программное обеспечение, его подразделяют на 2 группы:

1) прикладное программное обеспечение (реализует решение определенных задач и целиком структурирует вычислительный процесс информационной системы.

2) базовое программное обеспечение (структурирует процесс обработки данных в вычислительной машине и реализует корректную среду для исполнения программ).

Базовое программное обеспечение содержит:

- операционные системы;
- сервисные программы;
- трансляторы языков программирования;
- программы технического обслуживания.

Взаимодействие между аппаратной частью мобильного устройства и пользователей реализуют операционные системы, которые управляют процессом взаимодействия с информацией. Автоматизация процессов ввода и

вывода данных является важной функций операционной системы, а также менеджмент исполнения внутренних задач. Операционная система помещает программный продукт в память мобильного устройства, после чего производит наблюдение за процессом исполнения данной программы. Она также производит анализ ситуаций, которые мешают корректным вычислениям, после чего решает, какие действия необходимо предпринять.

В связи с этим, операционные системы подразделяют на:

1) сетевые;
2) однозадачные (для одного пользователя);
3) многозадачные (для многих пользователей).

В качестве основной операционной системы определена операционная система Android, разработанная компанией Google. Данная операционная система позволит реализовать корректное исполнение мобильного приложения онлайн-сервиса для музыкантов.

2.2 Среда программирования 1С

Для реализации программного продукта выбрана мобильная версия платформы «1С:Предприятие 8». Выбранная платформа «1С:Предприятие 8» - это многозадачная система, предназначенная для накопления и анализа информации. В связи с тем, что процессы сервиса могут динамически изменяться, платформа «1С:Предприятие 8» может адаптироваться с учетом индивидуальных требований онлайн-сервиса. Эту способность также называют «конфигурируемость», что означает способность доработки программного продукта под изменившиеся процессы сервиса.

Такая возможность реализуется благодаря тому, что мобильная версия платформы «1С:Предприятие 8» является не только программным продуктом, но также комплексом разных компонентов, которые можно использовать для индивидуальной настройки.

На текущий момент более миллиона компаний используют для работы компоненты платформы «1С:Предприятие 8».

Внутри мобильной версии платформы «1С:Предприятие 8» существует удобный и эффективный интерфейс, позволяющий пользователям комфортно и длительно выполнять работу.

Мобильная версия системы «1С:Предприятие 8» реализует множество способов исполнения программного решения. Среди таких способов как использования в режиме приложения на мобильном устройстве, так и персональное использование в качестве информационной системы на персональном компьютере.

Платформа «1С:Предприятие 8» определяется как открытая. Это означает, что система способна интегрироваться с любым программным и аппаратным обеспечением.

Модуль прав доступа реализует разрешение или запрещение работы пользователей с определенным блоком или модулем программного продукта.

Мобильная платформа «1С:Предприятие 8» обладает следующими преимуществами:

1. Предусмотрено использование различных систем хранения данных;
2. Многоплатформенность: приложение может исполняться под любыми операционными системами, для которых разработана платформа, такими как Android или iOS;
3. Интеграция с любыми программами и оборудованиями, которые поддерживают общепризнанные протоколы передачи данных и стандарты;
4. Удобное администрирование;
5. Отказоустойчивость, обеспечивается бесперебойность работы пользователей во время возникновения сбоев аппаратного или программного характера за счёт резервирования кластера, резервирования рабочих процессов и устойчивости к разрыву связи.

