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

Кафедра «Прикладная математика и информатика»
(наименование)

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

Мобильные и сетевые технологии
(направленность профиля / специализации)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему: «Разработка мобильного приложения для расчета стоимости коммунальных услуг»

Обучающийся Е. Д. Киреев
(Инициалы Фамилия) (личная подпись)

Руководитель к.т.н., доцент, О.В. Аникина
ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

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

Тольятти 2024

Аннотация

Тема выпускной квалификационной работы: «Разработка мобильного приложения для расчёта стоимости коммунальных услуг».

Актуальность работы связана с растущей потребностью в расширении клиентской базы ООО «Квартплата 24» и в предоставлении клиентам удобных и практичных инструментов в целях повышения уровня сервиса и уменьшения нагрузки на операторов компании.

Целью выпускной квалификационной работы является разработка мобильного приложения для расчёта стоимости коммунальных услуг.

Во введении затронуты актуальность работы, методы, объект и предмет исследования, цель работы и задачи для достижения цели.

Глава 1 посвящена анализу деятельности компании ООО «Квартплата 24», описанию экосистемы сервисов компании, анализу конкурирующих приложений и формированию требований к новой технологии.

Глава 2 работы содержит постановку задачи на разработку мобильного приложения, логическое и физическое моделирование мобильного приложения, рассматриваются вопросы информационного обеспечения и выбора архитектуры приложения.

Глава 3 посвящена непосредственно разработке программного обеспечения, выбору средств разработки и базы данных. Описывается функционал и проводится тестирование приложения и оценка соответствия требованиям.

В заключении рассматриваются цель и задачи работы, описываются результаты исследования и экономическая эффективность приложения.

Бакалаврская работа состоит из 50 страниц и включает 24 рисунка, 1 таблицу, 20 источников.

Abstract

The title of the graduation work is «Development of a mobile application for calculating the cost of utilities».

The relevance of the work is related to the growing need to expand the client base of Kvartplata 24 LLC and to provide clients with a convenient and practical tools in order to increase the level of service and reduce the load on the company's operators.

The aim of the work is to is to develop a mobile application for calculating the cost of utilities.

The introduction touches on the relevance of the work, methods, object and subject of research, the purpose of the work and tasks to achieve the goal.

Chapter 1 is devoted to an analysis of the activities of the company "Kvartplata 24" LLC, a description of the ecosystem of the company's services, an analysis of competing applications and the formation of requirements for new technology.

Chapter 2 of the work contains a statement of the problem of developing a mobile application, logical and physical modeling of a mobile application, issues of information support and choice of application architecture are considered.

Chapter 3 is devoted directly to software development, the choice of development tools and databases. The functionality is described and the application is tested and assessed for compliance with requirements.

In conclusion, the purpose and objectives of the work are discussed, the results of the study and the economic efficiency of the application are described.

The senior paper consists of 50 pages and includes 24 figures, 1 table, 20 sources.

Оглавление

Введение

В современном мире, сфера жилищно-коммунального обслуживания стала всеохватывающей, практически каждый человек является её потребителем, и она является частью любого развитого государства. Как и в любой другой сфере деятельности, применение современных IT решений – ключ к предоставлению наиболее комфортного сервиса для оказания услуг и развитию бизнеса как такового.

Разработка мобильного приложения для расчета стоимости коммунальных услуг будет выгодным решением, поскольку принесет пользу и выгоду как компании ООО «Квартплата 24», так и её клиентам. В первую очередь приложение нацелено на повышение уровня сервиса компании, что в свою очередь принесет ей больше довольных клиентов и повышение экономической эффективности компании. Клиенты же получат удобный инструмент, с помощью которого они смогут рассчитывать стоимости коммунальных услуг что положительно скажется на впечатлении от работы компании, поскольку зачастую подобные сервисы доступны только на веб-сайтах, что может быть неудобным для некоторых пользователей.

Актуальность темы связана с растущей потребностью в расширении клиентской базы ООО «Квартплата 24» и в предоставлении клиентам удобных и практичных инструментов в целях повышения уровня сервиса и уменьшения нагрузки на операторов компании.

Объектом исследования бакалаврской работы являются изучение приемов создания мобильного приложения, облегчающего расчеты стоимостей предоставляемых услуг.

Предмет исследования является проектирование, разработка и тестирование мобильного приложения для расчета стоимости коммунальных услуг.

Целью выпускной квалификационной работы является разработка мобильного приложения для расчета стоимости коммунальных услуг компании ООО «Квартплата 24»:

Для достижения поставленной цели необходимо выполнить список следующих промежуточных задач:

- провести анализ организации ООО «Квартплата 24»;
- провести сравнительный анализ существующих аналогов подобного мобильного приложения, сформировать требования к мобильному приложению;
- проанализировать технологии разработки мобильного приложения и выбрать наиболее подходящие для решения поставленной цели;
- непосредственно разработать мобильное приложение для расчета стоимостей коммунальных услуг;
- провести тестирование мобильного приложения.

Методы исследования: литературный обзор, анализ конкурентов, классификация, моделирование, тестирование, изучение электронных источников.

Практическая значимость разработки мобильного приложения для расчёта стоимости коммунальных услуг проявляется в возможности применения его клиентами в повседневной жизни или же в компаниях для более удобного и прозрачного учета коммунальных расходов. Приложение позволит пользователям легко и точно рассчитывать платежи, а компании получат инструмент для автоматизации процесса учета и оптимизации взаимодействуя с клиентами.

Работа состоит из введения, трех глав, разделенных на параграфы, заключения, списка использованной литературы и приложения.

Глава 1 Анализ предметной области

1.1 Характеристика компании ООО «Квартплата 24»

ООО «Квартплата 24» - федеральная IT-компания, использующая современные технологии в сфере обслуживания ЖКХ. Компания является участником проекта «Сколково», а также резидентом технопарка в сфере высоких технологий «Жигулевская долина». Организация состоит в НП СРО «Национальный жилищный конгресс» и включена Минстроем России в Банки решений Умного города и эффективных технологий в ЖКХ.

Компания умеет решать сложные задачи, стремится к постоянному самосовершенствованию и с удовольствием принимает в свои ряды единомышленников. Организация предоставляет свои услуги на рынке услуг в сфере IT с 1996 года и обслуживает свыше одного миллиона лицевых счетов.

Квартплата 24 обеспечивает более 30 способов электронной оплаты, принимает платежи по всей территории Российской федерации и за ее пределами, а также является единственным в РФ платежным сервисом, который обеспечивает моментальный прием и расщепление платежей за жилищно-коммунальные услуги на всех этапах прохождения платежа.

Цель компании – сделать самую непрозрачную сферу деятельности в России удобной и приятной не только для организаций, но также и для жителей.

Клиенты компании – это более 800 ТСЖ, ЖСК, управляющих и ресурсоснабжающих организаций расположенный по всех территории Российской Федерации. Инструменты, применяемые в организации, позволяют клиентам гарантированно получать поддержку вовремя и на должном уровне. На рисунке 1 представлена организационная структура компании.

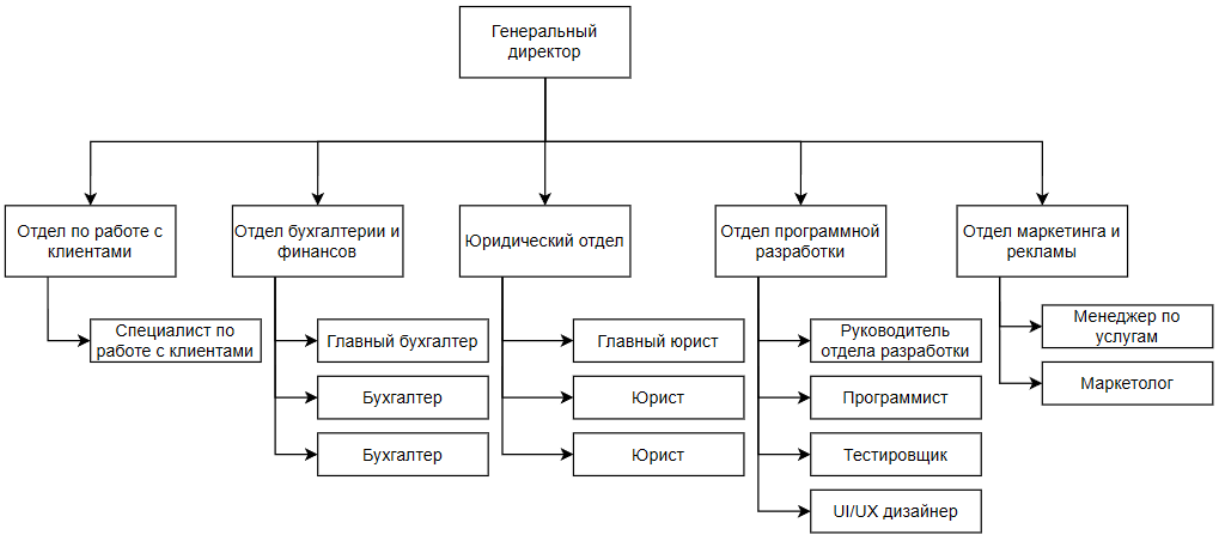


Рисунок 1 – Организационная структура компании ООО «Квартплата 24»

Рисунок отображает, что в компании существует несколько отделов, каждый из которых выполняет ту или иную функцию и решает не или иные задачи. Так, например, отдел программной разработки занимает разработкой программного обеспечения, которое в последствии будет использоваться сотрудниками компании для ускорения и упрощения процесса работы и повышения эффективности сотрудников или же клиентами для повышения качества сервиса предоставляемых услуг.

Данный отдел имеет прямое отношение к данной выпускной работе, поскольку в работе также разрабатываться прототип приложения, которое будут использовать потребители и клиенты компании.

1.2 Описание экосистемы организации

В компании разработана экосистема облачных сервисов, позволяющая автоматизировать расчеты и учеты оплат за жилищно-коммунальные услуги и сделать их удобнее. Система обеспечивает соответствие расчета законодательству и делает приемы платежей максимально прозрачными и при этом практически мгновенными.

Работая с экосистемой компании, клиенты экономят время и силы, система автоматически производит все сложные расчеты и вычисления за человека, а также формулирует необходимую отчетность. Наличие сервиса личного кабинета жителя делает жизнь миллионов людей гораздо проще и удобнее.

Экосистема, состоящая из более чем 10 облачных, внутренних и внешних сервисов, представляет из себя комплексное решение для расчета платы за жилищно-коммунальные услуги, распределения платежей, работы с дебиторской задолженностью, контроля деятельности организаций и формирования платежных документов.

