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

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

Кафедра «Прикладная математика и информатика»
(наименование)

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

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

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему Разработка мобильного приложения абонента Интернет-провайдера

Обучающийся Н.Г. Чистяков _____
(Инициалы Фамилия) (личная подпись)

Руководитель канд.пед.наук, доцент, Т.А. Агошкова
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Консультант И.Н. Усатова
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Тольятти 2022

Аннотация

Бакалаврская работа выполнена на тему «Разработка мобильного приложения абонента Интернет-провайдера».

Актуальность бакалаврской работы заключается в предоставлении Интернет-провайдеру инструментов оперативного взаимодействия с абонентом, а абоненту – удобного способа управления тарифами и услугами.

Бакалаврская работа структурно состоит из введения, трёх глав, заключения, списка использованных источников и приложения.

Введение содержит подробное описание актуальности выбранной темы, новизну работы, объект и предмет исследования, цель работы и задачи, необходимые для достижения цели.

Первая глава бакалаврской работы посвящена исследованию предметной области, формированию требований к проектируемому приложению и классификации этих требований. Кроме этого, проводится обоснование выбора платформы реализации мобильного приложения.

Во второй главе работы проводится моделирование архитектуры приложения, включающее функциональную и логическую модели мобильного приложения, диаграммы вариантов использования и взаимодействия компонентов системы, а также логическая структура базы данных.

В третьей главе бакалаврской работы описывается процесс выбора платформы реализации приложения и практическая реализация функций мобильного приложения.

Бакалаврская работа представлена в объёме 52 страниц, содержит 3 таблицы, 21 рисунок, 5 листингов программного кода и 1 приложение. Список использованный литературы содержит 35 источников, в т.ч. 5 на иностранном языке.

Abstract

The title of the graduation work is «Development of a mobile application for an Internet provider client».

The relevance of the bachelor's work lies in providing the Internet provider with tools for operational interaction with the subscriber, and also a convenient way to manage communication services for the subscriber.

The bachelor's work contains an introduction, three chapters, a conclusion, a list of references and an appendix.

The introduction contains a detailed description of the relevance of the chosen topic, the novelty of the work, the object and subject of research, the purpose of the work and the tasks necessary to achieve the goal.

The first chapter of the bachelor's work is devoted to the study of the subject area, the formation of requirements for the designed application and the classification of these requirements. In addition, the rationale for choosing a platform for the implementation of a mobile application is carried out.

In the second chapter of the work, modeling of the application architecture is carried out, including the functional and logical models of the mobile application, diagrams of use cases and interaction of system components, as well as the logical structure of the database.

The third chapter of the bachelor's work describes the process of choosing an application implementation platform and the practical implementation of the functions of a mobile application.

Bachelor's work is presented in the amount of 52 pages, contains 3 tables, 21 figures, 5 code listings and 1 appendix. The list of used sources contains 35 sources, including 5 in a foreign language.

Оглавление

Введение

Отрасль телекоммуникаций в настоящее время является одной из ключевых составных частей мировой экономики. И значимость обмена информацией, безусловно, растёт по мере развития общества. Своевременности и актуальности информационных потоков уделяют особое внимание международные организации и правительства большинства стран, поскольку телекоммуникации позволяют согласовывать работу отдалённых элементов любой компании, государственной системы и других индустриальных формирований [2], [21].

Интернет-провайдеры, предоставляющие услуги телекоммуникаций пользователям заинтересованы в привлечении большего числа клиентов, что достигается в том числе удобством предоставления услуг и их управлением. Сейчас почти каждый человек пользуется смартфоном, поэтому реализация управления теми или иными услугами в форме мобильного приложения – очевидный шаг для практически любого бизнеса, в том числе и для Интернет-провайдеров [14], [16].

То есть актуальность работы заключается в упрощении пользователям взаимодействия с Интернет-провайдером, потенциальном увеличении числа абонентов, получении альтернативного канала рекламы и информирования абонентов, а также снижении затрат на внутренние структуры поддержки пользователей.

Новизна работы заключается в том, что в приложении используются функции альтернативного отображения графического интерфейса посредством кэша для частичного использования приложения в отсутствие Интернет-соединения. Такая функция не реализована на данный момент ни в одном из приложений Интернет-провайдеров.

Объектом работы является деятельность Интернет-провайдера.

Предмет работы – технологии проектирования и разработки мобильных приложений.

Целью работы является разработка функционального мобильного приложения Интернет-провайдера.

Для достижения обозначенной цели необходимо выполнить следующие задачи:

– проанализировать деятельность Интернет-провайдера;

– определить на основе анализа требования к разрабатываемому мобильному приложению;

– разработать функциональную модель приложения на базе требований;

– спроектировать логическую структуру работы мобильного приложения;

– проанализировать современные инструменты разработки мобильных приложений и выбрать наиболее подходящий;

– осуществить практическую реализацию функций мобильного приложения посредством выбранного инструментария.

Работа опирается на практические методы решения задач и несёт практическую ценность в форме реализации требуемых функций.

Структурно ВКР состоит из введения, трёх глав, заключения и списка использованной литературы. Первая глава работы посвящена анализу исследуемой области и подготовке требований к мобильному приложению на основании этого анализа. Во второй главе рассматриваются теоретические принципы разработки приложения, планируется его архитектура и логика работы. Третья глава ВКР описывает программную реализацию выбранных функций мобильного приложения.

Глава 1 Формирование требований к мобильному приложению абонента Интернет-провайдера

1.1 Анализ предметной области

В Российской Федерации телекоммуникационная отрасль хоть и находятся в достаточно зрелой стадии своего развития, тем не менее регулярно претерпевает различные изменения касательно протоколов обмена информацией, расширения зон покрытия связи, повышения качества и стабильности услуг, и в целом идёт в ногу со временем, развивая услуги по обмену информации в соответствии с общемировыми тенденциями.

Внедрение пятого поколения мобильной связи (5G), происходящее в данный момент, ещё больше повышает требования к отрасли и вместе с тем расширяет инфраструктуру цифровой экономики. Если ещё 50 лет назад телекоммуникации ограничивались в основном телефонией, телевидением и радиотехнологиями, то сегодня высокоскоростным подключением к Интернету обладают даже бытовые чайники, автомобильные шлагбаумы, счётчики воды и электроэнергии, а также многие другие устройства [13].

Пандемия COVID-19 также укрепила позиции рынка телекоммуникаций, поскольку и государственным органам, и бизнесу в связи с переходом на удалённый формат работы пришлось полагаться на операторов связи [22]. По данным Минцифры, с 2015 года доходы компаний в телекоммуникационной сфере выросли больше чем на 150 миллиардов рублей (рисунок 1).

Обеспечением стабильного и надёжного подключения к сети Интернет занимаются специальные организации: интернет-провайдеры. Кроме этого, в перечень услуг, предоставляемых провайдером, часто входят:

– стационарная телефонная связь;

– мобильная связь;

– спутниковая связь;

– телевидение (в том числе IP-TV);

– производство собственного оборудования;

– предоставление услуг call-центров;

– предоставление и поддержка виртуальных корпоративных серверов.

Рисунок 1 – Динамика доходов российского телекоммуникационного рынка, млрд руб.

Интернет-провайдер может как иметь свой собственный магистральный канал связи, так и арендовать часть ресурсов у более крупного провайдера. Основными крупнейшими провайдерами в России на данный момент являются:

– ПАО Мобильные ТелеСистемы;

– ПАО Ростелеком;

– ПАО Мегафон;

– ПАО ВымпелКом (Билайн);

– АО «ТрансТелеКом».

Это так называемые «магистральные» провайдеры: компании, занимающиеся размещением собственных линий связи. Они предоставляют каналы передачи информации (оптоволоконные линии, вышки мобильной

связи, спутниковое и другое телекоммуникационное оборудование) в аренду менее крупным провайдерам, а те, в свою очередь, продают услуги связи конечному потребителю – физическим лицам и компаниям. Магистральные провайдеры вкладывают огромные ресурсы в создание сети широкополосных линий между городами и странами, обеспечивая связью крупные города и регионы. Масштабы линий магистрального провайдера на примере «Ростелеком» представлены на рисунке 2. Также существуют провайдеры, специализирующиеся на предоставлении вычислительных мощностей дата-центров и серверных кластеров [18].

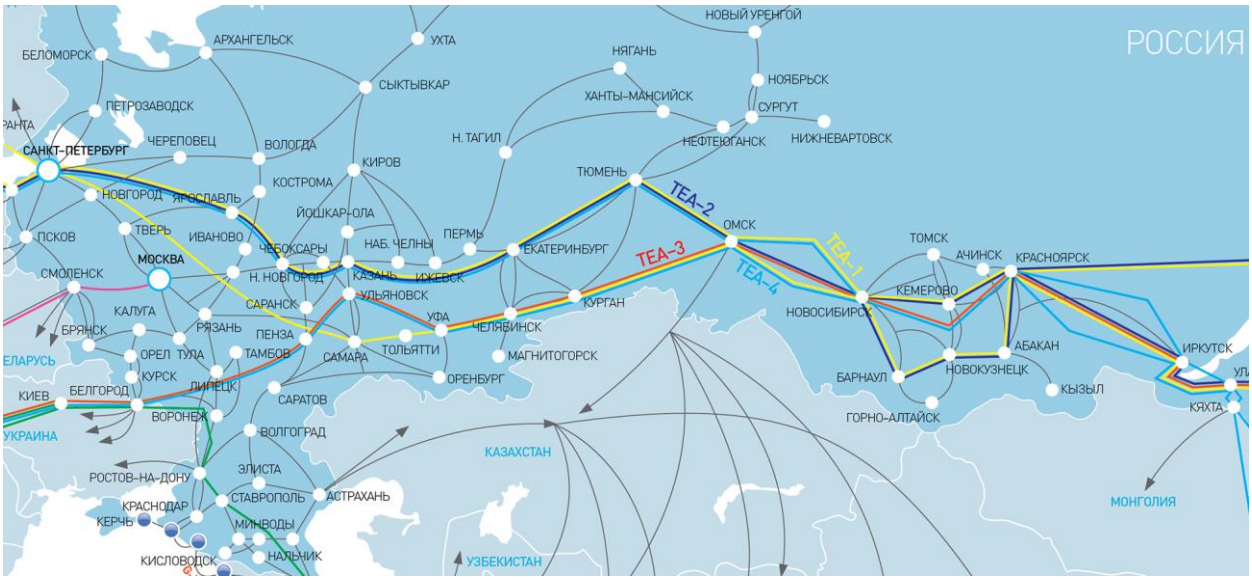

Рисунок 2 – Схема магистральных линий связи ПАО "Ростелеком" (фрагмент)

В задачи Интернет-провайдера, помимо предоставления доступа к глобальной сет Интернет также входит обеспечение безопасного и защищённого доступа к ресурсам. Это достигается за счёт использования специализированного программного обеспечения, DNS-серверов, магистральных маршрутизаторов и другого коммутационного оборудования [5]. Перед магистральными провайдерами также лежит ответственность за обеспечение блокировки нежелательных ресурсов, обеспечение анализа и

фильтрации трафика на территории страны, обеспечение функционирования и управления т.н. «Рунетом» (части Интернета, территориально находящейся в РФ и странах СНГ и представленной, по большей части, на русском языке), ограничение доступа к запрещённым законодательством ресурсам. В части выполнения этих требований магистральные провайдеры тесно сотрудничают с основным органом исполнительной власти по надзору в сфере связи, информационных технологий и СМИ – Роскомнадзором.