2.3 Описание реализованного мобильного приложения

При запуске мобильного приложения пользователь видит окно следующего вида (рисунок 4):

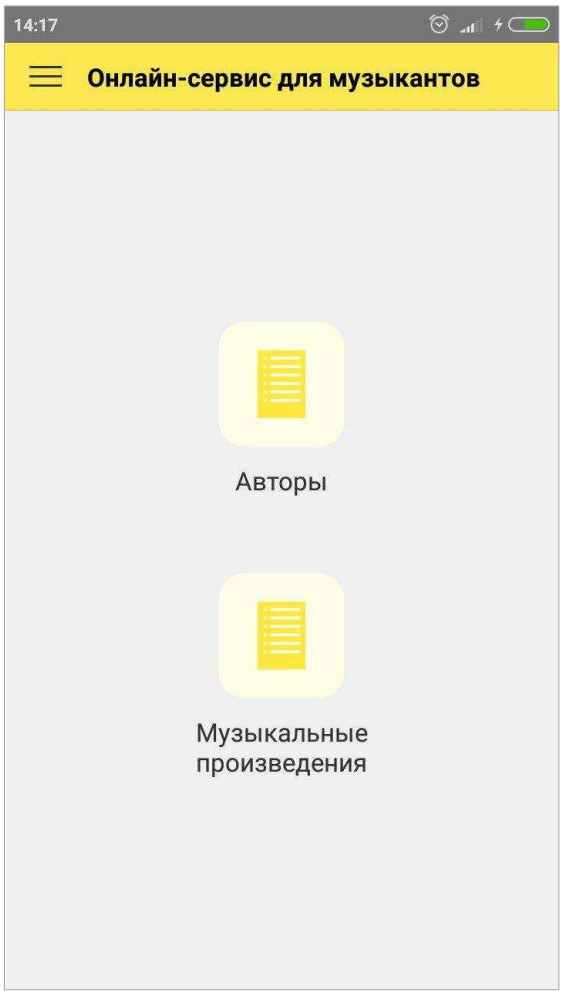

Рисунок 4 – Главное окно приложения

При работе с приложением, пользователь может перейти в режим работы с авторами, либо в режим работы с музыкальными произведениями. В данном окне пользователю предлагается сделать соответствующий выбор.

При нажатии на кнопку «Авторы» пользователь переходит в режим работы с авторами. В данном режиме пользователю открывается окно с базой авторов, из которых он может выбрать уже существующего автора или создать нового (рисунок 5).

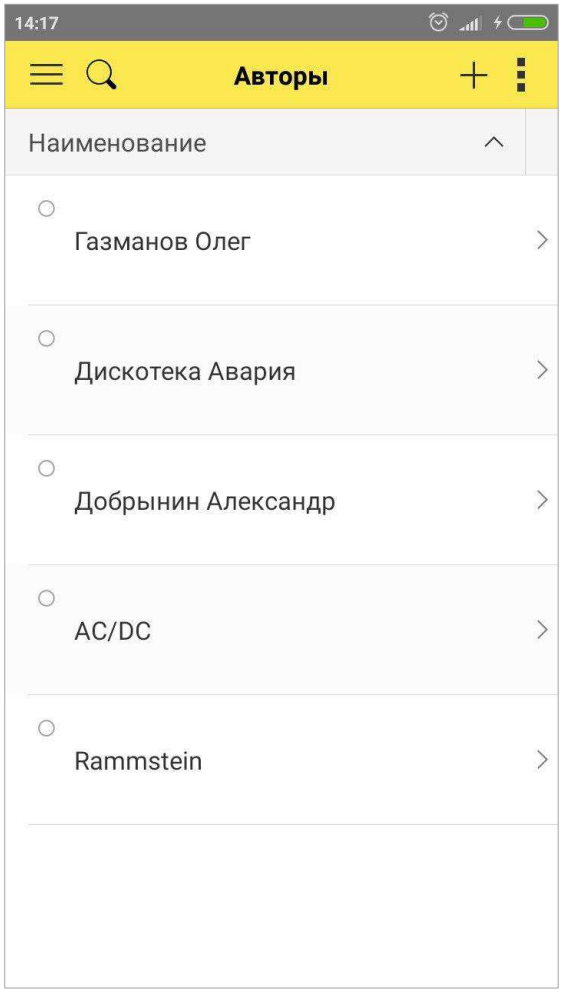

Рисунок 5 – Режим работы с авторами

При нажатии на существующего автора пользователь перейдет в окно карточки автора, в котором он увидит все поля автора с заполненными данными в них (рисунок 6). Также при желании пользователь может осуществить поиск среди существующих авторов по одному из полей.