Внешние сервисы предназначены для клиентского использования, внутренние же используются для реализации технологических процессов.

Основными сервисами экосистемы являются:

- личный кабинет жителя;
- сервис аналитики;
- прием платежей;
- сервисы для расчетов;
- сервис по работе с должниками;
- служба поддержки.

Подробное представление экосистемы сервисов показано на диаграмме на рисунке 2.

Рисунок 2 – Сервисы экосистемы в ООО «Квартплата 24»

Каждый из данных сервисов выполняет свою определенную функцию и при необходимости взаимодействует с другими, образуя организованную и гибкую систему, каждый сегмент которой выполняет конкретную задачу.

Например, личный кабинет пользователя включает в себя множество функций, доступных жителям в любое время, в любом месте и без очередей. Сервис предоставляет моментальный учет, удобство пользования и обеспечивает достоверность показаний. В свою очередь сервис по расчету коммунальных услуг, предоставляет перечень услуг, которые настраиваются индивидуального под задачи заказчика, имеет высокую степень автоматизации, высокую скорость расчета, выгрузку начислений и автоматический перерасчет при изменении данных.

1.3 Сравнительный анализ используемых аналогов

Перед разработкой нового приложения важно выполнить несколько подготовительных этапов, один из них - сравнительный анализ аналогов или же приложений со схожим функционалом. Рассмотрим несколько приложений, которые также предоставляют услуги в сфере ЖКХ, рассмотрим их плюсы и минусы, оценим все это, чтобы в последствии вобрать все лучшее в новое приложение. Так как данная разработка в первую очередь будет предназначаться для расчетов стоимостей коммунальных услуг, особое внимание нужно уделить именно реализации этого аспекта в приложениях.

Для анализа были рассмотрены следующие сервисы:

- Квартплата.Онлайн: Приложение для полного контроля коммунальных услуг, достоинствами которого являются возможность добавлять множество квартир в одном приложении, интуитивно понятный интерфейс и простота использования;
- Калькулятор ЖКХ: Онлайн сервис для расчета размера платы за жилищно-коммунальные услуги, удобный сервис, который не только предоставляет возможность рассчитать стоимость услуги, но также предоставляет формулы, используемые во время расчета;
- Портал ЖКХ: Личный кабинет для потребителей жилищных коммунальных услуг в Иркутской обрасти, включает в себя расчеты коммунальных платежей, формирование отчетов, а также уведомления о предстоящих платежах.

Проведем сравнительный анализ указанных приложений для выявления их достоинств и недостатков. Анализ конкретных аспектов приложения представлен значениями от 0 до 5, где 0 – это минимальная оценка, а 5 – максимальная оценка.

Таблица 1 – Сравнительный анализ аналогов

Наименование	Калькулятор ЖКХ	Квартплата.Онлайн	Портал ЖКХ
Удобство ввода данных	5	4	3
Отображение стоимости услуг	2	3	4
Скорость расчета услуг	3	4	5
Формирование отчета	2	5	3
Интуитивная навигация	4	2	3
Удобство дизайна	3	3	4
Отзывчивость и скорость работы приложения	5	5	3

Цель разработанного приложения – удобное решение в вопросе расчета стоимости коммунальных услуг. Приложение должно предоставлять пользователю возможность не только указывать показатели необходимые для расчета, но также редактировать саму формулу вычисления стоимости услуг.

Приложение должно иметь возможность указывать несколько объектов недвижимости и подключать к каждому из них те или иные услуги.

Не смотря на то что существуют данные приложения, они не содержат весь необходимый функционал, тем более не имеют возможности расширения и интерпретации их в экосистему компании, предполагается что в последствии приложение будет интерпретировано в систему сервисов и будет взаимодействовать с ними.

Подчеркнув плюсы и минусы каждого их этих приложений, мы можем использовать из в ходе разработки и при формировании требований.

1.4 Формирование требований для разрабатываемого приложения

Практически ко всем разрабатываемым приложением выдвигается список требований, которые должны быть учтены при разработке приложения, это делается для определения характеристик, которые должно иметь приложение, а также для обеспечения соответствия его конечного продукта ожиданиям пользователей.

Требования к разрабатываемым приложениям можно подразделить на две категории – функциональные и нефункциональные.

Функциональные требования – это требования к поведению системы, они определяют функциональность разрабатываемого программного обеспечения, т.е. описывают возможности, которые предоставляет система [7]. Функциональные требования определяют, как необходимо реализовать проект, включают в себя бизнес-требования и пользовательские требования.

На основе результатов изучения и сравнения используемых аналогов можно выявить следующие функциональные требования:

- мобильное приложение должно предоставлять возможность регистрации и авторизации;
- мобильное приложение должно предоставлять возможность выхода из учетной записи пользователя;
- мобильное приложение должно предоставлять возможность ввода и обработки показаний для расчёта стоимостей различных услуг;
- мобильное приложение должно предоставлять возможность занесения данных о нескольких объектах недвижимости;
- мобильное приложение должно предоставлять возможность составления отчета по всем услугам для всех объектов недвижимости.

Также, как и для любого другого приложения должны быть поставлены и нефункциональные требования.

Нефункциональные требования – это требования к характеру поведения системы, описывают как должна работать система, свойства и ограничения, накладываемые на разрабатываемое программное обеспечение.

Опишем основные нефункциональные требования, предъявляемые к приложению для расчета стоимости коммунальных услуг:

- приложение должно работать стабильно и без сбоев;
- архитектура приложения должна быть гибкой и включать в себя возможность добавления новых функций;
- время отклика интерфейса приложения должно быть минимальным для предоставления удобства пользователю;
- мобильное приложение должно работать только в режиме портретной ориентации экрана;

Также важно понимать возможные взаимодействия пользователя с приложением, для этого построим диаграмму вариантов использования. Для построения диаграммы воспользуемся онлайн сервисом Draw.io.

Draw.io – это бесплатный онлайн-сервис, который помогает создавать блок схемы, прототипы, инфографику и диаграммы любого вида. Для этого в Draw.io есть все необходимые элементы: блок-схемы, стрелки, таблицы — целые наборы фигур и знаков.

На диаграмме вариантов использования, представленной на рисунке 3, отображено предположительное взаимодействие пользователя с приложением.

Рисунок 3 – Диаграмма вариантов использования приложения

Пользователю должна быть предоставлена возможность добавлять новые объекты, редактировать их характеристики, а также подключать к ним новые виды услуг.

Выводы по главе 1

В первой главе ВКР, была разобрана деятельность компании ООО «Квартплата 24», а также разобрана её экосистема. Был проведен анализ аналогов, который показал основные характеристики, которые отличают аналоги друг от друга. По результатам анализа был выдвинут список требований к проектируемому приложению.

Глава 2 Проектирование мобильного приложения

2.1 Постанова задачи на разработку мобильного приложения

На сегодняшний день практически у каждого человека есть мобильное устройство, которым он пользуется в повседневности облегчая свою жизнь и решая сложные задачи возлагая их на мобильные приложения. Задача данной работы состоит в том, чтобы создать мобильное приложение, способное хранить в себе информацию о всех необходимых пользователю объектах недвижимости, их характеристики и подключенные к ним услугам, предоставлять пользователю возможность персонализации расчётных формул услуг. Приложение должно быть удобным для пользователя и простым в навигации, а также способным хранить в себе большие объемы данных для различных пользователей и выводить только соответствующую информацию.

Мобильное приложение – это программное приложение, предназначенное для работы на мобильном устройстве, таком как смартфон или планшет, предоставляющее пользователю определенные функции или услуги [6].

Преимущества мобильного приложения:

- повышение лояльности и вовлеченности клиентов;
- улучшенный пользовательский интерфейс и персонализация;
- конкурентное преимущество на рынке;
- повышенная доступность и удобство;
- повышенная эффективность и производительность;
- портативность и доступ к данным в любое время;
- изменение и сохранение данных в реальном времени;
- синхронизация данных с различными устройствами.

2.2 Выбор архитектуры мобильного приложения

После формирования требований для мобильного приложения и постановки задачи на разработку можно перейти непосредственно к формированию логической и физической модели приложения и выбору архитектуры. Архитектура мобильного приложения подразумевает под собой формирование структуры и взаимодействий компонентов приложения.

Функционал приложения подразумевает хранение данных пользователя в облачном сервисе, из которого в любой момент времени пользователь может получить данные на любом устройстве. В следствии этого разумно разработать клиент-серверное приложение и сформировать трехзвенную архитектуру, которая будет состоять из непосредственно мобильного приложения, промежуточной базы данных и сервера-обработчика запросов [15]. Таким образом мы сможем реализовать добавление, редактирование и удаление данных на удаленном сервере [4].

Сформируем множество сценариев поведения пользователей, чтобы понимать каким функционалом должно обладать приложение:

- пользователь скачал приложение впервые и регистрируется;
- пользователь скачал приложение на новое устройство и хочет авторизоваться в свой аккаунт;
- пользователь создает новый объект недвижимости и указывает его характеристики;
- пользователь хочет добавить новый вид услуги в объект;
- пользователь почет отредактировать характеристики объекта;
- пользователь хочет отвязать существующий вид услуги от объекта;
- пользователь хочет просмотреть итоговую стоимость услуг как по одному объекту, так и по всем.

Далее выделим функциональные блоки приложения и схему навигации между ними. В первую очередь должен быть реализованы блоки авторизации

и регистрации, они позволят пользователям создавать новые аккаунты и входить в уже существующие, это должна быть самая первая форма, которую увидит пользователь при первом заходе в приложение. Далее после регистрации, пользователь попадет в блок главной страницы приложения или же «Дашборд», на главной странице будет отображаться список всех созданных пользователем объектов недвижимости, а также кнопка создания нового объекта и кнопка редактирования профиля. Исходя из этого необходимо реализовать блоки создания и редактирования объектов недвижимости, в которых пользователь сможет указать все необходимые характеристики. Также необходимо выделить блок настроек аккаунта. В этом блоке пользователь сможет редактировать свой профиль или же какие-то системные настройки приложения.

Подводя итог из всего выше сказанного, перечислим список функциональных блоков приложения:

- авторизация и регистрация;
- главная страница приложения;
- создание и редактирование объекта недвижимости;
- создание и редактирование услуги;
- настройки аккаунта.

Также опишем структуру диалога мобильного приложения:

- окно входа и регистрации с переходом на главный экран;
- главный экран с возможность перехода на все остальные;
- экран создания объекта;
- экран просмотра объекта;
- экран редактирования объекта;
- экран настроек аккаунта.