Таким образом, деятельность Интернет-провайдеров представлена очень широким спектром задач, но, как и любая другая коммерческая компания, прежде всего Интернет-провайдер ставит перед собой задачу получения прибыли. Самый надёжный и очевидный способ увеличения доли прибыли – привлечение новых клиентов и переманивание клиентов у конкурентов, для чего провайдерами регулярно разрабатываются акционные, скидочные и рекламные кампании. Каждый провайдер старается предложить пользователям разнообразные тарифные планы, чтобы увеличить долю присутствия на рынке. Стоимость услуг при этом не всегда напрямую зависит от скорости доступа и объёма трафика. Нередки ситуации, что в рекламных целях провайдер предлагает дешёвый пакет высокоскоростного трафика, а затем старается вернуть упущенную выгоду дополнительными услугами [27].

1.2 Анализ услуг, предоставляемых Интернет-провайдером

Первые попытки стандартизировать работу провайдеров были реализованы в 1988 году международным комитетом телефонии и телеграфии CCITT (Comité Consultatif International Téléphonique et Télégraphique) [24]. Комитет предложил стандартизированную модель управления телекоммуникациями, подкреплённую опытом ведущих компаний и едиными принципами стандартизации. Модель получила название TMN (Telecommunications Management Network). Представленная модель состоит из пяти логических уровней (рисунок 3).

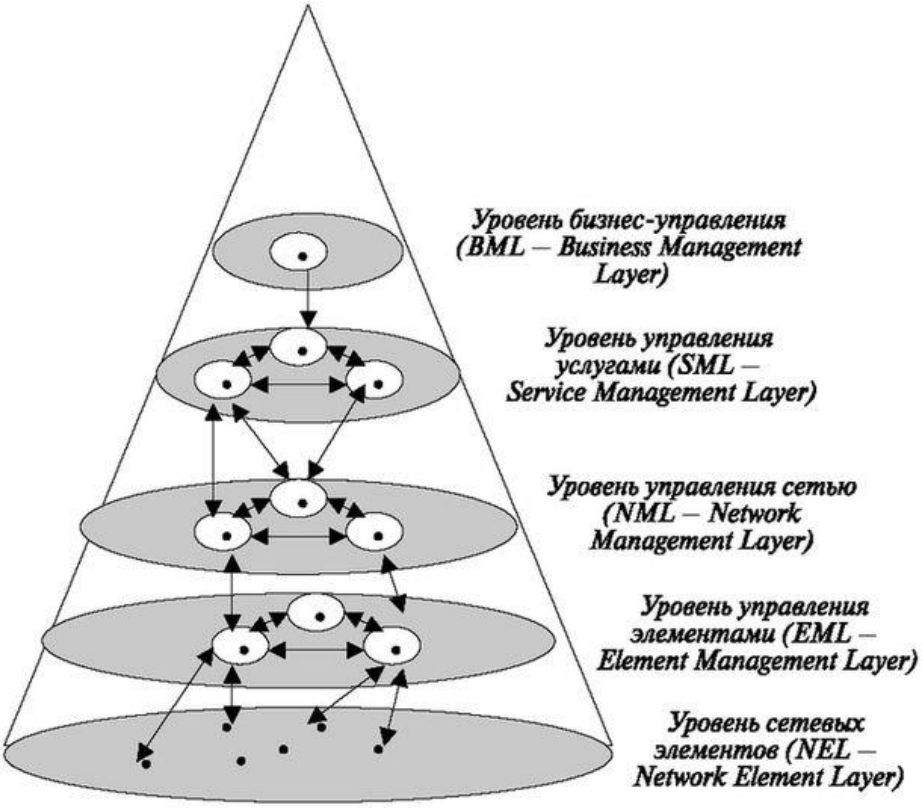

Рисунок 3 – Логическая схема модели TMN

Самый нижний уровень, уровень сетевых элементов, предполагает обмен информацией между инфраструктурой TMN и базами данных со служебной информацией. На этом уровне сетевые элементы согласуют работу сети связи и TMN [35]. Следующий уровень группирует отдельные сетевые элементы и обеспечивает управление этими группами. На этом уровне реализуется массовое управление оборудованием, например, фиксируются ошибки оборудования, реализуется управление пулами связи, проводится контроль рабочей температуры и аппаратной загрузки оборудования и т.д. Уровень управления сетью даёт представление обо всей сети в целом. На нём определяются таблицы маршрутизации, отслеживается общая производительность сети и её пропускная способность, контролируется объём трафика, проходящего через сеть. Вышерасположенный уровень управления услугами затрагивает конечного пользователя – именно на нём реализуется контроль качества предоставленных услуг, подключение новых абонентов,

фиксация адресов в сети, взаимодействие со сторонними провайдерами, биллинг и многое другое. Высший уровень предусматривает стратегическое и тактическое планирование бизнеса в целом, сеть на нём рассматривается не с технической точки зрения, а как инструмент реализации задач бизнеса. Таким образом, архитектура TMN задаёт иерархическую связь для процедур управления сети, но при этом не требует физического вмешательства в административное программное обеспечение для сегментации этих процедур [19].

На каждом уровне модели TMN присутствует 5 функциональных областей (рисунок 4).

Рисунок 4 – Функциональные области каждого уровня TMN

Они определяют общий набор протоколов и услуг, обеспечивающих обмен управляющей информацией при взаимодействии между управляющим процессом и управляемым ресурсом с использованием общего набора сообщений. Более детально с функциями каждого уровня можно ознакомиться в документах стандарта серии M.32xx.

Впоследствии, при эксплуатации модели были выявлены недостатки, которые привели к разработке более усовершенствованных и узконаправленных концепций. Наиболее существенными минусами TMN были трудности с расчётом эффективности внедрения системы сетевого управления, сложный для интерпретации язык описания модели, затруднения, связанные с переносом модели на имеющиеся протоколы (в том числе такой широко известный протокол как IP), а также общая техническая недоработанность концепции [28].

Следующим этапом унификации систем управления телекоммуникационными сетями стала модель TOM (Telecom Operations Map). Первая версия была представлена в 1995 г. некоммерческой ассоциацией TMF (рисунок 5).

Рисунок 5 – Модель TOM

Дальнейшие инициативы консорциума по исследованию бизнес-процессов в сфере телекоммуникаций объединились в проект New Generation Operation Systems and Software (NGOSS), в которую входит усовершенствованная методология бизнес-процессов телекоммуникационной отрасли eTOM (Enhanced Telecom Operations Map) [33].

Именно от этой концепции мы будем отталкиваться при разработке приложения интернет-провайдера, поскольку на данный момент eTOM является эталоном, на который опирается абсолютное большинство телекоммуникационных компаний. Кроме того, eTOM в большей степени регулирует взаимоотношения с клиентом, партнёрами и другими сторонними организациями. Операционные процессы eTOM представлены на рисунке 6.

Рисунок 6 – Операционные процессы в модели eTOM

Для разработки приложения наибольший интерес представляют процессы взаимодействия с клиентами и управления услугами. Управление ресурсами и связь с поставщиками и партнёрами реализуется, как правило, за

счёт специализированного ПО, которое к теме ВКР не относится. На рисунке 7 выделены процессы, которые возможно реализовать посредством мобильного приложения [30].

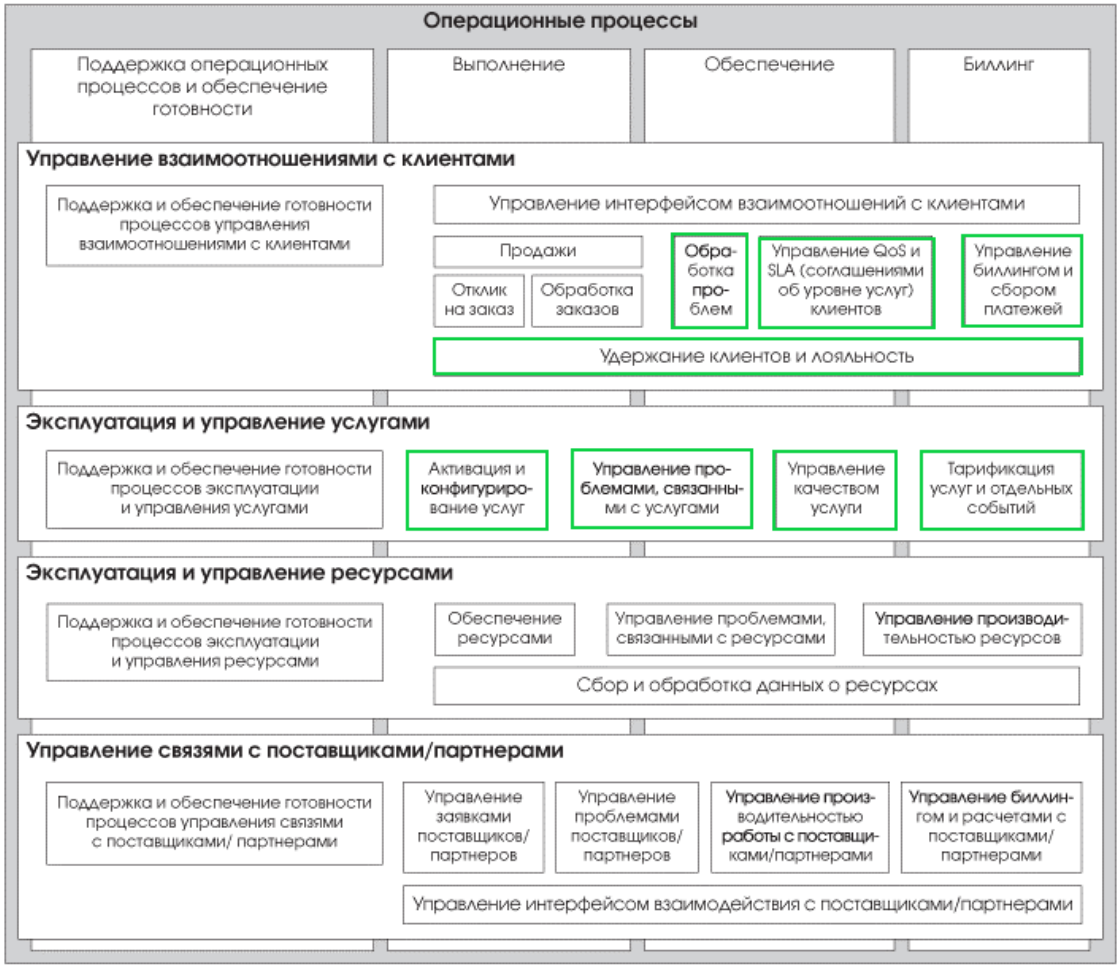

Рисунок 7 – Процессы концепции eTOM, реализуемые в приложении

В управлении взаимоотношений с клиентами основные услуги, реализуемые посредством приложения – биллинг и техническая поддержка клиента. Это основные возможности, от которых напрямую зависит качество предоставляемой услуги. Также необходимо обеспечить возможность активации и конфигурирования услуг (выбора и смены тарифа, дополнительных услуг, например, «родительский контроль» или «статический IP»), управления качеством услуг (в виде возможности оценки качества

предоставляемых услуг и сервисов), тарификации и просмотра истории платежей со стороны клиента.

1.3 Формирование требований к проектируемому мобильному приложению и постановка задачи

Исходя из услуг, предоставляемых Интернет-провайдером, а также внутренних процессов, сформируем требования к возможностям мобильного приложения. Формулирование требований будем проводить согласно классификации FURPS+. Сокращение FURPS расшифровывается следующим образом:

– Functionality, функциональность продукта;

– Usability, удобство использования;

– Reliability, надёжность;

– Performance, производительность;

– Supportability, поддерживаемость [1].

Символ «+» в настоящей классификации соответствует второстепенным атрибутам и различным ограничениям проектирования.