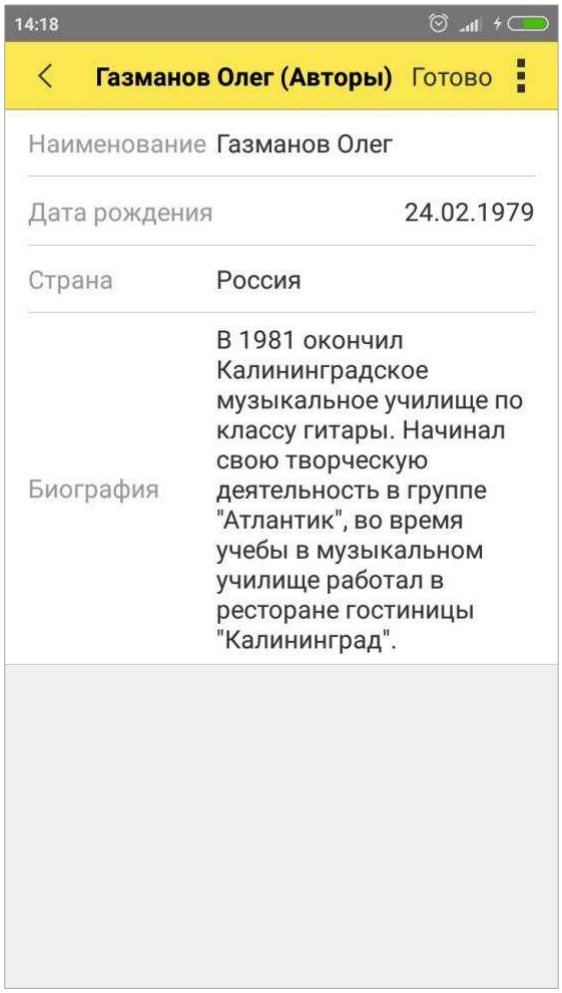

Рисунок 6 – Карточка существующего автора

При нажатии на кнопку «Музыкальные произведения» в главном окне приложения, пользователь перейдет в режим работы с музыкальными произведениями. В данном режиме пользователь увидит список музыкальных произведений в виде авторов и названий каждого из них (рисунок 7).