2.3 Разработка и описание логической модели

Исходя из описанных выше сценариев взаимодействия пользователя и выделенных функциональных блоков опишем логическую модель приложения. Логическая модель подразумевает под собой графическое отображение того, как различные компоненты приложения взаимодействуют между собой при тех или иных сценариях пользователя.

Построим логическую схему работы пользователя с мобильным приложением, которая будет отражать выделенные блоки приложения и структуру диалога мобильного приложения. Схема логической работы приложения представлена на рисунке 4.

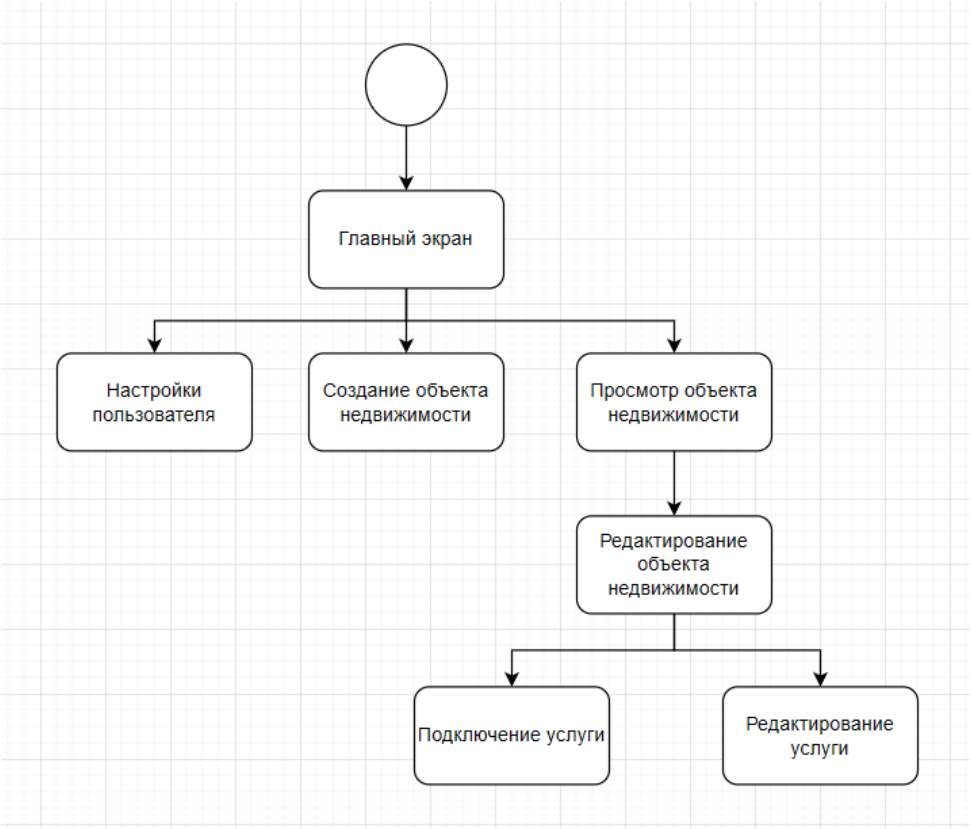


Рисунок 4 – Логическое взаимодействие компонентов

Далее построим структурную схему мобильного приложения, которая будет отображать компоненты, содержащиеся в программном продукте. Из

модулей можно выделить модуль аутентификации, включающий в себя регистрацию и авторизацию, а также модуль обработки запросов, этот модуль отвечает за передачу введённых пользователем данных в пользовательском интерфейсе в базу данных, а также отображение в интерфейсе необходимых для пользователя данных из базы данных [5]. Структурная схема мобильного приложения представлена на рисунке 5.

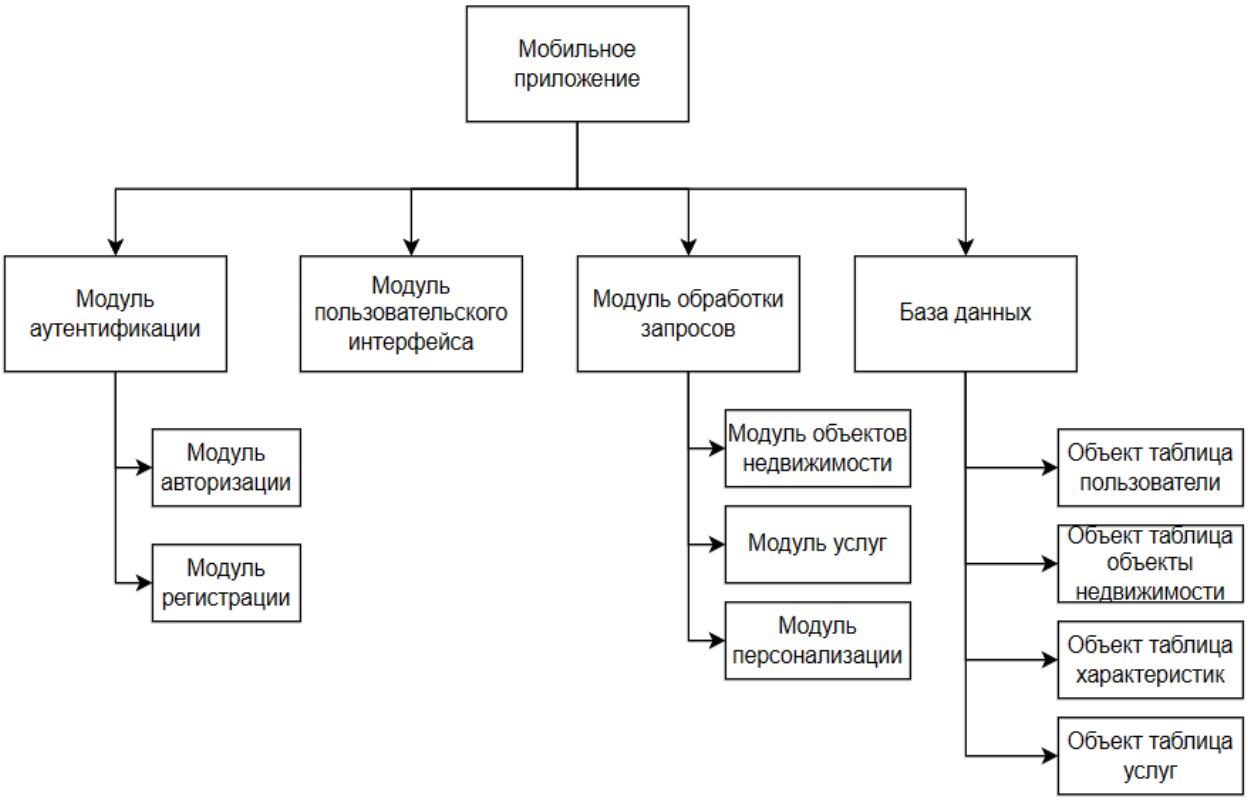


Рисунок 5 – Структурная схема мобильного приложения

Модуль аутентификации как упоминалось выше отвечает за регистрацию и авторизацию пользователя. Модуль обработки запросов содержит все необходимые компоненты для передачи информации указанной пользователем в базу данных. База данных в свою очередь будет содержать все необходимые таблицы для хранения данных о пользователях и объектах недвижимости. Подобный метод разделения информации на таблицы

позволяет при необходимости удобно передавать объекты от одного пользователя к другому без потери привязанных к ним характеристик и услуг.

Теперь декомпозируем некоторые из модулей приложения, чтобы наглядно отобразить их работу. На рисунках 6 и 7 отображены функциональные схемы модулей авторизации и регистрации.

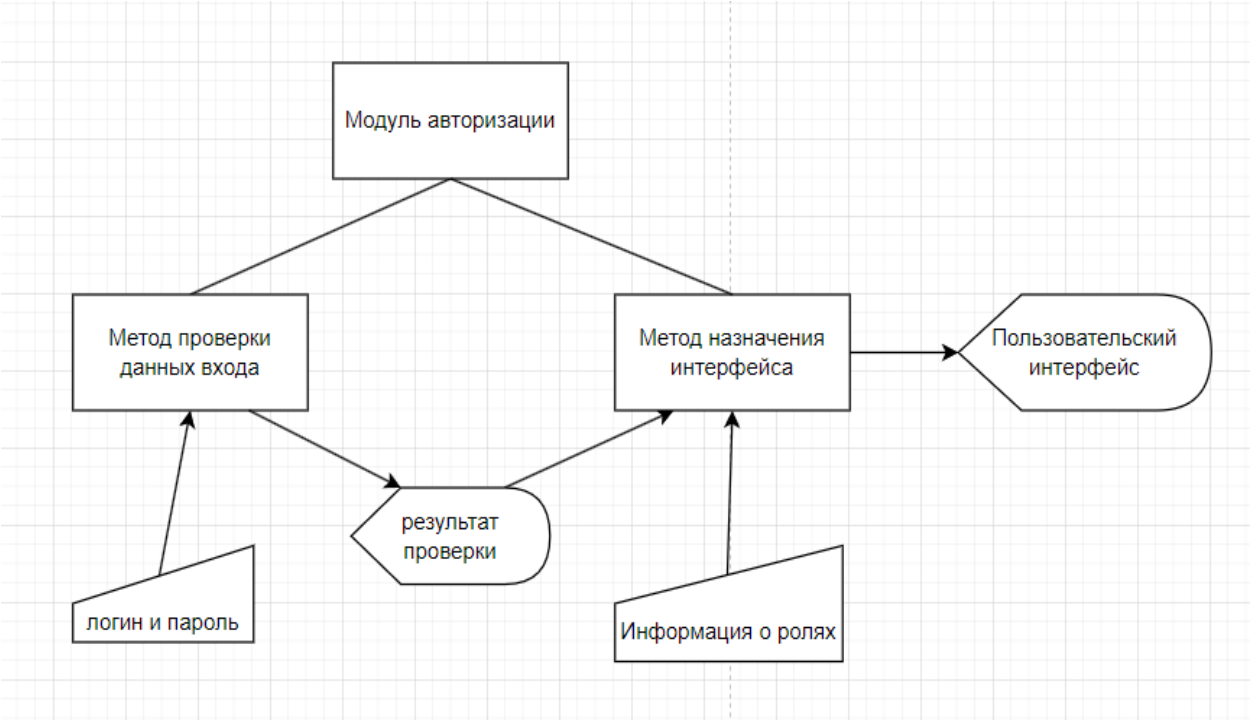


Рисунок 6 – Функциональная схема модуля авторизации

Во время авторизации пользователь вводит соответствующую информацию в поля авторизации, в данном случае это логин и пароль, после введения этих данных пользователь нажимает на кнопку войти, в этот момент модуль должен проверить эти данные и удостовериться в том, что они присутствуют в базе данных, после чего получить от неё все необходимые и связанные с пользователем данные чтобы в последствии пользователь смог продолжить работу и ними.