В блоке взаимоотношений с клиентами ключевыми процессами являются управление платежами, а также обработка проблем. Следовательно, в мобильном приложении необходимо реализовать возможность пополнения счёта абонента, историю платежей и зачислений, а также возможность связи абонента с техподдержкой посредством чата или звонков. При необходимости абонент должен иметь возможность заказать обратный звонок от техподдержки. Такой подход позволит более равномерно распределить нагрузку на операторов технической поддержки провайдера.

Из блока управления услугами мобильное приложение должно содержать возможности активации и конфигурирования услуг (тарифов), их тарификации и управления качеством услуг. Очевидно, что активация услуг и тарифов должна осуществляться в онлайн-режиме, тогда как просмотр

условий тарифов и отдельных услуг должен быть доступен абоненту без подключения к сети [17]. Процессы обработки проблем, связанных с услугами, перекликаются с обязанностями технической поддержки. Большинство Интернет-провайдеров имеют две линии поддержки: первая линия обрабатывает входящие заявки и консультирует абонентов по вопросам тарифов, услуг и их оплаты, а вторая линия решает технические проблемы с подключением и работой сети [26].

Кроме этого, мобильное приложение Интернет-провайдера должно содержать информацию рекламного характера для привлечения абонентов к дополнительным тратам. Это могут быть рекламные сообщения о новых тарифах, акциях, программах лояльности, предложениях партнёров и т.д. Итоговые требования к программному продукту представлены в Таблице А.1 (Приложение А). Приоритет реализации требований представлен значениями от 1 до 5, где 1 – наивысший приоритет, 5 – наименьший [31].

Таким образом, разрабатываемое приложение имеет 10 функциональных требований, которые определяют список основных возможностей приложения и приоритеты в их реализации.

1.4 Выбор платформы реализации мобильного приложения

Необходимо определить, для какой платформы будет разрабатываться мобильное приложение. Отталкиваясь от того, что целью разработки является упрощение процессов взаимодействия с провайдером как можно большего числа абонентов, опираться следует, в первую очередь, на наиболее массовую платформу. Такой подход позволит сразу охватить большую часть текущих абонентов Интернет-провайдера, и с большей вероятностью подойдёт потенциальным клиентам [21].

Согласно сведениям независимого инструмента по анализу веб-трафика statcounter.com, подавляющее большинство мобильных устройств в

Российской Федерации по состоянию на 2021 год работают под управлением операционной системы Android (рисунок 8).

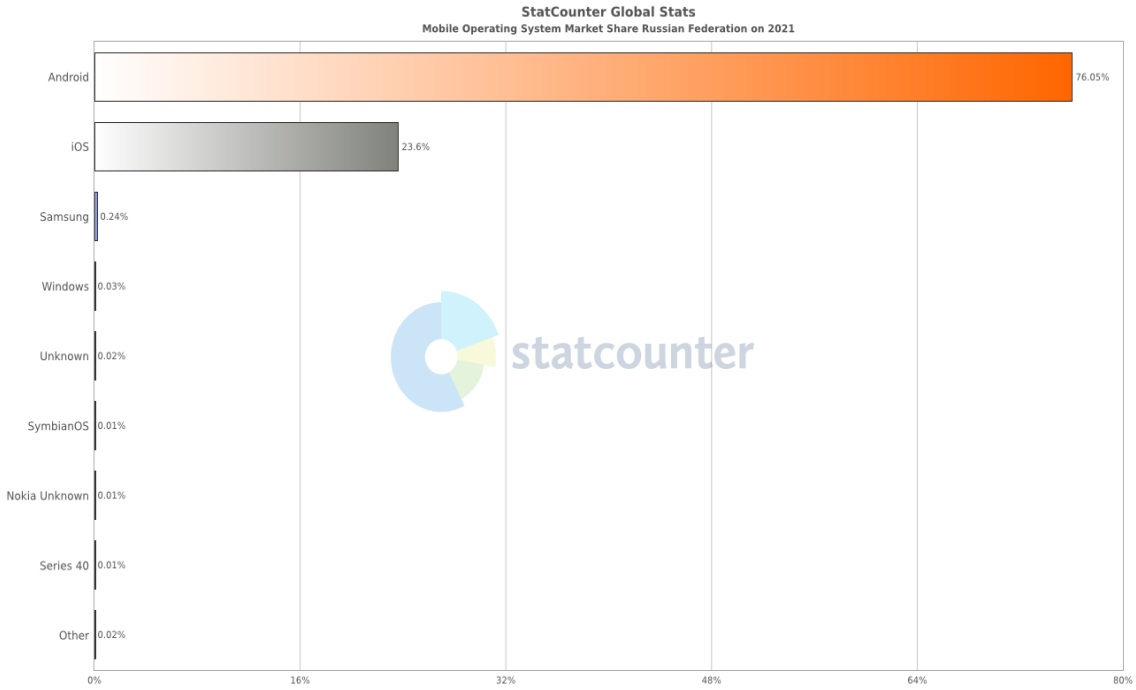

Рисунок 8 – Операционные системы мобильных устройств в России

Таким образом, разработка мобильного приложения будет осуществляться под операционную систему Android.

Выводы по Главе 1

В первой главе ВКР был проведён анализ предметной области, предметно обоснована актуальность выбранной темы, был проведён аналитический обзор услуг, предоставляемых Интернет-провайдером, сформированы функциональные требования к разрабатываемому мобильному приложению, было определено, для какой платформы будет разрабатываться мобильное приложение.

Глава 2 Проектирование мобильного приложения абонента Интернет-провайдера

2.1 Проектирование архитектуры мобильного приложения

Определив, какие задачи должно решать приложение, мы можем приступить к проектированию его архитектуры.

Поскольку разрабатываемое приложение не является сложным ни структурно, ни в реализации, но имеет жёсткие требования к производительности, лучшим вариантом будет использование так называемой «чистой архитектуры», предложенной популяризатором научного подхода к разработке приложений Fernando Cejas (рисунок 9) [4], [10].

Рисунок 9 – Структура «чистой архитектуры»

Суть «чистой архитектуры» мобильного приложения – независимость работы внутренних слоёв от внешних. То есть, сущность приложения не

меняется в зависимости от вариантов использования, сценарии остаются неизменными при смене контроллеров и шлюзов, а они, в свою очередь, не зависят от изменений в базы данных, интерфейс и другие внешние интерфейсы взаимодействия с пользователем [9]. На представленной схеме уже можно понять, как именно данные от пользователя попадают в базу данных и обратно, но, если расположить схему более линейно, потоки данных становятся очевидными (рисунок 10).

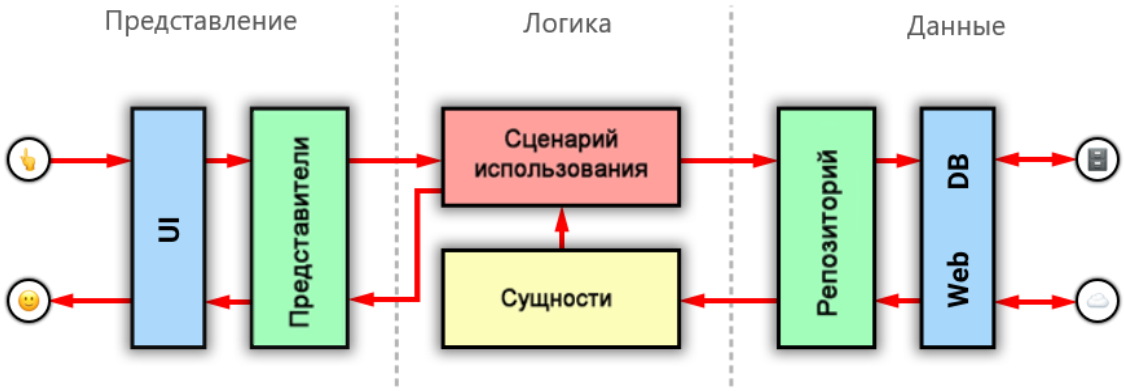

Рисунок 10 – Потоки данных в приложении с «чистой архитектурой»

Событие, инициализируемое пользователем, попадает в представление, затем переходит виртуальную границу к логике приложения, отрабатывается сценарий использования путём запроса в репозиторий, который получает необходимые данные из базы данных или веб-сервера, на базе этих данных создаётся сущность, которая проходит обратный путь через сценарий использования и отображается в интерфейсе пользователя [12].

Исходя из требований к приложению, оптимальным будет использование четырёхзвенной архитектуры сервиса. Первым звеном в ней является непосредственно клиентское мобильное приложение, которое отображает пользовательский интерфейс, и с которым напрямую взаимодействует клиент. Следующее звено в цепочке сервиса – сервер приложения, реализующий всю прикладную бизнес-логику работы

приложения [15]. Этот сервер является связующим между клиентским приложением и сервером базы данных, который хранит всю информацию, необходимую для работы приложения: учётные записи, информацию о лицевом счёте, информацию об услугах и т.д. Четвёртый элемент системы – сторонний сервис платежей [29]. Он обрабатывает платежи, поступающие от клиентов, а информация о транзакциях поступает на сервер базы данных. Схема взаимодействия компонентов приложения представлена на рисунке 11.

Рисунок 11 – Схема взаимодействия компонентов мобильного приложения

Данная архитектура позволит реализовать все представленные требования, и даже в случае временной неработоспособности одного из компонентов приложения, остальные продолжат работу в штатном режиме. Такая возможность реализуется посредством использования локального хранилища данных на устройстве абонента. При запуске приложения, в первую очередь, проверяется доступность сервера приложения и сервера базы данных, затем информация в локальном хранилище данных обновляется и остаётся доступной даже при отсутствии Интернет-соединения. Все информационные функции приложения доступны в офлайн-режиме.

2.2 Функциональное моделирование мобильного приложения

С проектируемой системой взаимодействуют три основных пользователя:

– Клиент. В перечень его возможностей входят просмотр информации в мобильном приложении, пополнение баланса, подключение и отключение услуг;

– Представитель провайдера. В эту группу входят сотрудники технической поддержки и администратор, у которого есть возможность изменения информации рекламных сообщений, стоимости и условий тарифов и услуг и т.д.;

– Биллинг-система. Система получает платежи клиентов и возвращает на сервер БД информацию о транзакции. При этом с мобильным приложением биллинг-система напрямую не взаимодействует, посредником для неё выступает сервер приложения [3].

Составим таблицу сценариев использования (таблица 1) и диаграмму вариантов использования (рисунок 12), которые описывают взаимодействие пользователей с приложением.