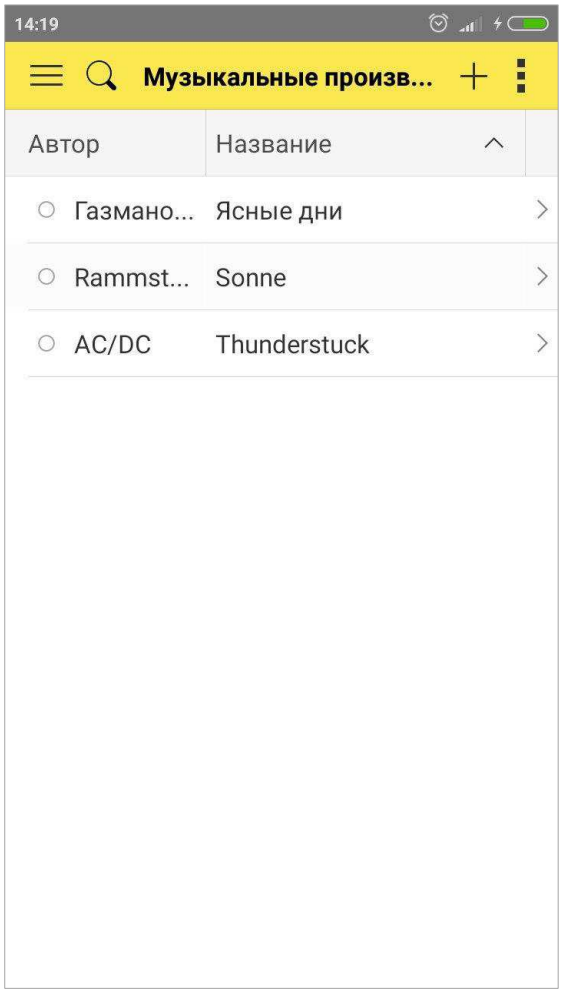

Рисунок 7 – Режим работы с музыкальными произведениями

При выборе какого-либо из музыкальных произведений, пользователю откроется карточка музыкального произведения. В данной карточке пользователь сможет увидеть автора, название и среднюю оценку по данному музыкальному произведению (рисунок 8).

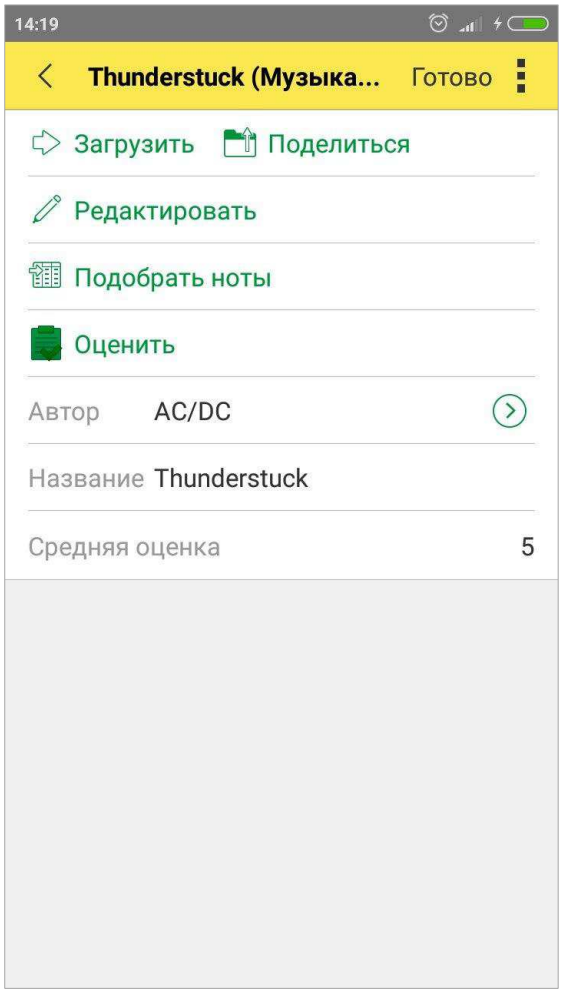

Рисунок 8 – Работа с определенным музыкальным произведением

Воспользовавшись данной формой, пользователь может произвести следующие операции с музыкальным произведением с помощью соответствующих кнопок:

– Загрузить музыкальное произведение в текущую карточку;

– Поделиться музыкальным произведением с другими пользователями;

– Отредактировать музыкальное произведение в текущей карточке;

– Подобрать ноты для музыкального произведения;

– Оценить музыкальное произведение.

При выполнении любой из этих операций, мобильное приложение будет обращаться к серверной части онлайн-сервиса. Таким образом, для исполнения

данных функций, приложению будет необходимо, чтобы мобильное устройство было подключено к Интернет.

ЗАКЛЮЧЕНИЕ

Целью данной выпускной квалификационной работы являлась разработка мобильного приложения для повышения скорости и эффективности взаимодействия пользователей с онлайн-сервисом для музыкантов. В процессе реализации выпускной квалификационной работы разработан программный продукт с использованием современного инструмента разработки мобильных приложений на базе платформы «1С: Предприятие 8».

В процессе разработки программного продукта проанализированы процессы взаимодействия пользователей с онлайн-сервисом, приобретен навык проектирования БД и работы с ними, изучен язык программирования для разработки мобильных приложений на базе платформы «1С:Предприятие 8».