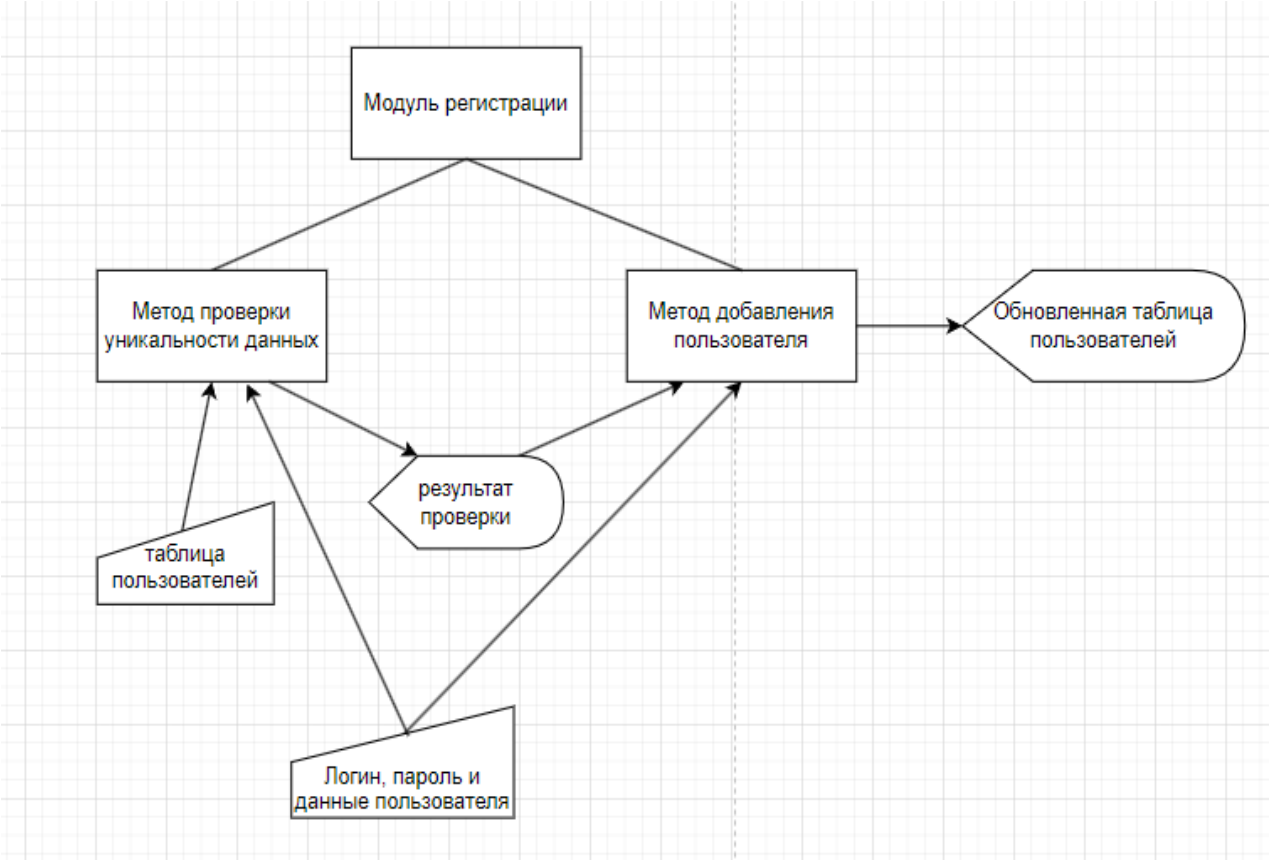


Рисунок 7 – Функциональная схема модуля регистрации

Во время регистрации пользователь вводит в соответствующую форму данные в данном случае логин и пароль с помощью которых в будущем планирует авторизоваться в данном приложении.

Получив данную информацию модуль должен проверить не используется ли каким-либо другим пользователем этот логин, чтобы не допустить возникновения коллизии, если такой логин не занят, то система регистрирует пользователя и сохраняет информацию о нем в базе данных [12].

2.4 Разработка и описание физической модели

Проектирование физической модели является завершающий этапом проектирования программного продукта. В данном разделе опишем работу клиент-серверной архитектуры, которая будет использоваться в мобильном

приложении для передачи данных на удаленный сервер, а также спроектируем базу данных, которая будет использоваться для хранения данных. Общая архитектура клиент-серверного приложения представлена на рисунке 8.

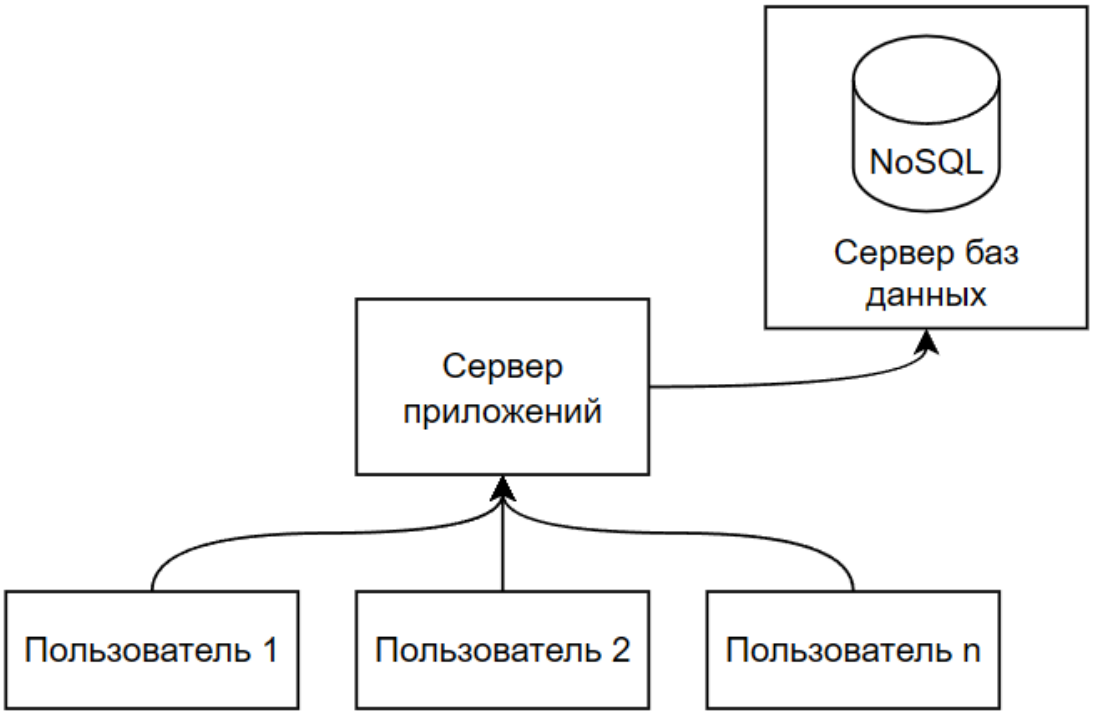


Рисунок 8 – Схема клиент серверного приложения

Подобное решение очень удобное и практичное в тех случаях, когда необходимо размещать данные пользователей в облаке. Данные хранящиеся на сервере баз данных синхронизируются в реальном времени с каждым подключенным клиентом. Обобщим и перечислим преимущества подобной архитектуры приложения:

- синхронизация данных каждый раз, когда данные изменяются;
- доступ к данным можно организовать как с мобильного устройства, так и с помощью веб-сервисов;
- масштабирование по нескольким базам данных;
- разграничение доступа пользователей к тем или иным данным;

– приложение остается отзывчивым даже в автономном режиме.

Чтобы лучше понимать процессы взаимодействия клиента и сервера в подобной архитектуре построим схему взаимодействия и передачи данных в клиент серверном приложении. Подобная схема представлена на рисунке 9.

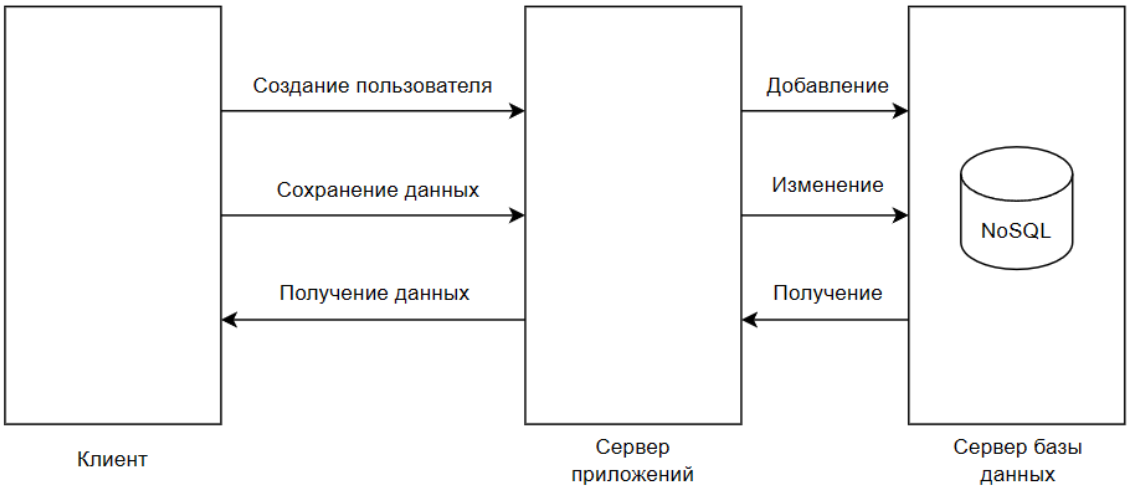


Рисунок 9 – Взаимодействие клиента с сервером

В основе взаимодействия клиент серверного приложения лежит принцип того, что такое взаимодействие начинает клиент, сервер же свою очередь лишь отвечает клиенту в зависимости от запроса [10]. В рамках приложения для расчета стоимости коммунальных услуг, клиент, он же пользователь может зарегистрироваться, авторизоваться или же внести изменения в свои объекты недвижимости в зависимости от полученного сервером приложений запроса в сервер базы данных будут внесены те или иные изменения или же будут получены соответствующие данные.