Таблица 1 – Описание сценариев использования приложения

Название	Описание сценария	Инициатор
Аутентификация	Функция, реализующая проверку подлинности пользователя на основании введённого логина и пароля	Клиент
Просмотр состояния лицевого счёта	Функция, которая загружает информацию о транзакциях по лицевому счёту с сервера базы данных в локальное хранилище мобильного приложения	Клиент
Просмотр доступных тарифов и услуг	Реализуется посредством запроса к базе данных, в ответ на который пользователю отправляются доступные в данный момент тарифы и услуги	Клиент
Подключение/отключение услуги	Функция, реализующая отправку на сервер приложения запроса на подключение или отключение определённой услуги из списка	Клиент

Продолжение таблицы 1

Название	Описание сценария	Инициатор
Обращение в поддержку	Функция, отправляющая сообщение клиента на сервер приложения, где из него формируется заявка в техподдержку, которая принимается свободным оператором	Клиент
Пополнение лицевого счёта	Функция, которая осуществляет перенаправление клиента на веб-сервис системы обработки платежей, на котором пользователь может пополнить лицевой счёт	Клиент
Смена условий тарифа/услуги	Реализуется посредством внесения изменений в таблицы базы данных, содержащие информацию о тарифах и услугах	Провайдер
Добавление и удаление (блокировка) пользователей	Добавление в базу данных логинов и паролей (хешей паролей) новых абонентов, а также блокировка расторгнувших договор	Провайдер
Изменение содержания информационных баннеров	Заменяет содержание баннеров на актуальное, согласно сведениям базы данных	Провайдер

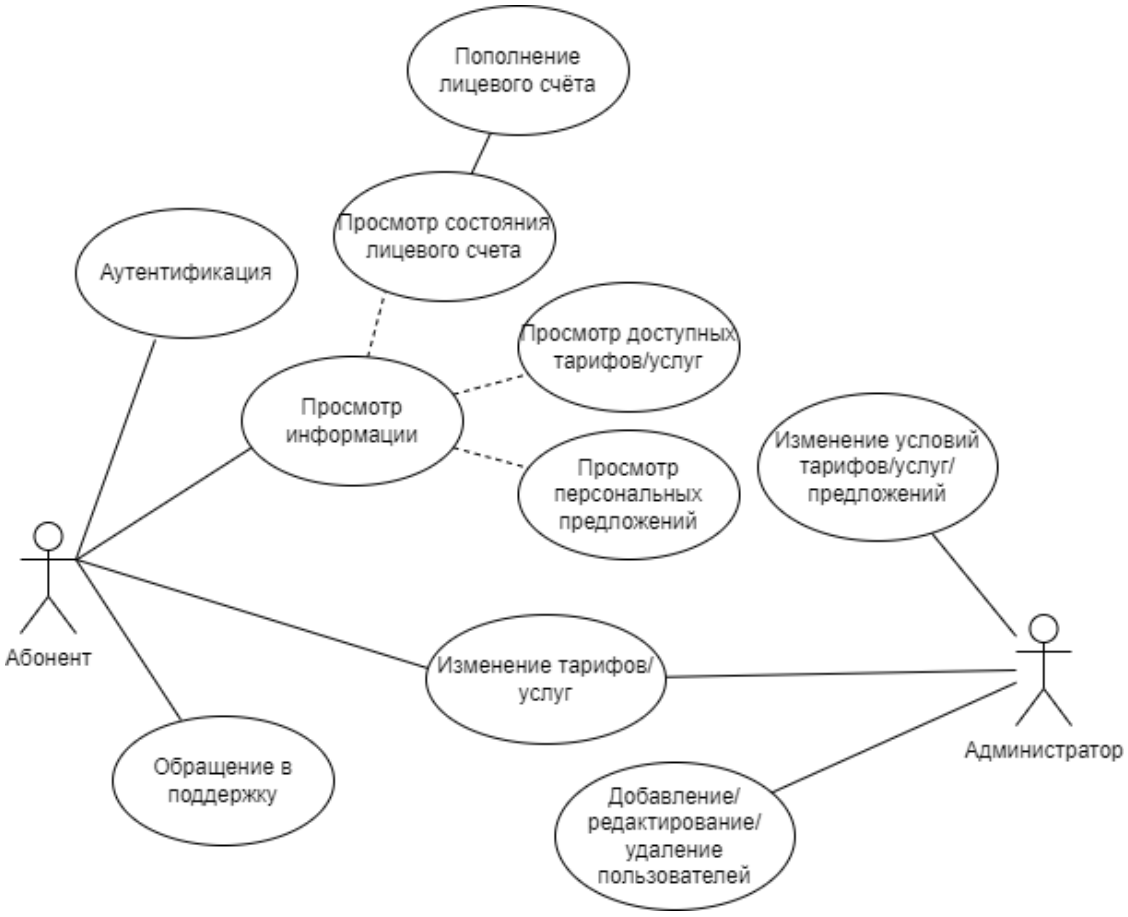

Рисунок 12 – Диаграмма вариантов использования

Рассмотрим подробнее основные пункты сценариев использования мобильного приложения клиентом (таблица 2).

Таблица 2– Сценарий «Просмотр состояния лицевого счёта»

Название: Просмотр состояния лицевого счёта
Условие выполнения: пользователь успешно прошёл этап аутентификации
Основной сценарий выполнения: 1. Пользователь запрашивает состояние лицевого счёта. 2. Мобильное приложение отправляет запрос на сервер приложения. 3. Сервер приложения формирует SQL-запрос и перенаправляет его на сервер БД. 4. Сервер БД отправляет ответ на запрос. 5. Сервер приложения перенаправляет ответ на запрос мобильному клиенту. 6. Мобильное приложение сохраняет полученные данные в кэш с меткой времени.
Альтернативный сценарий выполнения (при отсутствии Интернет-соединения): 1. Пользователь запрашивает состояние лицевого счёта. 2. Мобильное приложение загружает данные из кэша и уведомляет пользователя о необходимости Интернет-подключения для актуализации данных.

Указанный сценарий наглядно демонстрирует поведение мобильного приложения в условиях отсутствия Интернет-соединения [6]. Рассмотрим сценарий, требующий обязательного наличия подключения к сети (таблица 3).

Таблица 3 – Сценарий «Пополнение лицевого счёта»

Название: Пополнение лицевого счёта	
Условие выполнения: пользователь успешно прошёл этап аутентификации; устройство подключено к сети Интернет	
Основной сценарий выполнения: – Пользователь инициализирует пополнение лицевого счёта. – Мобильное приложение перенаправляет пользователя на сервис приёма платежей – Пользователь указывает данные дебетовой или кредитной карты – Платёжная система обрабатывает транзакцию и отправляет результат на сервер БД	
Транзакция одобрена Сервер базы данных вносит изменения в таблицу со сведениями о лицевом счете клиента Мобильное приложение сообщает пользователю об успехе операции Мобильное приложение инициирует сценарий «Просмотр состояния лицевого счёта»	Транзакция отклонена Сервер базы данных отправляет сообщение об ошибке транзакции на сервер приложения Мобильное приложение сообщает пользователю об ошибке операции
Альтернативный сценарий выполнения (при отсутствии Интернет-соединения): Не предусмотрено	

Таким образом, мобильное приложение опирается в своей работе на два основных компонента – сервер приложения и реляционную БД, данные из которой постоянно загружаются мобильным приложением в память устройства. Для корректной работы мобильного приложения требуется разработать логическую модель взаимодействия компонентов системы.

2.3 Разработка логической модели мобильного приложения

Логическая модель необходима для наглядного отображения логической связи между объектами проектируемой системы в зависимости от их свойств и атрибутов. Другими словами, логическая модель описывает взаимоотношения между различными сущностями системы формальным языком, сосредотачиваясь на детальном описании объектов, их свойств и взаимосвязей друг с другом [32]. Для разработки логической модели используется язык визуального моделирования UML.

UML – это стандартный язык для определения, визуализации, конструирования и документирования программных систем. UML не является языком программирования, но его инструменты могут использоваться для генерации кода на разных языках с использованием диаграмм UML. В основу UML положена объектно-ориентированная концепция проектирования информационных систем, реализуемая с помощью набора диаграмм. [5]

Логическое взаимодействие элементов приложения представлено на рисунке 13.

Поскольку в рамках данной выпускной квалификационной работы рассматривается разработка именно мобильного приложения абонента Интернет-провайдера, опустим элементы сервера приложения и сосредоточим внимание на реализации клиентской части и базы данных, используемых в работе клиентского мобильного приложения.

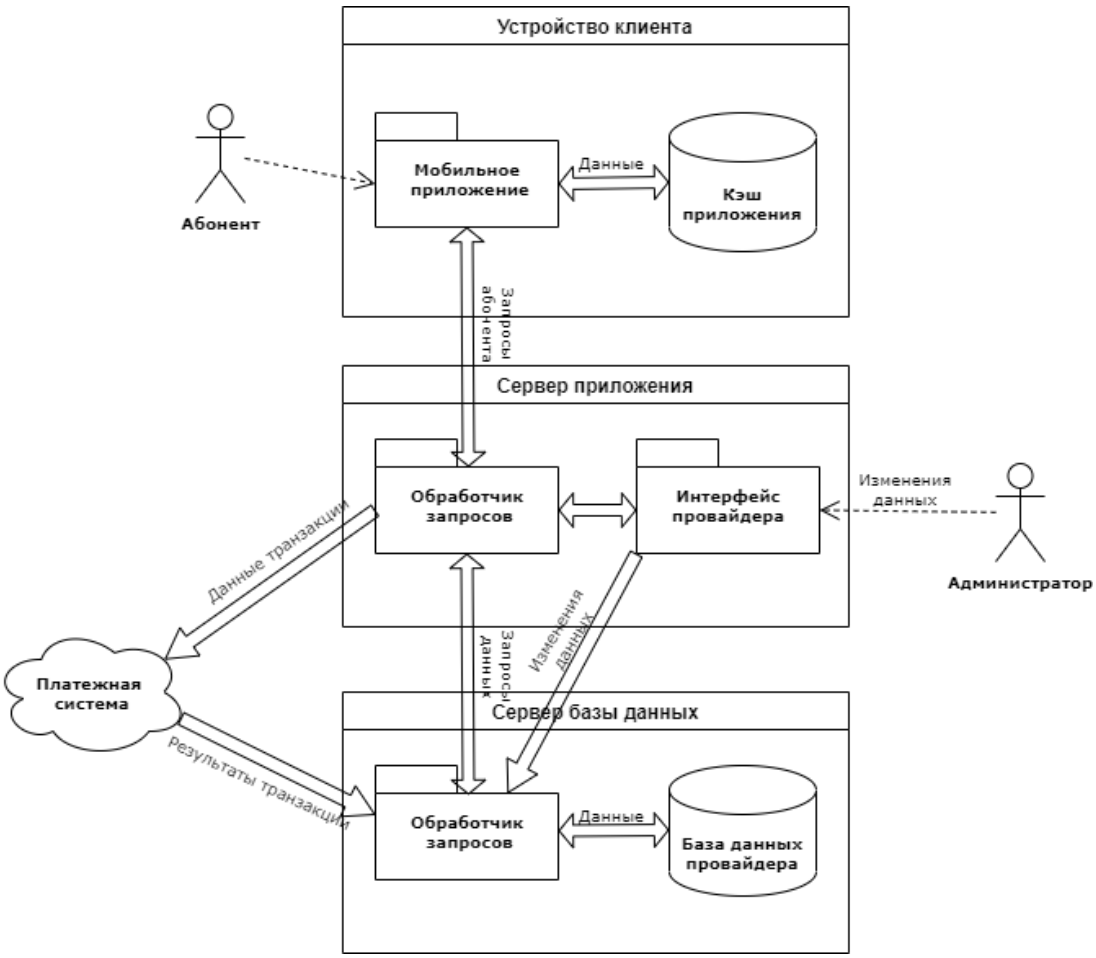

Рисунок 13 – Взаимодействие компонентов системы

Для уточнения процесса обмена данными между абонентом и базой данных необходимо разработать логическую структуру базы данных, используемых в приложении [8].

2.4 Разработка логической структуры базы данных мобильного приложения

Для понимания логики работы приложения с данными, используемыми в процессе, необходимо разработать логическую схему представления данных. Такая схема включает в себя элементы данных приложения и логические связи между ними. Логическая структура базы данных позволяет определить, как

пользователь и приложение взаимодействуют с каждым типом данных, используемых в приложении, а также то, в каком виде они хранятся.

Схема данных приложения представлена на рисунке 14 [20].