С целью выполнения поставленной цели были решены следующие задачи:

– анализ предметной области;

– выявление процессов онлайн-сервиса, требующих реализации в мобильном приложении;

– формирование технического задания на разработку мобильного приложения;

– обеспечение ведения и поддержки базы данных;

– создание удобного интерфейса, интуитивно-понятных связей между диалогами ввода информации.

С помощью реализованного продукта пользователи онлайн-сервиса смогут использовать сервис на своих мобильных устройствах, что увеличит величину охвата аудитории потенциальных пользователей.

Это приведет к увеличению потока регистраций новых пользователей, усилению активности уже зарегистрированных пользователей, а также в целом к повышению лояльности пользователей к онлайн-сервису.

По сути, использование онлайн-сервиса теперь не ограничивается только полноценными компьютерами или ноутбуками, пользователи имеют возможность получать все услуги более мобильно и не менее полноценно, чем пользователи веб-версии на компьютере.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Тельнов, Ю.Ф. Информационные системы и технологии: Научное издание. / Ю.Ф. Тельнов. – Москва: ЮНИТИ, 2016. – 303 с.
2. Парамонов, И.В. Разработка мобильных приложений для платформы Android: учебное пособие / И.В. Парамонов; Ярославский государственный университет им. П.Г. Демидова. – Ярославль: ЯрГУ, 2013. – 88 с.
3. Федотова, Е.Л. Информационные технологии и системы: Учебное пособие / Е.Л. Федотова. – Москва: ИД ФОРУМ, НИЦ ИНФРА, 2013. – 352 с.
4. Ульман, Д. Основы реляционных баз данных / Д. Ульман, Д. Уид: Издательство Лори, 2006. – 384 с.
5. Рубичев, Н.А. Измерительные информационные системы / Н.А. Рубичев. – Москва: Дрофа, 2010. – 334 с.
6. Кашаев, С. Программирование в «1С:Предприятие 8.3». / С. Кашаев – Санкт-Петербург: Питер, 2014. – 451 с.
7. Бородакий, Ю.В. Информационные технологии. Методы, процессы, системы / Ю.В. Бородакий, Ю.Г. Лободинский. – Москва: ГЛТ, 2004. – 456 с.
8. Радченко, М. Архитектура и работа с данными «1С:Предприятия 8.2» / М. Радченко, Е. Хрусталева – Москва: 1С-Паблишинг, 2011. – 268 с.
9. Миков, А.И. Информационные процессы и нормативные системы в IT: Математические модели. Проблемы проектирования. Новые подходы / А.И. Миков. – Москва: КД Либроком, 2013. – 256 с.
10. Иванова, Г.С. Технология программирования: учеб. для вузов / Г.С. Иванова. - Москва: Издательство МГТУ им. Н. Э. Баумана, 2002. – 319 с.
11. Хрусталева, Е. Разработка сложных отчетов в «1С:Предприятии 8». Система компоновки данных / Е. Хрусталева. – Москва: 1С-Паблишинг, 2012. – 458 с.

12. Федорова, Г. Информационные системы. / Г. Федорова – Москва: Academia, 2013. – 208 с.

13. Норенков, И.П. Автоматизированные информационные системы: Учебное пособие / И.П. Норенков. - Москва: МГТУ им.Баумана, 2011. - 342 с.

14. Пирогов, В. Информационные системы и базы данных: организация и проектирование. / В. Пирогов. – Санкт-Петербург: БХВ-Петербург, 2005. – 492 с.

15. Голицина, О. Информационные системы и технологии. / О. Голицина, И. Попов, Н. Максимов – Москва: Инфра-М, 2014. – 400 с.

16. Чаадаев, В.К. Информационные системы компаний связи. Создание и внедрение / В.К. Чаадаев. – Москва: Эко-Трендз, 2004. - 256 с.

17. СТО 4.2–07–2014 Система менеджмента качества. Общие требования к построению, изложению и оформлению документов учебной деятельности. – Введ. 30.12.2013. – Красноярск: ИПК СФУ, 2014. – 60 с.