Реляционные базы данных представляют собой базы данных, которые используются для хранения и предоставления доступа к взаимосвязанным элементам информации [8]. Реляционные базы данных основаны на реляционной модели – интуитивно понятном табличном способе

представления данных. Каждая строка, содержащая в таблице такой базы данных, представляет собой запись с уникальным идентификатором. Такой подход позволит хранить данные в удобном виде, обеспечивая безопасность данных правильное распределение прав доступа пользователей к тем или иным данным, удобное добавление изменение или удаление тех или иных фрагментов данных. Пример реляционной базы данных для мобильного приложения расчета стоимости коммунальных услуг представлен на рисунке 10.

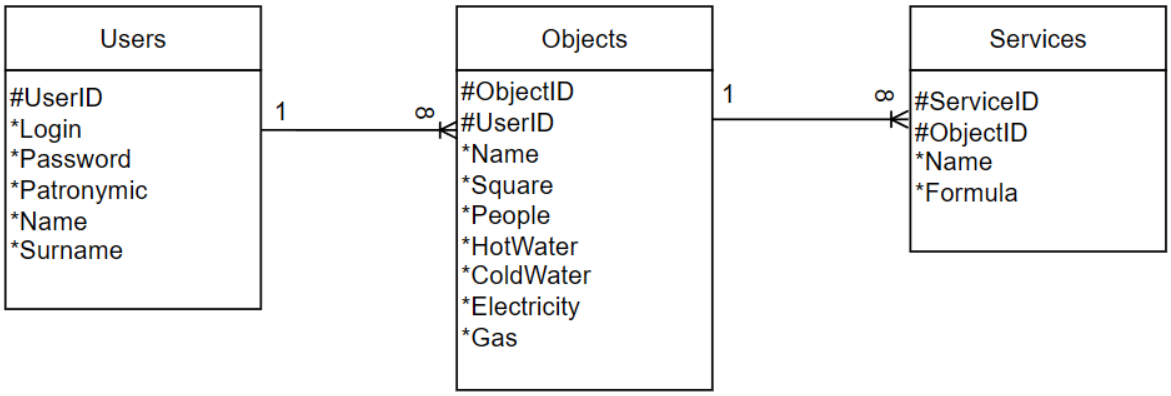


Рисунок 10 – Пример таблиц базы данных

Таблица под названием “Users” хранит в себе информацию о пользователях, которые зарегистрировались в приложении. Сервер приложений будет обновлять её в случае, когда новый пользователь зарегистрировался в системе и добавлять в неё соответствующие поля, и сверять данные в случае есть пользователь пытается авторизоваться. Таблица “Objects” содержит в себе все объекты недвижимости, которые создают пользователи, чтобы идентифицировать какие объекты относятся к определённому пользователю, необходимо установить связь один ко многим, в нашем случае один пользователь может иметь несколько объектов недвижимости, благодаря такой связи, приложение сможет установить какие

данные выводить тому или иному пользователю. Соответственно в приложении реализована возможность добавлять, изменять и удалять данные из этой таблицы, важно учитывать, что пользователь имеет право редактировать только не поля, которые было созданы им же. Таблица “Services” будет содержать в себе информацию об слугах, подключенных к объекту, поскольку приложение должно предоставлять пользователю возможность редактировать формулу расчёта разумно хранить неё целиком, чтобы пользователь в любой момент времени смог внести в неё изменения. Это таблица также определить ID объекта недвижимости, чтобы установить к какому объекту относится та или иная услуга.

Выводы по главе 2

Вторая глава ВКР посвящена проектированию мобильного приложения.

Была выполнена постановка на разработку мобильного приложения, выбрана и описана архитектура, описано логические взаимодействие компонентов приложения, разработана структурная схема мобильного приложения.

Также разработана физическая модель приложения, описана схема клиент серверного приложения, разобрано взаимодействие клиента с сервером и разработана база данных для мобильного приложения.

Глава 3 Реализация мобильного приложения

3.1 Выбор средств разработки

Описав логическую и физическую модели мобильного приложения можно переходить к непосредственной разработке приложения. На данном этапе реализации проекта важно определиться с тем, какие средства разработки будет лучше использовать для реализации приложения, чтобы итоговый продукт соответствовал всем установленным требованиям и включал в себя выше указанную структуру и логику взаимодействия компонентов. Поскольку разработка ограничивается мобильными гаджетами, в первую очередь стоит определить для каких телефонов будет разработано приложение.

В настоящий момент на рынке мобильных устройств популярны смартфоны таких компаний как: Apple, Samsung, Huawei, Xiaomi. Важно отметить что устройства от компании Apple, такие как iPhone, iPad и другие используют свою фирменную операционную систему под названием IOS. В свою очередь остальные компании производят устройства с операционной системой Android на борту. Учитывая преобладание подобных устройств у рядового пользователя было принято решение разрабатывать приложение для операционной системы Android. Android – это операционная система для мобильных устройств основанная на Linux. Согласно информации, из интернета в 86% смартфонов, проданных во всем мире в 2020 году, установлена именно эта операционная система [2][9].

Далее необходимо определить язык программирования, на котором будет написано приложение. Изучив подробнее информацию об операционной системе Android выясняется, что самый популярный язык программирования для разработки приложений используемых в этой системе – это язык программирования Java. Java – один из самых популярных и используемых языков программирования не только в мобильной разработке,

но и в других, таких как веб-разработка, frontend и backend разработки и других [3]. Не смотря на то что язык Java был разработан ещё в 1995 году, и со временем появлялись новые, более современные языки программирования, Java все ещё является вторым по популярности языком программирования в мире [1][14]. Поскольку операционная система Android позволяет работать с Java приложениями, а Java в свою очередь имеет множество возможностей и библиотек для удобной разработки было принято решение разрабатывать приложение именно на этом языке [11].

Далее необходимо определиться со средой разработки или же IDE. IDE или же интегрированная среда разработки – это комплекс средств разработки программного обеспечения, который зачастую содержит в себе все необходимые инструменты и средства для удобной и комфортной работы во время разработки программного обеспечения [16]. Среда обычно включает в себя инструменты для упрощения конструирования графического интерфейса пользователя, то есть отображение его в удобной форме, чтобы во время разработки нагляднее видеть, как будет смотреться и использоваться готовый продукт. Также подобные средства разработки включают в себя браузеры классов или же диаграммы иерархии классов. Это очень удобное решение, в случаях разработки больших проектов с применением объектно-ориентированного программирования, чтобы удобно просматривать взаимосвязи и наследственные связи между классами, реализованными в проекте [13].

В качестве IDE для разработки приложения для расчета стоимости коммунальных услуг было выбрано средство разработки Android Studio. Android Studio – это среда разработки специализирующаяся именно на разработке мобильных приложений под операционную систему Android на языке Java [17]. Программа обладает всеми перечисленными выше инструментами, которые можно использовать во время разработки и не только. На время жизненного цикла программы разработчики множество раз обновляли её и добавляли в неё новые средства, облегающие разработку

приложений для программистов. В программе предусмотрены многочисленные шаблоны для общих элементов программирования для операционной системы Android, а также есть удобные помощники и справочные материалы. В окне программы демонстрируется структура создаваемого проекта, благодаря чему процесс написания программы является более удобным и гибким. К тому же в программе все вносимые изменения отображаются в режиме реального времени, поэтому можно сразу же оценивать их полезность и целесообразность [20]. Многофункциональная среда разработки Android Studio обладает полным набором всех необходимых инструментов и функций, благодаря которым даже такое трудоемкое и сложное дело, как создание приложения для Android становится простым и интересным. При этом программа работает надежно и стабильно, проста в освоении и отлично подходит как для профессиональных разработчиков, так и для новичков в программировании приложений для ОС Android [18].

Разработка Android приложения имеет не только рекламный характер, но и упрощает внедрение новых сервисов и услуг для клиентов.

3.2 Разработка пользовательского интерфейса

Определившись со средствами разработки приложения перейдем к непосредственному проектированию приложения, или же разработке пользовательского интерфейса. Перед реализацией непосредственной логики работы приложения, для начала нужно определить, как оно будет выглядеть и как пользователь будет с ним взаимодействовать.

В разделе 2.2 работы под названием выбор архитектуры мобильного приложения, уже были описаны основные элементы, содержащиеся в мобильном приложении. Поскольку приложения на Android в своей работе используют окна, называемые Activity, будет разумным построить навигационную карту приложения, в которой будут описаны все используемые в приложении Activity, на этой навигационной карте будут

отображены все кнопки и другие элементы пользовательского интерфейса для каждого из окон, а также переходы между этими окнами. Это позволит наглядно отобразить общий вид приложения, что облегчит процесс непосредственной реализации мобильного приложения в Android Studio [19].

Для создания навигационной карты мобильного приложения было решено использовать онлайн сервис под названием Pixso. Pixso – это бесплатный веб-сервис предоставляющий удобные инструменты для проектирования интерфейса пользователя. Сервис также поддерживает формат многопользовательской совместной работы, обладает облачной синхронизацией файлов в реальном времени, с функциями рисования и разметки. Результат разработки навигационной карты пользовательского интерфейса приложения показан на рисунке 11.

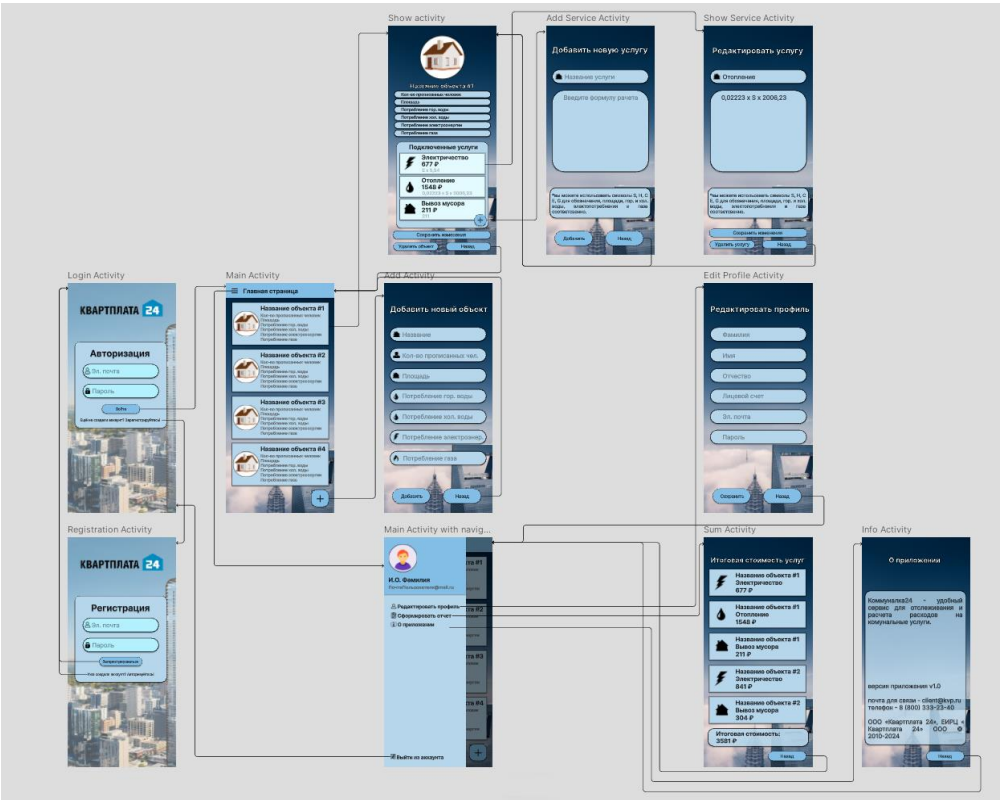


Рисунок 11 – Навигационная карта приложения для расчета стоимости коммунальных услуг

На рисунке представлена навигационная карта приложения, которая содержит в себе все экраны или же Activity, которые будут содержаться в приложении. Каждое из окон выполняет ту или иную задачу, например, создание нового объекта недвижимости или же просмотр информации об уже созданном объекте. В совокупности весь этот интерфейс позволит решить поставленную перед приложением задачу, пользователь сможет работать с объектами и услугами в отдельных окнах. Рассмотрим некоторые ключевые окна, и опишем их функционал. На рисунке 12 представлен интерфейс, отвечающий за авторизацию пользователя.