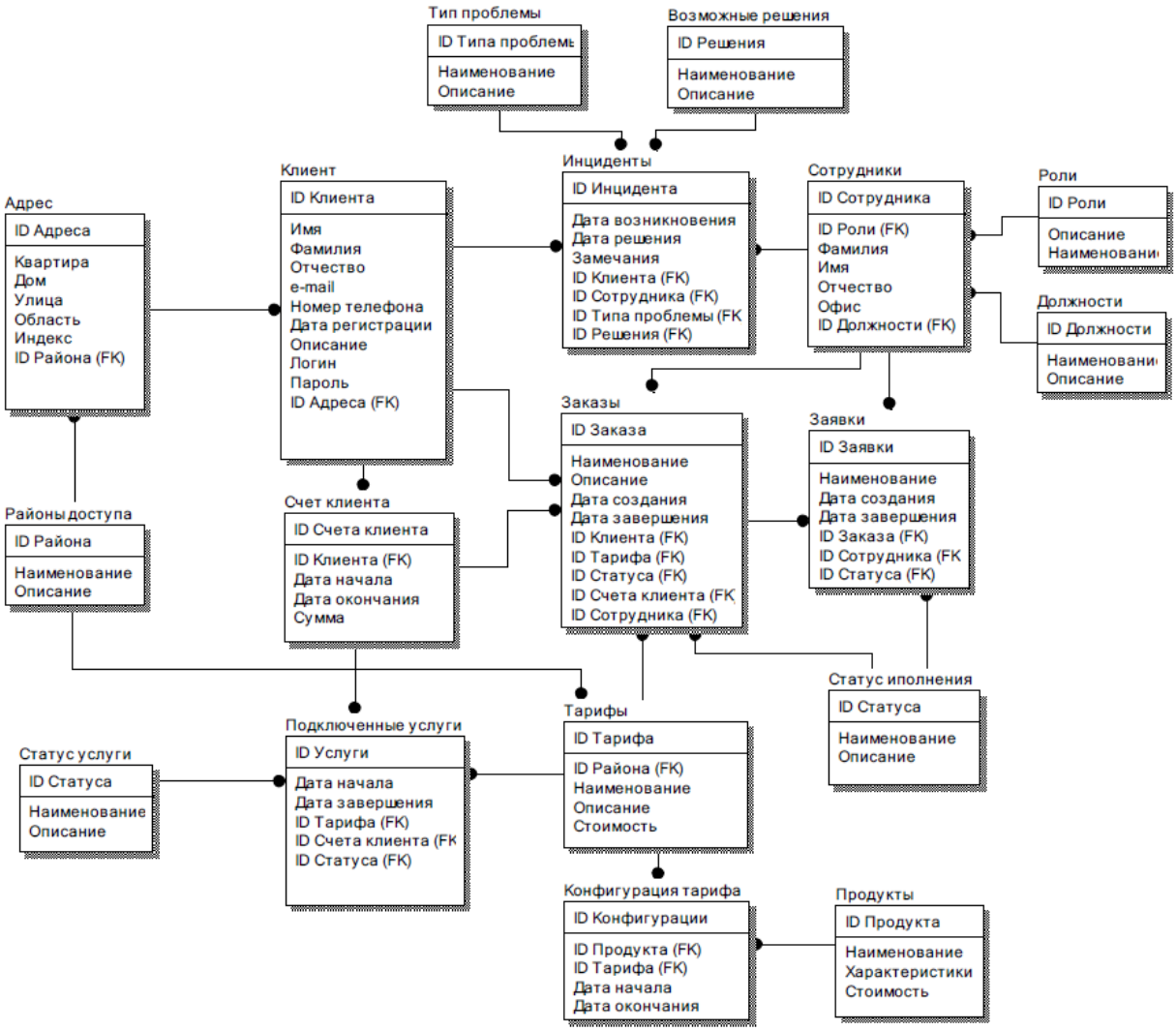

Рисунок 14 – Логическая структура базы данных мобильного приложения.

С базой данных работает как клиент, так и сотрудники провайдера, поэтому основная сущность базы данных – это Клиент. Благодаря ей, администраторы и сотрудники поддержки могут понимать, кто именно взаимодействует с ними. Сущности Счёт, Услуги и Адрес предоставляют дополнительную информацию о клиенте как самому пользователю, так и сотрудникам провайдера.

Сущности Заказы и Инциденты определяют все взаимодействия клиента и провайдера: Заказы обеспечивают смену тарифов, подключение дополнительных услуг и другие платные операции, а Инциденты фиксируют случаи обращения в службу поддержки [7].

Стоит отметить, что для сотрудников поддержки существует «база знаний» в виде готовых решений типовых проблем для облегчения решения задач. Сущность Сотрудники представляет собой список сотрудников компании-провайдера с обозначением ролей в системе. Доступные тарифы хранятся в соответствующей сущности и имеют региональную зависимость: доступность тарифа зависит от района расположения абонента.

В физической реализации база данных будет выглядеть следующим образом (рисунок 15).

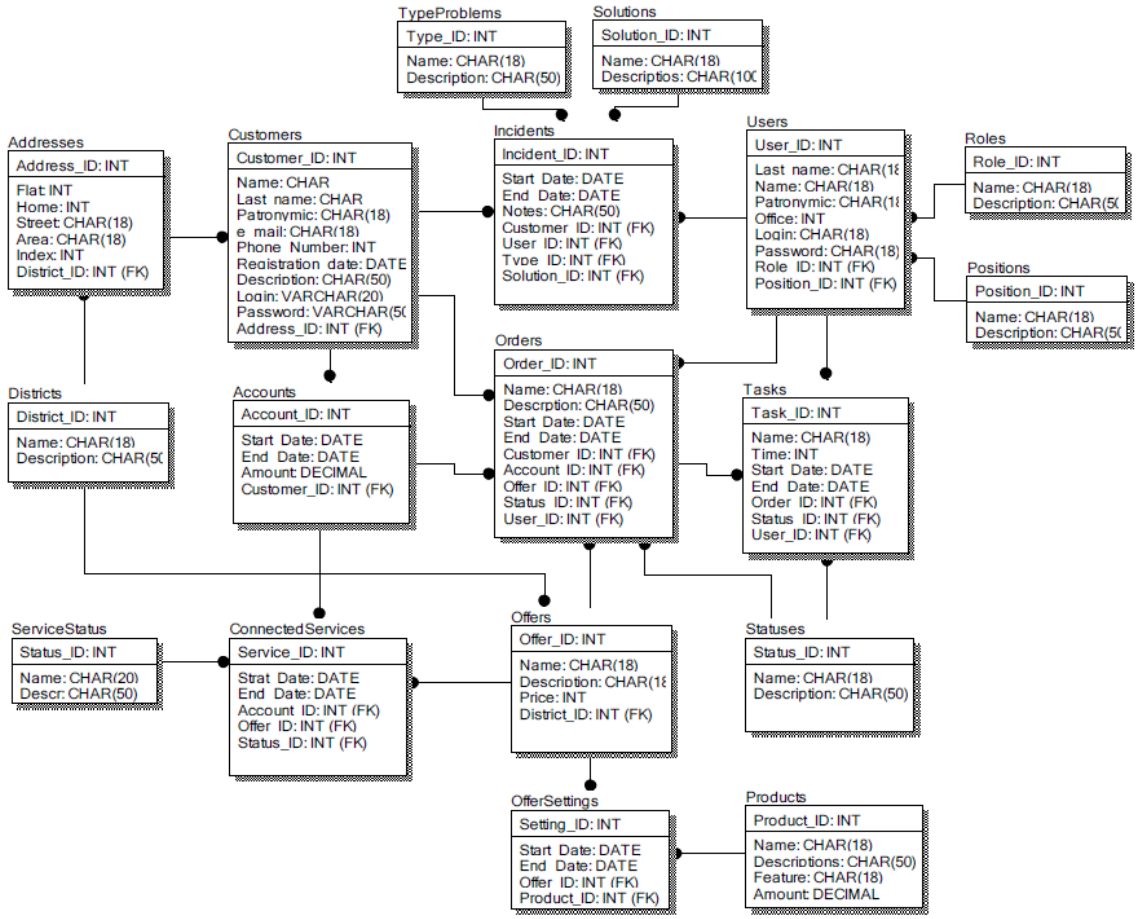

Рисунок 15 – Физическая реализация модели базы данных

Для управления базой данных мы будем использовать СУБД MySQL. MySQL – свободная реляционная система управления базами данных. MySQL надежный, производительный и функциональный. Благодаря множеству функций MySQL эта база данных является отличной системой управления баз данных. Скорость - одна из его характерных черт [6].

Выводы по Главе 2

Во второй главе была определена оптимальная архитектура для разрабатываемого мобильного приложения на основании функциональных требований к нему. Затем была спроектирована схема взаимодействий всех компонентов мобильного приложения, обосновано использование выбранной схемы для обеспечения стабильной работы компонентов приложения.

Также была сформирована функциональная схема работы приложения и сценарии использования. Сценарии использования рассматривают выполнение функций приложения при наличии и отсутствии у абонента Интернет-соединения, что повышает общую функциональность разрабатываемого приложения.

Затем была построена логическая модель приложения для наглядного отображения связей всех объектов и сущностей, взаимодействующих в рамках разрабатываемой структуры.

Кроме этого, была спроектирована база данных, используемая приложением, как в логическом, так и в физическом виде, были определены сущности базы данных, связи между ними и выбран инструмент управления базой данных.

Глава 3 Реализация мобильного приложения абонента Интернет-провайдера

3.1 Выбор средств разработки мобильного приложения

Проведём анализ инструментов разработки приложений на выбранной платформе.

Разработчиками Android рекомендуется использование IDE Android Studio, плод сотрудничества компаний Google и JetBrains. Android Studio является официальным мультиплатформенным средством разработки Android-приложений, получает регулярную поддержку и обновления, именно в ней первыми появляются новые возможности обновлённых версий OS Android. Данная IDE поддерживает разработку на Java, C++ и Kotlin, который позиционируется как приоритетный для разработки приложений на Android. Также в Android Studio доступно множество шаблонов для реализации типовых приложений, инструменты глубокого анализа и рефакторинга кода и быстрой компиляции.

Следующий инструмент разработки создан так же компанией JetBrains. IntelliJ IDEA – IDE с поддержкой многих языков программирования, в том числе упомянутые Java и Kotlin. К плюсам этой среды относится интуитивный интерфейс, возможность реализации различных компонентов продукта (мобильное приложение, веб-приложение, серверная часть) в рамках одной IDE, вне зависимости от используемых языков разработки [11]. Кроме этого, присутствует поддержка создания интерфейса приложений удобным drag-and-drop конструктором и широкий набор плагинов для реализации большинства функций. IntelliJ IDEA представлена в двух вариантах: Community Edition, с открытым исходным кодом, и Ultimate, с расширенной поддержкой и дополнительным функционалом для веб-разработки и крупных предприятий.

Среди наиболее простых вариантов разработки Android приложений можно выделить Appery.io. Этот инструмент позиционируется как

возможность создать полноценное веб или мобильное кроссплатформенное приложение без непосредственной работы с кодом. Вся разработка максимально визуализирована: функции и процессы собираются из блоков посредством drag-and-drop интерфейса, контекстная справка разъясняет все трудности восприятия, а для доработки приложения доступен редактор исходного кода [23].

Для разработки мобильного приложения Интернет-провайдера будет использоваться IDE Android Studio, поскольку она предоставляет наиболее широкий спектр инструментов тестирования и отладки кода непосредственно в процессе разработки, а также удобную эмуляцию операционной системы Android и встроенный редактор UI [34]. Кроме того, большинство функциональных требований приложения покрываются плагинами, доступными именно в этой IDE.