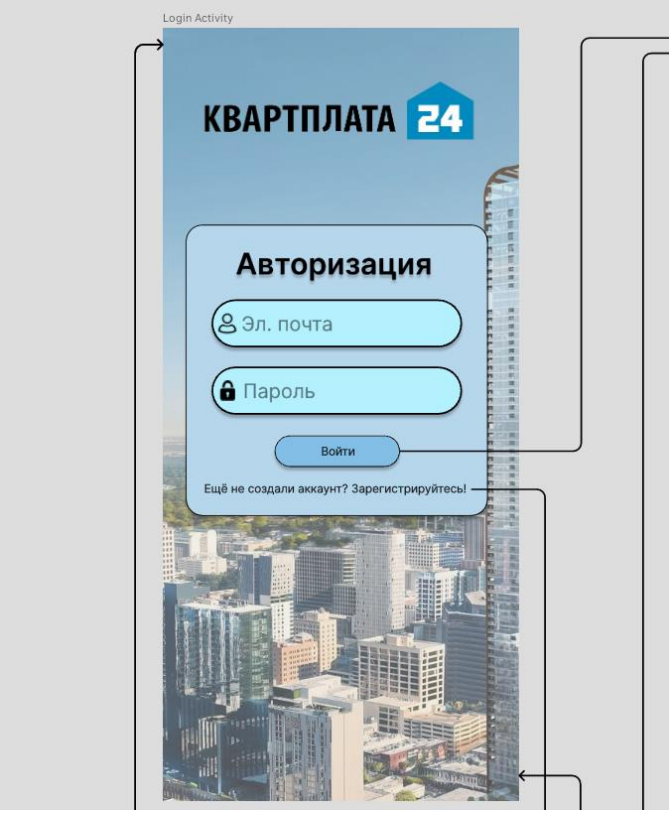


Рисунок 12 – Экран авторизации пользователя

Данная форма предоставляет пользователю возможность авторизоваться чтобы в последствии пользоваться функционалом приложения. Экранная форма регистрации выглядит аналогично, она состоит из ввода пароля и логина, но её логика отличается от формы авторизации, в ней пользователь

создает свой аккаунт для последующей авторизации. Далее перейдем к разбору Main Activity, которая представлена на рисунке 13.

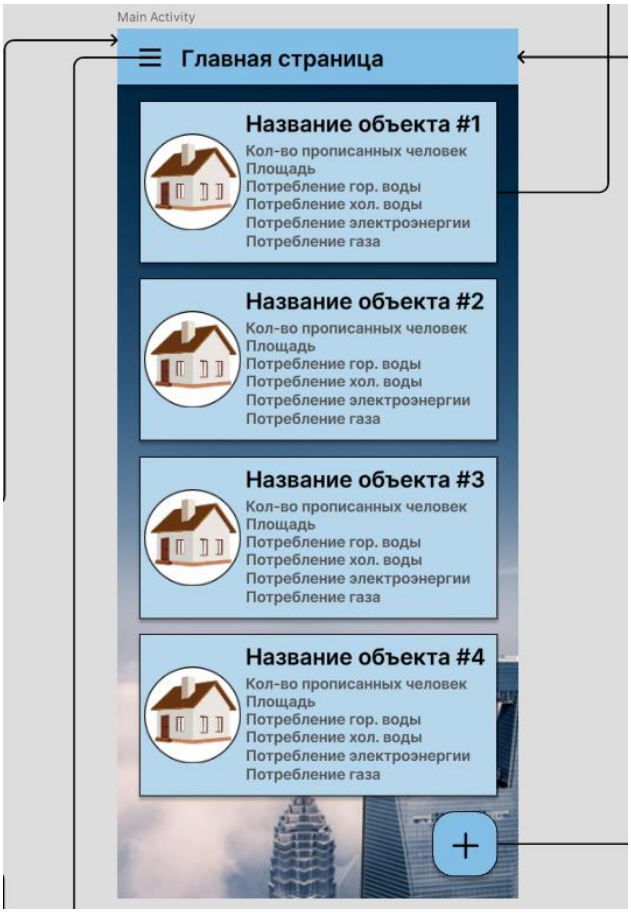


Рисунок 13 – Главная страница приложения

Пройдя процедуру авторизации, пользователь попадает на главную страницу, на ней отражены все объекты недвижимости, созданные и настроенные пользователем, а также есть кнопка создания нового объекта в правом нижнем углу, нажав на неё пользователю откроется окно создания нового объекта недвижимости. Также пользователь может вызвать боковую панель нажав на кнопку в левом верхнем углу, боковая панель или же navigation drawer представлена на рисунке 14.

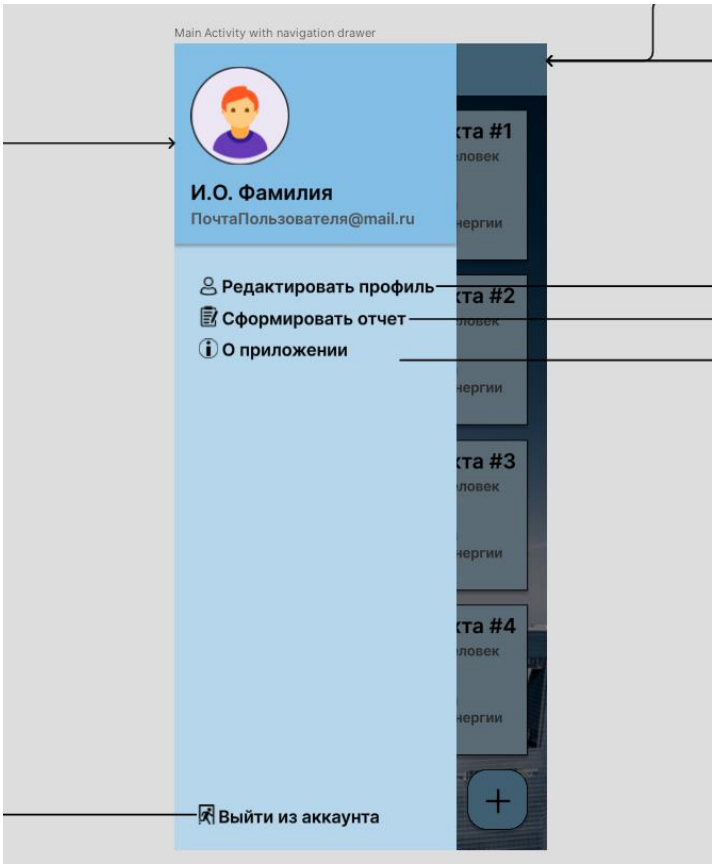


Рисунок 14 – Боковая панель приложения

На боковой панели отображена основная информация о пользователе, а также имеется несколько кнопок для навигации в приложении. Кнопка выход из аккаунта соответственно возвратит пользователя на экранную форму авторизации, это очень удобно в случае если несколько разных людей будут пользоваться приложением. Нажав на кнопку о приложении пользователю отобразиться экранная форма с основной информации о приложении, а также контакты компании в случае если пользователю по той или иной причине нужно будет связаться с сотрудниками службы поддержки компании. Кнопка редактировать профиль перенесет пользователя в Activity в которой он сможет указать или же отредактировать информацию о себе. Кнопка сформировать отчет перенесет пользователя в экранную форму содержащую информацию о всех подключенных услугах к объектам недвижимости, а также о суммарных

расходах на все услуги. Экран итоговой стоимости услуг представлен на рисунке 15.

Рисунок 15 – Экран итоговой стоимости всех подключенных услуг

На данной форме в виде списка отображаются все подключенные коммунальные слуги, указано название объекта недвижимости, к которому подключена услуга, а также название самой услуги и её стоимость, в самом низу суммируется стоимость всех подключенных услуг и отображается отдельно.

Далее рассмотрим форму обзора объекта недвижимости, в которою будет попадать пользователь, нажав на один из объектов недвижимости на главном экране, экранная форма просмотра объекта недвижимости представлена на рисунке 16.