3.2 Программная реализация мобильного клиента приложения

Для авторизации пользователя в приложении используется две отдельные функции: одна для подключения в случае наличия Интернет-соединения, другая – для подключения в оффлайн-режиме.

Функция онлайн-аутентификации осуществляется с использованием сторонних плагинов auth0. При практическом применении приложения, соответственно, данные функции должны быть переписаны на систему авторизации провайдера. В текущей реализации функция аутентификации пользователя представлена далее (листинг 1).

Листинг 1 - Программный код функции авторизации пользователя

```
private fun login() {
    WebAuthProvider
        .login(account)
```

```
        .withScheme(getString(R.string.com_auth0_scheme))
        .withScope(getString(R.string.login_scopes))
        .withAudience(getString(R.string.login_audience,
      getString(R.string.com_auth0_domain)))
        .start(this, object : Callback<Credentials, AuthenticationException> {
          override fun onFailure(exception: AuthenticationException) {
            showSnackBar(getString(R.string.login_failure_message, exception.getCode()))
          }
          override fun onSuccess(credentials: Credentials) {
            cachedCredentials = credentials
            showSnackBar(getString(R.string.login_success_message, credentials.accessToken))
            updateUI()
            showUserProfile()
          }
        })
  }
```

Метод login() использует класс WebAuthProvider, который позволяет приложению использовать средства аутентификации auth0 [25]. В методе присутствуют следующие функции:

– .login() инициирует процесс и определяет аккаунт, используемый приложением.

– .withScheme() определяет, куда будет перенаправлен пользователь после аутентификации, а именно протокол (http или https), веб-страницу или часть приложения.

– .withScope() определяет, какую именно информацию о пользователе приложение может использовать после успешной аутентификации, в том числе: openid – id пользователя; profile – базовая информация о пользователе (фамилия, имя, логин, фотография и т.д.); email – электронная почта пользователя; read:current_user – флаг, обеспечивающий для приложения доступ только в режиме чтения.

– .withAudience() определяет, через какой url приложение проводит соединение до сервиса auth0.

– .start() объединяет все получившиеся в результате работы параметры, создаёт на их основе объект WebAuthProvider и открывает страницу аутентификации.

Визуально страница авторизации пользователя представлена на рисунке 16.

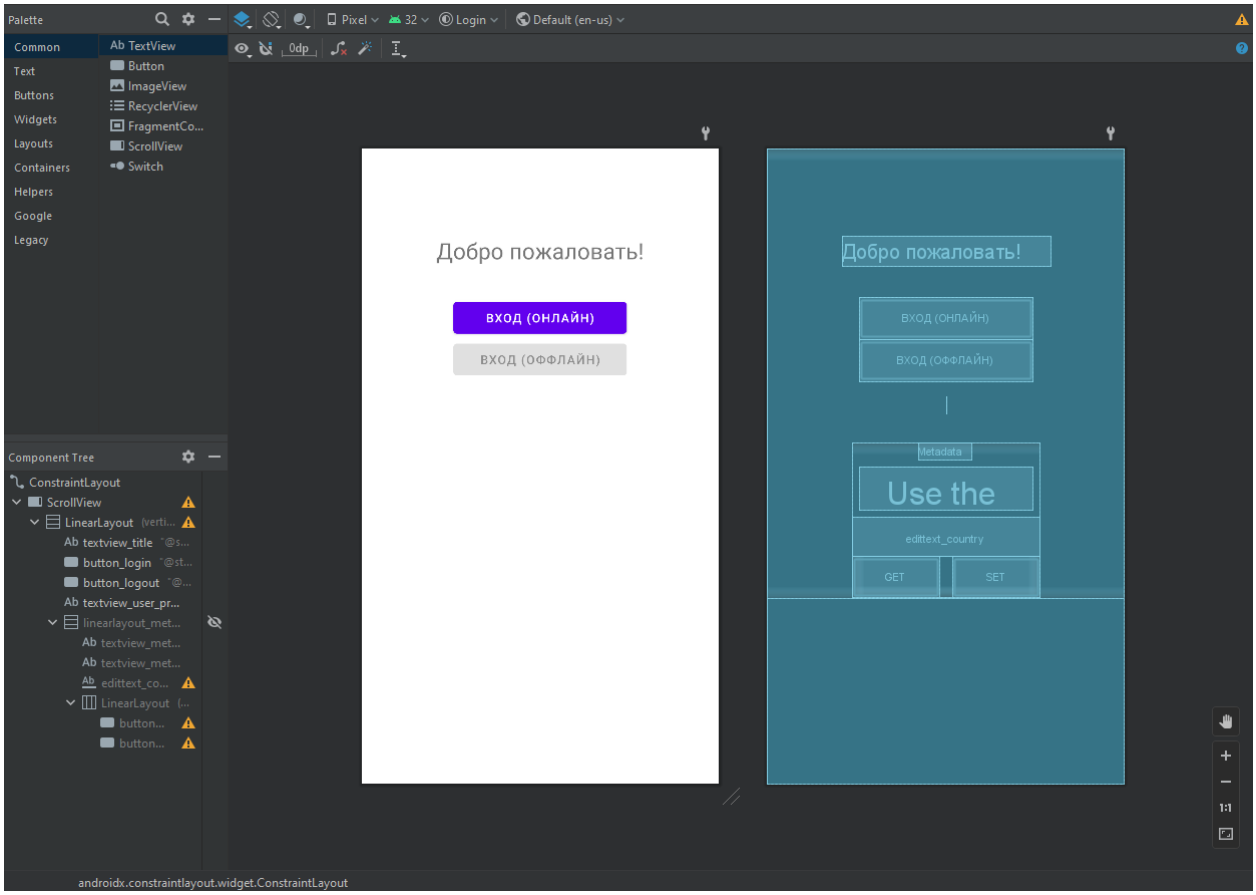

Рисунок 16 – Графическое отображение стартового экрана приложения

Таким образом, нажатие на кнопку «Вход (онлайн)» открывает веб-страницу, где пользователь вводит свои учётные данные (рисунок 17). При успешной аутентификации, приложение сохраняет данные в кэш в целях последующего использования при входе.

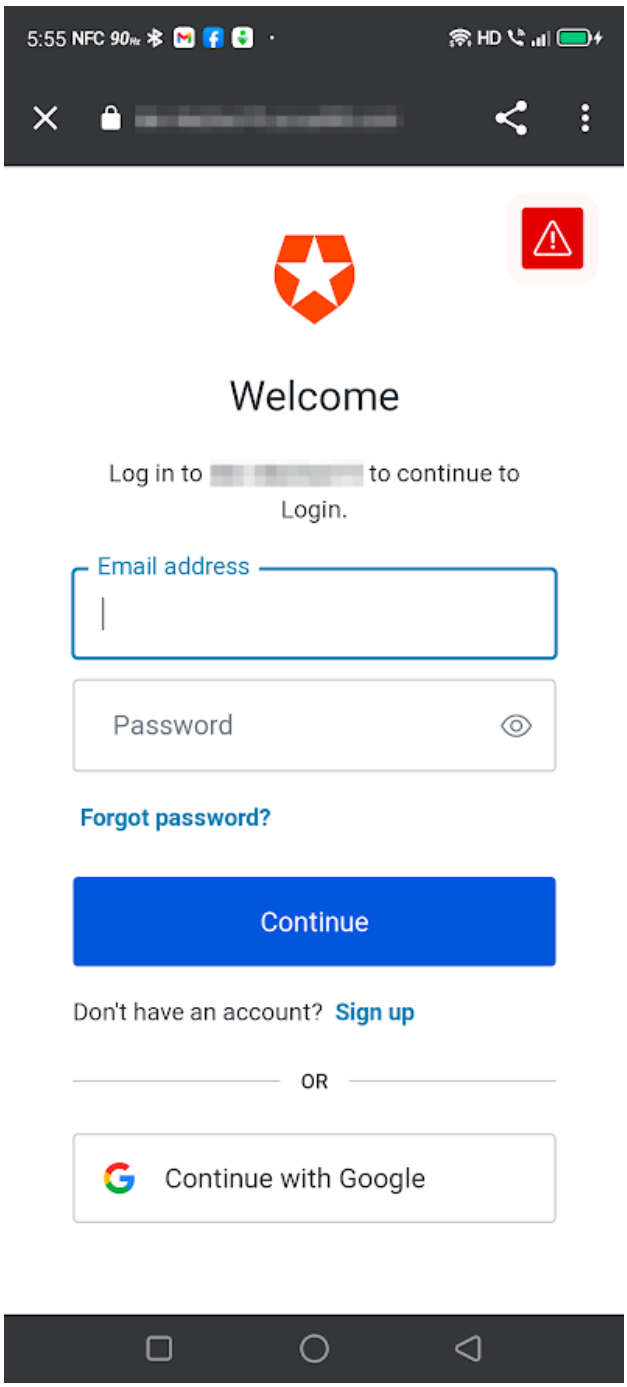

Рисунок 17 – Веб-версия формы аутентификации

Далее включается в работу метод getUserMetadata() (листинг 2), который выясняет, какие свойства присущи текущему пользователю, например, тариф пользователя, адрес, дополнительные услуги, персональные рекламные предложения и т.д. Графическое представление метода изображено на рисунке 18.

Листинг 2 - Программный код функции получения данных пользователя

```
private fun getUserMetadata() {
   if (cachedCredentials == null) {
      return
   }
   val usersClient = UsersAPIClient(account, cachedCredentials!!.accessToken!!)
   usersClient
      .getProfile(cachedUserProfile!!.getId()!!)
      .start(object : Callback<UserProfile, ManagementException> {
         override fun onFailure(exception: ManagementException) {
showSnackBar(getString(R.string.general_failure_with_exception_code,
               exception.getCode()))
         }
         override fun onSuccess(userProfile: UserProfile) {
            cachedUserProfile = userProfile
            updateUI()
            val country = userProfile.getUserMetadata()["country"] as String?
            binding.edittextCountry.setText(country)
         }
      })
}
private fun setUserMetadata() {
   if (cachedCredentials == null) {
      return
   }
   val usersClient = UsersAPIClient(account, cachedCredentials!!.accessToken!!)
   val metadata = mapOf("country" to binding.edittextCountry.text.toString())
   usersClient
      .updateMetadata(cachedUserProfile!!.getId()!!, metadata)
      .start(object : Callback<UserProfile, ManagementException> {
         override fun onFailure(exception: ManagementException) {
showSnackBar(getString(R.string.general_failure_with_exception_code,
```

```
                exception.getCode()))
            }
            override fun onSuccess(profile: UserProfile) {
                cachedUserProfile = profile
                updateUI()
                showSnackBar(getString(R.string.general_success_message))
            }
        })
    }
```

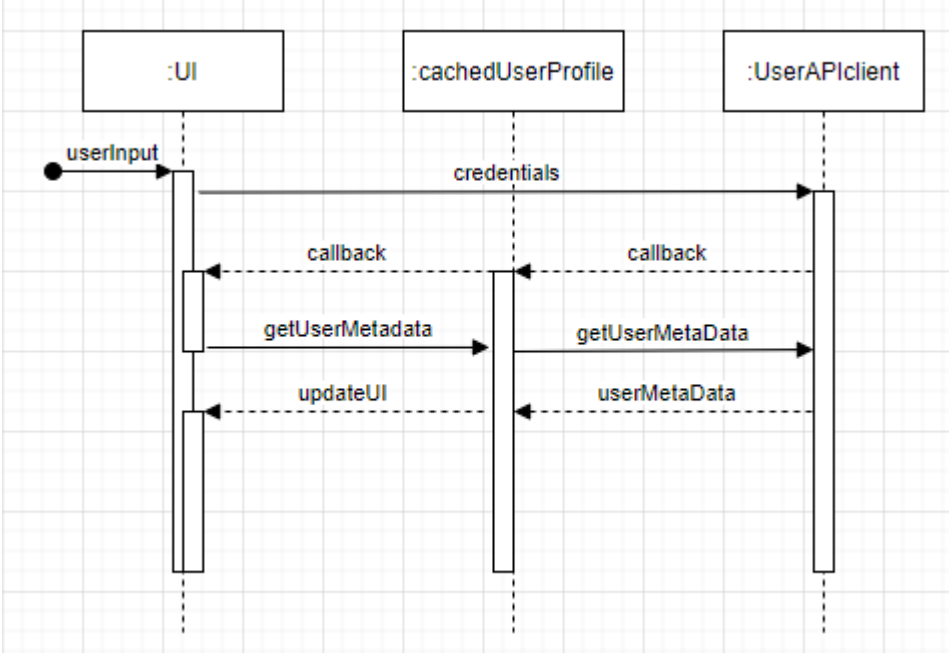

Рисунок 18 – Диаграмма последовательности метода getUserMetadata

Все данные сохраняются в кэш и используются для отображения интерфейса приложения при аутентификации пользователя в офлайн-режиме. Стоит отметить, что каждая функция изначально проверяет наличие сохранённых данных пользователя, чтобы избежать ошибок, связанных с отсутствием активного пользователя.

Следующий метод отвечают за обновление графического интерфейса приложения после успешной аутентификации (листинг 3).

Листинг 3 - Программный код функции обновления интерфейса

```
private fun updateUI() {
    val isLoggedIn = cachedCredentials != null

    binding.textviewTitle.text = if (isLoggedIn) {
        getString(R.string.logged_in_title)
    } else {
        getString(R.string.logged_out_title)
    }
    binding.buttonLogin.isEnabled = !isLoggedIn
    binding.buttonLogout.isEnabled = isLoggedIn
    binding.linearlayoutMetadata.isVisible = isLoggedIn
    binding.textviewUserProfile.isVisible = isLoggedIn
    val userName = cachedUserProfile?.name ?: ""
    val userEmail = cachedUserProfile?.email ?: ""
    binding.textviewUserProfile.text = getString(R.string.user_profile, userName, userEmail)
    if (!isLoggedIn) {
        binding.edittextCountry.setText("")
    }
}
private fun showSnackBar(text: String) {
    Snackbar
        .make(
        binding.root,
        text,
        Snackbar.LENGTH_LONG
    ).show()
}
```

При помощи этих методов в совокупности с getUserMetadata() отрисовывается интерфейс приложения в зависимости от того, прошёл ли

пользователь аутентификацию онлайн или офлайн. Офлайн-аутентификация использует кэшированные данные, полученные ранее в ходе работы приложения. После успешной работы метода updateUI() пользователь видит главный экран приложения с его основными функциями (рисунок 19).

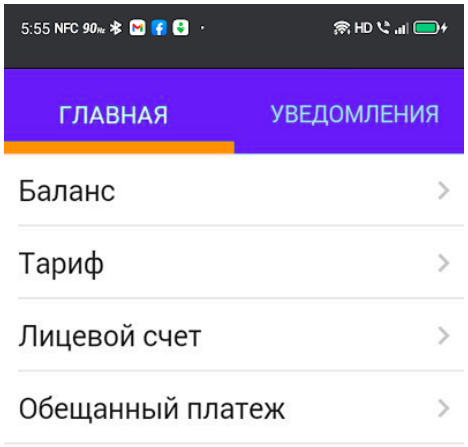

Рисунок 19 – Фрагмент главного экрана приложения

Таким образом, удалось разработать приложение с функцией аутентификации, не зависящей от наличия у абонента Интернет-соединения в конкретный временной момент использования приложения. Часть функционала может быть доступна офлайн, а аутентификация в этом случае происходит за счёт данных кэша.

3.3 Программная реализация использования сторонних платёжных сервисов

Одна из важнейших функций мобильного приложения со стороны клиента – возможность пополнения баланса лицевого счёта. Для реализации этой функции мы используем форму WebView, которая отображает безопасную страницу платёжной системы внутри приложения, не являясь его частью [32]. Таким образом, пользователь передаёт информацию о способе оплаты напрямую на сервер платёжной системы. Для защиты пользователя

также применяется 3DSecure. Конкретное использование функций WebView будет зависеть от API платёжной системы, но в общем случае реализация метода выглядит следующим образом (листинг 4). Диаграмма последовательности процесса представлена на рисунке 20.

Листинг 4 - Реализация перенаправления на платёжную систему

```
private WebView mWebView;
  LinearLayout linNtWkOff;
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    linNtWkOff = findViewById(R.id.linNtWkOff);
    mWebView = (WebView) findViewById(R.id.webView);
  }
  @Override
  protected void onResume() {
    super.onResume();
    if (checkInternetConnection()){
      setNetWorkON();
    }else {
      setNetWorkOFF();
    }
    mWebView.setWebViewClient(new MyBrowser());
    WebSettings webSettings = mWebView.getSettings();
    webSettings.setJavaScriptEnabled(true);
    webSettings.setLoadsImagesAutomatically(true);
    mWebView.loadUrl("-----write your url------");
  }
  private class MyBrowser extends WebViewClient {
    @Override
```

```
    public boolean shouldOverrideUrlLoading(WebView view, String url) {
        view.loadUrl(url);
        return true;
    }
}
@Override
public boolean onKeyDown(int keyCode, KeyEvent event) {
    if (event.getAction() == KeyEvent.ACTION_DOWN) {
        switch (keyCode) {
            case KeyEvent.KEYCODE_BACK:
                if (mWebView.canGoBack()) {
                    mWebView.goBack();
                } else {
                    finish();
                }
                return true;
        }
    }
    return super.onKeyDown(keyCode, event);
}
public void setNetWorkON(){
    linNtWkOff.setVisibility(View.GONE);
    mWebView.setVisibility(View.VISIBLE);
}
public void setNetWorkOFF(){
    linNtWkOff.setVisibility(View.VISIBLE);
    mWebView.setVisibility(View.GONE);
}
private boolean checkInternetConnection() {
    ConnectivityManager manager = (ConnectivityManager)
      getSystemService(Context.CONNECTIVITY_SERVICE);
    NetworkInfo ni = manager.getActiveNetworkInfo();
    boolean isConnected;
```

```
        if (ni != null && ni.getState() == NetworkInfo.State.CONNECTED) {
            isConnected =true;
        } else {
            isConnected= false;
        }
        return isConnected;
    }
}
```

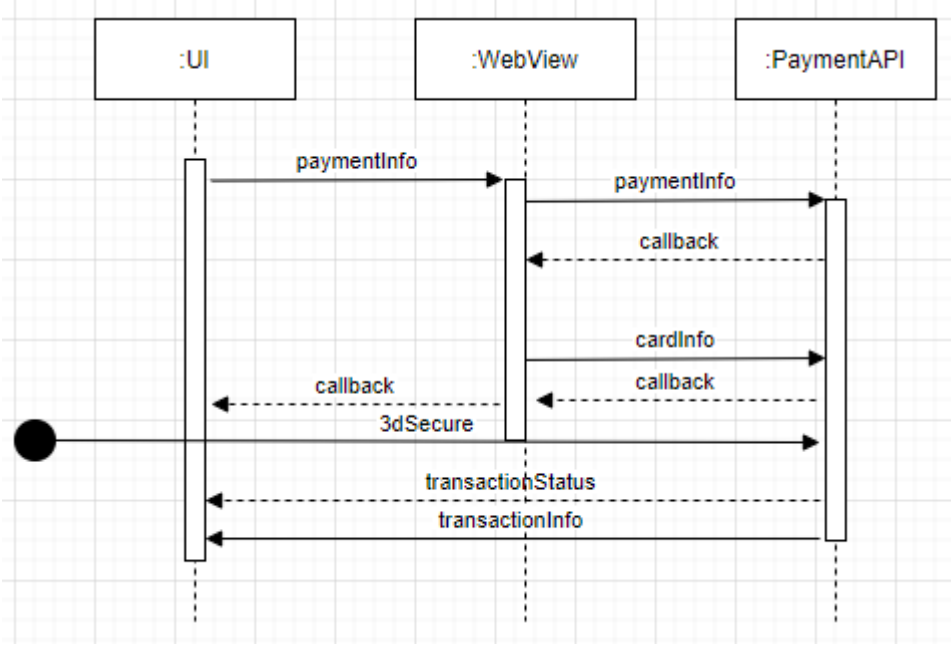

Рисунок 20 – Диаграмма последовательности реализации оплаты

Данная функция реализует отображение элемента WebView сайта платёжной системы с формой ввода платёжных данных (рисунок 21). По окончании обработки платежа инициируется возврат в приложение (листинг 5).

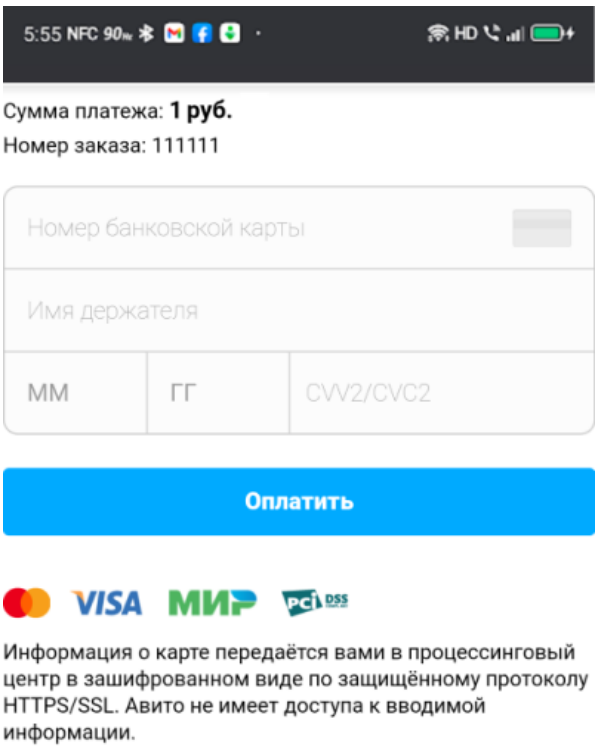

Рисунок 21 – Отображение страницы оплаты

Листинг 5 - Функция возврата в приложение

```
override fun onCreate(savedInstanceState: Bundle){
    super.onCreate(saveInstanceState)
    val action: String? = intent?.action
    if (action === android.content.Intent.ACTION_VIEW) {
        val data: Uri? = intent?.data
        val paymentId: String? = data.getQueryParameter("id")
    }
}
```

Функция возвращает id платежа, с которым приложение может работать в дальнейшем, а также значение успеха выполнения платежа. Для успешного и неуспешного платежа рекомендуется использовать перенаправление на разные страницы приложения.

Выводы по Главе 3

В третьей главе работы был проведён аналитический обзор средств разработки android-приложений, проведено сравнение наиболее популярных IDE и определён инструментарий для дальнейшей реализации функций приложения.

Далее были реализованы базовые функции приложения, проверена его работоспособность и описаны технически нюансы реализации функций аутентификации и отображения интерфейса приложения.

Также описано использование элементов WebView для реализации платежей посредством мобильного приложения и техническая реализация в общем виде, поскольку частная реализация использует API платёжной системы и получателя платежей. Данных такого характера в открытом доступе нет, поэтому частные случаи реализации рассмотрены не были.

В результате работы над ВКР был получен опыт проектирования и разработки мобильных приложений для операционной системы android, опыт использования языка Kotlin и использования сторонних API для реализации функций приложения.