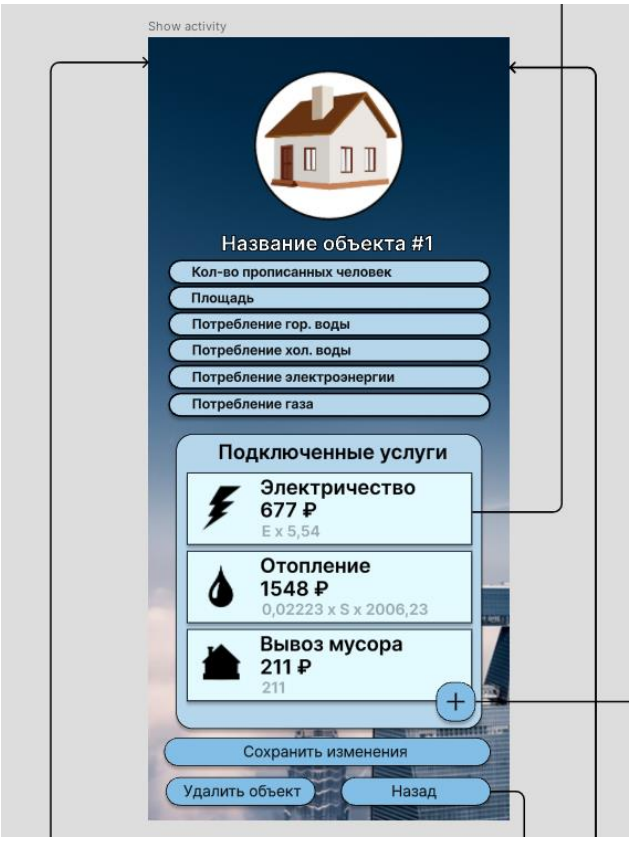


Рисунок 16 – Экранная форма объекта недвижимости

Попав в данную Activity, пользователь может просмотреть подробную информацию о конкретном объекте недвижимости, выбранным пользователем на главном экране приложения. В этом окне пользователь может посмотреть информацию об объекте недвижимости и подключенных к нему услугах. Пользователь может указать новые данные и сохранить изменения, или же вовсе удалить созданный им объект недвижимости. Также нажав на кнопку со знаком плюса, пользователь попадет в окно подключения услуги объекта недвижимости, в случае если пользователь нажмет на одну из подключенных услуг, пользователь попадет в экранную форму редактирования услуги, данная форма представлена на рисунке 17.

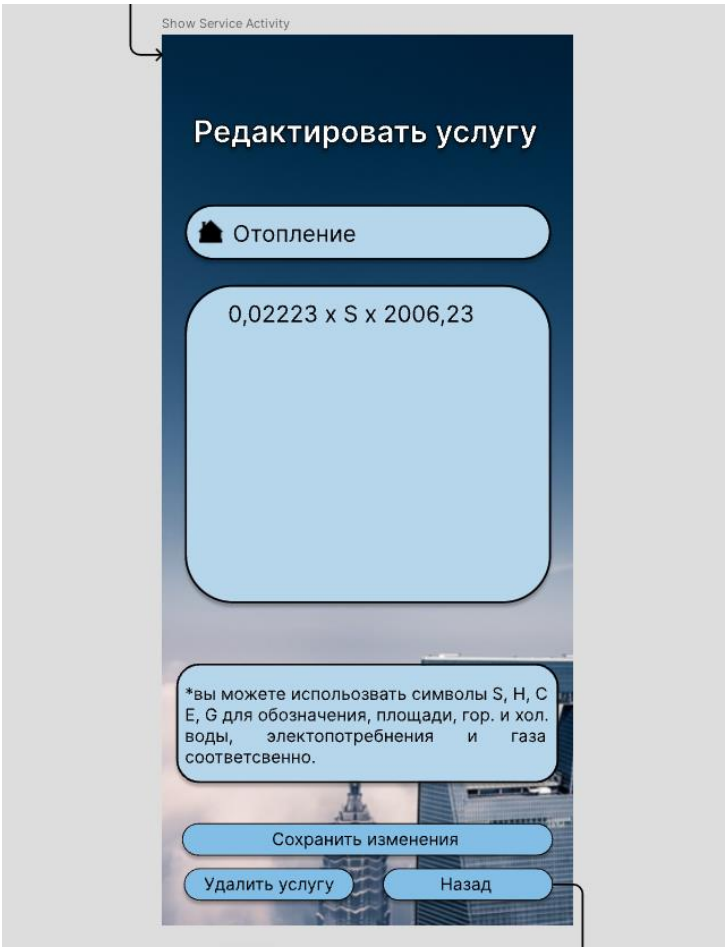


Рисунок 17 – Экранная форма редактирования услуги

На рисунке представлена экранная форма редактирования услуги, форма создания услуги выглядит аналогично за исключением отличающихся в нижней части формы кнопок, кнопки удалить и сохранить изменения отсутствуют, вместо них кнопка создать.

Чтобы создать услугу пользователю необходимо указать название услуги, а также формулу с помощью, которой будет рассчитываться стоимость в зависимости от характеристик объекта, которые пользователь указывал ранее.

Данных подход позволит очень гибко настраивать услуги, пользователь сможет указать, как и услуги с фиксированной платой, не зависимой ни от каких параметров, так и сложные формулы, в которых для расчета стоимости используются характеристики объекта недвижимости.

3.3 Реализация мобильного приложения

После определения внешнего вида мобильного приложения и его функционала можно перейти к непосредственной реализации приложения, создав проект в Android Studio и подключив все необходимые библиотеки в конфигурационном файле перейдем к поочередному созданию экранных форм. Для начала необходимо создать xml файлы каждой из Activity, которые мы описали выше. В этих файлах описывается UI составляющая, описываются какие элементы интерфейса будут присутствовать на данный конкретной экранной форме, это могут быть как картинки, так и поля для ввода текста, для всех этих элементов необходимо указать размеры, паддинг, выравнивание и так далее, также обязательно необходимо указать ID данного конкретного элемента. Эти ID в последствии будут использованы в java классах, и будут связывать эти объекты пользовательского интерфейса с описанными в коде объектами данного типа класса, чтобы конкретно указывать куда будет передаваться информация из поля для ввода текста, что будет делать та или иная кнопа, и откуда будет браться информация для отображения её в поле текса или же какую именно картинку необходимо отобразить в данном конкретном месте. Реализацию java классов опишем ниже, а пока что разберем xml файлы.

Необходимо создать xml файлы, для экранов авторизации, регистрации, главного экрана, экрана создания услуги и объекта, экрана просмотра услуги и объекта, а также экранов, содержащих информацию о пользователе и приложении.

Опишем основные экранные формы, содержание в себе наибольшее количество элементов, например, форма авторизации пользователя.

Начнем с создания экранных форм авторизации и регистрации пользователя, на рисунке 18 представлена часть кода, отвечающая за формирования авторизации пользователя.

Рисунок 18 – часть xml файла activity_login.xml

Экранная форма должна содержать кнопку авторизации, которая будет переносить пользователя на главный экран в случае успешной авторизации, выше кнопки расположены два элемента позволяющие пользователю ввести логин и пароль, также под кнопкой расположен текст, информирующий пользователя о том, что он может зарегистрировать аккаунт в случае если он этого ещё не сделал, нажав на этот текст пользователь попадет в Activity, отвечающую за регистрацию нового аккаунта. Её интерфейс схож с авторизацией, есть также 2 поля для ввода логина и пароля, кнопка зарегистрироваться и текст, сообщающий пользователю о том, что он может авторизоваться если у него уже есть аккаунт.

Далее опишем xml файл для главной активности на которой будут отображаться все созданные пользователем объекты недвижимости.

Отображать данные будем с помощью recycler view в виде списка, для этого предварительно опишем как будет выглядеть элемент в списке.

На рисунке 19 представлена часть кода, описывающая карточку для отображения объекта недвижимости.

Рисунок 19 – часть xml файла user_item.xml

На карточке располагаются изображение и несколько текстовых полей, в которые будут передаваться данные из базы данных. Также для главной активности добавим боковую панель, определим в ней соответствующие кнопки, и добавим отдельную кнопку, располагающуюся слева снизу экрана.

Далее опишем экраны создания нового объекта недвижимости и редактирования профиля пользователя, они структурно похожи между собой.

Обе активности содержат кнопку назад, чтобы вернуться на главный экран и кнопки создать и сохранить чтобы перенести данные в базу данных. Помимо кнопок в этих активностях также есть несколько полей для ввода текста, информация которую укажет пользователь будет передаваться в базу данных.

Нажав на любой из созданных пользователем объектов на главном экране, пользователь попадет в экранную форму просмотра и редактирования информации об это объекте. Добавим изображение, текстовые поля в которых пользователь сможет отредактировать информацию, кнопки для внесения изменений и удаления объекта недвижимости, а также отдельный список, с кнопкой создания, в котором будут отображаться все подключенные услуги. Для этого списка также создадим карточку, которая будет использоваться как элемент в списке, она будет содержаться в себе название, формулу расчета и итоговую стоимость услуги.

Опишем экранные формы для создания, редактирования и удаления услуг, они схожи между собой и содержат два поля для текстового ввода и кнопки возврата на главный экран, создания, сохранения изменений или удаления данных. Итоговая стоимость будет рассчитываться автоматически по формуле, которую укажет пользователь.

После создания xml файлов и интерфейса приложения можно переходить к backend разработке. В java классах описывается логическая составляющая той или иной экранной формы, мы описываем что произойдет при нажатии на кнопку, или куда будет передаваться информация из поля для ввода текста.

Начнем с описания работы экранов авторизации и регистрации, часть кода для авторизации, отвечающая за нажатие на кнопку авторизоваться представлена на рисунке 20.

```
loginButton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        String email = loginEmail.getText().toString();
        String pass = loginPassword.getText().toString();

        if (!email.isEmpty() && Patterns.EMAIL_ADDRESS.matcher(email).matches()) {
            if (!pass.isEmpty()){
                auth.signInWithEmailAndPassword(email,pass)
                        .addOnSuccessListener(new OnSuccessListener<AuthResult>() {
                            @Override
                            public void onSuccess(AuthResult authResult) {
                                userName = getUserNameWithoutExtension(email);
                                Toast.makeText( context: LoginActivity.this,  text: "Login Successful", Toast.LENGTH_SHORT).show();
                                startActivity(new Intent(LoginActivity.this, MainActivity.class));
                                finish();
                            }
                        }).addOnFailureListener(new OnFailureListener() {
                            @Override
                            public void onFailure(@NonNull Exception e) {
                                Toast.makeText( context: LoginActivity.this,  text: "Login Failed", Toast.LENGTH_SHORT).show();
                            }
                        });
            } else {
                loginPassword.setError("Password cannot be empty");
            }
        } else if (email.isEmpty()){
            loginEmail.setError("Email cannot be empty");
        } else {
            loginEmail.setError("Please enter valid email");
        }
    }
});
```

Рисунок 20 – код реализующий нажатие на кнопку авторизации

В активности авторизации определим откуда считывается почта для авторизации и откуда пароль, далее опишем логику работы кнопки, в первую очередь проверим чтобы оба поля для ввода текста были не пустые, далее проверка на то, что указанная пользователем действительно имеется в базе данных, проверим что указанный пароль соответствует сохраненному в базе данных, после чего локально сохраняем информацию о пользователе, чтобы при следующем запуске приложения ему не нужно было заново вводить логин и пароль, также передадим информацию о логине в главную активность, чтобы в последствии использовать эту информацию для отображения данных.

Блок схема работы авторизации представлена на рисунке 21.