Заключение

В ходе выполнения бакалаврской работы были сформированы требования к проектируемому приложению на основе анализа предметной области и обзора услуг, предоставляемых Интернет-провайдером. Требования были представлены согласно методологии FURPS+ и разделены на функциональные и нефункциональные. Также был проведён анализ популярности мобильных платформ для дальнейшей разработки мобильного приложения. В результате анализа была выбрана платформа Android, поскольку реализация мобильного приложения для неё потенциально затронет большее число абонентов Интернет-провайдера.

Затем было проведено проектирование архитектуры мобильного приложения, в результате которого выбор был остановлен на реализации распределённого приложения с четырёхзвенной архитектурой типа «клиент-сервер». Далее в ходе работы была реализована функциональная и логическая модели приложения, а также логическая и физическая структура базы данных, которая используется в приложении.

Последняя глава работы посвящена практической реализации проектируемых функций и техническим особенностям их исполнения при использовании IDE Android Studio, выбранной на основе анализа инструментов разработки мобильных приложений.

В результате работы были выполнены задачи, поставленные изначально, и достигнута цель – создано функциональное мобильное приложение с возможностью работы как при наличии Интернет-соединения, так и без него. В ходе разработки приложения были получены навыки проектирования функциональных и логических структур приложений, практические навыки работы с языком Kotlin и средой разработки Android Studio. Дальнейшая работа в выбранном направлении позволит создать полноценный коммерческий продукт, пригодный для извлечения прибыли.

Список используемой литературы

1. Афонин, В.В. Моделирование систем [Электронный ресурс] : учебно-практическое пособие / В.В. Афонин, С.А. Федосян. – Москва : Бином, 2016. – 231 с. – ISBN 978-5-9963-0352-6.

2. Балдин, К. В. Информационные системы в экономике [Электронный ресурс] : учебник / К. В. Балдин, В. Б. Уткин. – 7-е изд. – Москва : Дашков и К°, 2012. – 395 с. – ISBN 978-5-394-01449-9.

3. Буренин, С. Н. Web-программирование и базы данных [Электронный ресурс] : учеб. практикум / С. Н. Буренин. – Москва : Моск. гуманит. ун-т, 2014. – 120 с. – ISBN 978-5-906768-17-9.

4. Зарипов, А.К. Сравнение времени отклика репликации mysql с mongodb // Вестник магистратуры, 2021. №5-6 (116).

5. Каннет, М. Основы Scrum. Практическое руководство по гибкой разработке ПО / М. Каннет. – Москва : Вильямс, 2016. – 544 с.

6. Коберн, А. Современные методы описания функциональных требований к системам / А. Коберн. – Москва : Лори, 2012. – 264 с.

7. Корнипаев, И. Требования для программного обеспечения: рекомендации по сбору и документированию / И. Корнипаев. – Москва : Издательство «Книга по требованию», 2013. – 118 с. 15.

8. Лафорте, Р. Структуры данных и алгоритмы в Java / Р. Лафорте. – Санкт-Петербург : Питер, 2016. – 704 с.

9. Леоненков А.В. Объектно-ориентированный анализ и проектирование с использованием UML и IBM Rational Rose. [Электронный ресурс] : учебное пособие / А.В. Леоненков. Москва, Саратов : ИНТУИТ, Вузовское образование, 2017. 318 с. – URL: http://www.iprbookshop.ru/67388.html (дата обращения: 12.02.2022).

10. Логинов, В.Н. Информационные технологии управления: учеб. пособие / В.Н. Логинов. – Москва : КноРус, 2013. – 240 с.

11. МакГрат, М. Программирование на Java для начинающих / М. МакГрат. – Москва : Эксмо, 2016. – 193 с.

12. Максимов, Н.В. Информационные технологии в профессиональной деятельности: учебное пособие / Н.В. Максимов, Т.Л. Партыка, И.И. Попов. – Москва : Форум, 2010. – 496 с.

13. Максимов, Н.В. Современные информационные технологии: учеб. пособие / Н.В. Максимов, Т.Л. Партыка, И.И. Попов. – Москва : Форум, 2013. – 512 с.

14. Мельников, В.П. Информационные технологии: Учебник для студентов высших учебных заведений / В.П. Мельников. – Москва : ИЦ Академия, 2009. - 432 с.

15. Остервальдер, А. Построение бизнес-моделей. Настольная книга стратега и новатора / А. Остервальдер. – Москва : ООО «Альпина Паблишер», 2010. – 288 с.

16. Паклин, Н.Б. Бизнес-аналитика: от данных к знаниям / Н.Б. Паклин, В.И. Орешков. – Санкт-Петербург : Питер, 2013 – 706 с.

17. Перерва, А. Д. Путь аналитика. Практическое руководство IT-специалиста / А. Д. Перерва, В. А. Иванова. – 2-е изд., – Санкт-Петербург: Питер, 2005. – 304 с.

18. Петров, В.Н. Информационные системы / В.Н. Петров. – Санкт-Петербург: Питер, 2002. – 688 с.

19. Репин, В.В. Бизнес-процессы. Моделирование, внедрение, управление / В.В. Репин. – Москва : Манн, Иванов и Фербер, 2013. – 511 с.

20. Мартин, Р.К. Быстрая разработка программ. Принципы, примеры, практика / Р.К. Мартин, Д.В. Ньюкирк, Р.С. Косс – Москва : Вильямс, 2004. – 752 с.

21. Румянцева, Е.Л. Информационные технологии: учеб. пособие / Е.Л. Румянцева, В.В. Слюсарь. – Москва : ИД ФОРУМ, НИЦ ИНФРА-М, 2013. – 256 с.

22. Степанов, А.А. Основы представления бизнес-процессов: практикум для студентов вузов и участников программ дополнительного образования / А.А. Степанов. – ERPACADEMY.RU, 2015. – 18с.

23. Сьерра, К. Изучаем Java / К. Сьерра. – Москва : Эксмо, 2016. – 720 с.

24. Тихонов Э.Е. Информационные технологии в управлении: учеб. пособие. / Э.Е. Тихонов – Ставрополь : Изд-во СКФУ, 2015. – 402 с.

25. Федотова, Е.Л. Информационные технологии в науке и образовании : учеб. пособие / Е.Л. Федотова, А.А. Федотов. – Москва у: ИД ФОРУМ, НИЦ ИНФРА, 2013. – 336 с.

26. Федотова, Е.Л. Информационные технологии в профессиональной деятельности : учеб. пособие / Е.Л. Федотова. – Москва : ИД ФОРУМ, НИЦ ИНФРА-М, 2012. – 368 с.

27. Федотова, Е.Л. Информационные технологии и системы: учеб. пособие / Е.Л. Федотова. – Москва : ИД ФОРУМ, НИЦ ИНФРА-М, 2013. – 352 с.

28. Хлебников, А.А. Информационные технологии : учеб. пособие / А.А. Хлебников. – Москва : КноРус, 2014. – 472 с.

29. Шматалюк, А. Моделирование бизнеса / А. Шматалюк, М. Ферапонтов, А. Громов, М. Каменнова. – Москва : Весть-МетаТехнология, 2001. – 333 с.

30. Шилдт, Г. Java. Руководство для начинающих / Г. Шилдт. – Москва : Вильямс, 2012. – 624 с.

31. Baesens, B. Beginning Java Programming: The Object-Oriented Approach / B. Baesens, A. Backiel, S. Vanden Broucke. – 1st edition, Wrox, 2015.

32. Fernando Cejas. Architecting Android...The clean way? [Электронный ресурс] – URL: https://fernandocejas.com/2014/09/03/architecting-android-the-clean-way/ (дата обращения:16.01.2022).

33. Sathyan, J. Fundamentals of EMS, NMS and OSS/BSS / J. Sathyan – 1st edition, // Auerbach Publications, – 2010. – ISBN 978-1420085730.

34. Martin R. C., Clean Code: A Handbook of Agile Software Craftsmanship / R.C. Martin, // Prentice Hall. – 2008. – ISBN 978-0132350884.

35. Kendal S., Object Oriented Programming using C# / S. Kendal, // Bookboon. – 2015. – ISBN 978- 8776818142.

Приложение А

Требования к мобильному приложению

Таблица А.1 - Требования к мобильному приложению

№	Требование	Приоритет	Описание
Functionality			
Аутентификация пользователя		1	Необходимое требование для идентификации абонента и его лицевого счёта
Оплата услуг Интернет-провайдера		1	Приложение должно содержать возможность пополнения лицевого счёта
Просмотр истории платежей и зачислений		2	Позволяет абоненту планировать дальнейшие пополнения и списания, а также обеспечивает прозрачность расходов
Чат с технической поддержкой		1	Удобнее ряду клиентов, позволяет оптимизировать нагрузку на линии техподдержки провайдера
Активация и деактивация тарифов и услуг		1	Пользователь должен иметь возможность смены тарифа, подключения и отключения услуг
Просмотр доступных тарифов и услуг, а также их подробных условий		1	Страницы описания для каждой услуги и тарифа
Заказ обратного звонка технической поддержки		3	Запрос обратного звонка, содержащий время, в которое абоненту будет удобно принять звонок
Просмотр акций и рекламных предложений		5	Информационные страницы и баннеры
Приостановка действия услуг		2	Временная деактивация услуги или тарифа в рамках договора на период, например отпуска
Просмотр новостей и информационных сообщений		3	Информационные страницы, сообщения и баннеры о технических работах, проводимых провайдером, закрытии линеек тарифов и других важных изменениях
Usability			
Удобный и понятный интерфейс		1	Каждое действие должно быть интуитивно понятным, используются стандартные жесты для управления приложением и логичные пункты в меню
Контекстная справка		3	Для пунктов меню и отдельных действий возможна реализация вызова контекстной справочной информации
Комфортная цветовая гамма приложения		4	Цветовая гамма должна быть контрастной, но не напрягать глаза пользователя при длительной работе

Продолжение таблицы А.1

№	Требование	Приоритет	Описание
4	Читабельные и понятные шрифты	5	Весь текст должен быть удобен для чтения пользователем
Reliability			
Доступность системы 24/7		1	Приложение (хотя бы частично) должно быть доступно клиенту вне зависимости от внешних факторов (работа сервера, баз данных, интернет-соединение)
Автономность отдельных частей системы и их независимость от стабильности интернет-соединения		2	Большая часть информации хранится на устройстве абонента и подгружается с сервера провайдера при наличии Интернет-соединения, что позволит пользоваться справочными функциями приложения без активного доступа к сети
Низкая периодичность сбоев		2	Сбои в работе приложения не должны носить массовый и регулярный характер
Performance			
Время запуска приложения не более 3-х секунд		3	-
Время отклика приложения на действия пользователя не более 1 секунды		2	-
Supportability			
Адаптивная вёрстка приложения для корректного отображения на различных устройствах		2	Элементы интерфейса должны одинаково располагаться и быть доступными на различных устройствах вне зависимости от разрешения и соотношения сторон экрана
Возможность расширения функционала системы		2	При необходимости, функционал приложения может быть расширен за счёт использования доступных API
Корректная работа функционала на различных устройствах		1	Приложение должно стабильно работать на любых устройствах, соответствующих минимальным системным требованиям
Дополнительные требования и ограничения			
Получение информации из сторонних баз данных и сервисов Интернет-провайдера		-	Приложение получает необходимую информацию о лицевом счёте, доступных услугах и т.д. с сервера Интернет-провайдера
Реализация уведомлений приложения (push)		-	Приложение должно уведомлять абонента о важных событиях (необходимость оплаты счёта, технические работы и т.д.)