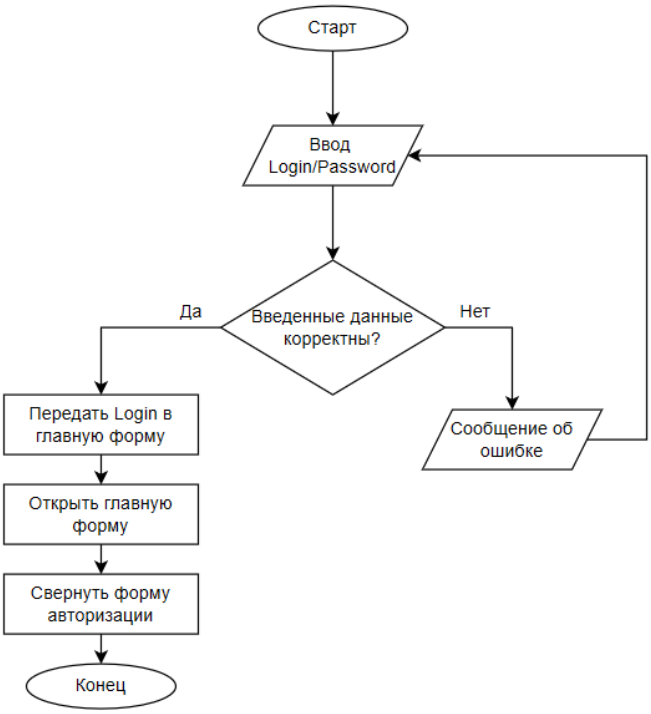
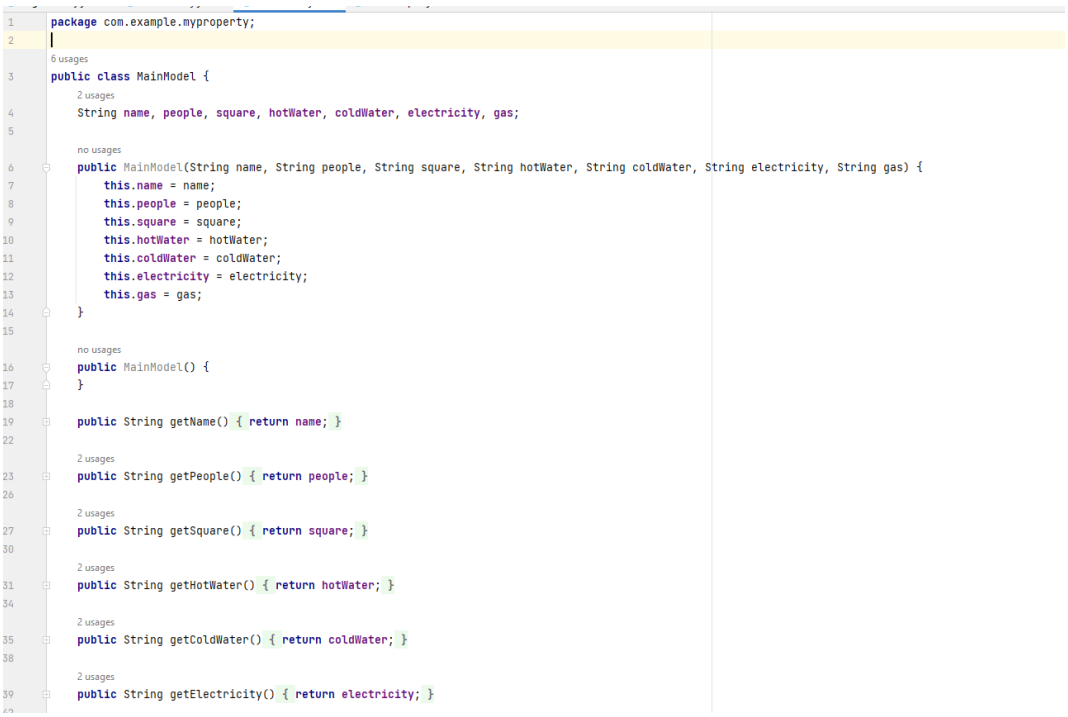


Рисунок 21 – Блок схема процесса авторизации

Активность регистрации работает похожим образом, также определяем где пользователь вводит данные логина и пароля, описываем логику для текста под кнопкой, который будет переносить нас на экран авторизации. Единственной что отличается это логика кнопки регистрации, при регистрации мы проверяем нет ли такого логина в базе данных, если его нет, то мы передаем данные в базу данных, в случае если такой пользователь уже зарегистрирован выводим на экран пользователю соответствующее уведомление.

Далее опишем главную Activity, добавим боковую панель определим куда пользователя будет переносить при нажатии на ту или иную кнопку, и перейдем к реализации отображения списка. Для того чтобы отобразить список всех объектов пользователя необходимо создать адаптер, а также модель данных, модель содержит в себе все поля из базы данных, модель данных для объекта недвижимости представлена на рисунке 22.



```
package com.example.myproperty;

6 usages
public class MainModel {
    2 usages
    String name, people, square, hotWater, coldWater, electricity, gas;

    no usages
    public MainModel(String name, String people, String square, String hotWater, String coldWater, String electricity, String gas) {
        this.name = name;
        this.people = people;
        this.square = square;
        this.hotWater = hotWater;
        this.coldWater = coldWater;
        this.electricity = electricity;
        this.gas = gas;
    }

    no usages
    public MainModel() {
    }

    public String getName() { return name; }

    2 usages
    public String getPeople() { return people; }

    2 usages
    public String getSquare() { return square; }

    2 usages
    public String getHotWater() { return hotWater; }

    2 usages
    public String getColdWater() { return coldWater; }

    2 usages
    public String getElectricity() { return electricity; }
```

Рисунок 22 – Модель объекта недвижимости

В модели необходимо создать два конструктора, один пустой, другой содержит в себе все параметры, которые были определены в базе данных заблаговременно. Также создадим геттеры, которые в последствии понадобятся для отображения данных. После описания адаптера и модели определяем их в главной активности и используем для отображения данных пользователя, важно, чтобы на главном экране отображались только данные конкретного пользователя которой авторизован в приложении, поэтому на предыдущем этапе мы передавали информацию о пользователе в главную форму.

Форма отображения будет работать аналогичным образом, также есть список услуг, которые привязаны именно к этому объекту, единственное отличие, это возможность удаления информации об объекте из базы данных или внесение изменений, для этого описываем логику работы кнопок и текстовых полей.

Разберем добавление новой информации в базу данных на примере экранной формы для создания нового объекта недвижимости, часть кода реализующего экранную форму представлена на рисунке 23.

```
1 usage
private void dataInsert() {


    Map<String, Object> map = new HashMap<>();
    map.put("name", edName.getText().toString());
    map.put("people", edPeople.getText().toString());
    map.put("square", edSquare.getText().toString());
    map.put("hotWater", edHotWater.getText().toString());
    map.put("coldWater", edColdWater.getText().toString());
    map.put("electricity", edElectricity.getText().toString());
    map.put("gas", edGas.getText().toString());


    FirebaseDatabase.getInstance().getReference().child(LoginActivity.getUserName()).push().setValue(map).addOnSuccessListener(new OnSuccessListener<Void>() {
        @Override
        public void onSuccess(Void unused) {

            Toast.makeText(getApplicationContext(), text: "data  insert complete", Toast.LENGTH_SHORT).show();


        }
    }).addOnFailureListener(new OnFailureListener() {
        @Override
        public void onFailure(@NonNull Exception e) {

            Toast.makeText(getApplicationContext(), text: "data insert failed", Toast.LENGTH_SHORT).show();

        }
    });
}

1 usage
private void dataD() {
    edName.setText("");
    edElectricity.setText("");
    edGas.setText("");
```

Рисунок 23 – добавление данных в базу данных

Экранная форма содержит несколько полей для ввода текста, в эти поля пользователь вводит информацию об объекте, важно проверить чтобы все поля были заполнены какими-то данными, чтобы передача данных прошла исправно, после проверки, переносим данные в соответствующие поля базы данных, не забыв указать ключ пользователя и уникальный идентификатор данного объекта недвижимости. Блок схема добавления данных в базу данных представлена на рисунке 24.

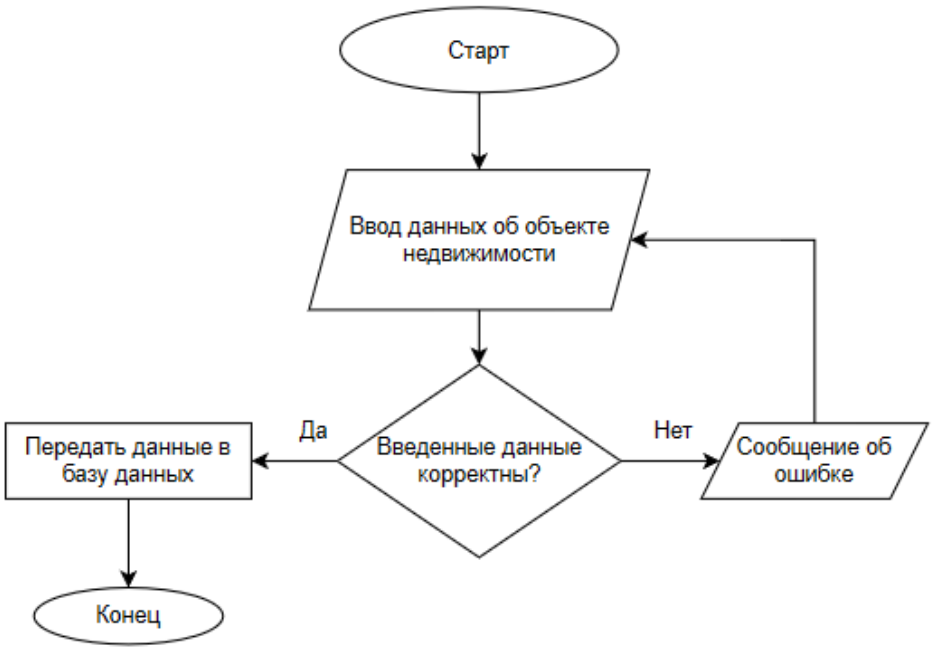


Рисунок 24 – Добавление данных в базу данных

Похожим образом работает изменение данных в активности для просмотра данных об объекте недвижимости. Важно чтобы при внесении изменений в базу данных, были изменены именно те данные которые планировал изменить пользователь, поэтому при переходе на экран объекта, мы передаем уникальный идентификатор этого объекта.

Внесение изменений в профиле пользователя работает также, единственное отличие - это изменение полей в таблице пользователей за место таблицы с объектами, пользователь может изменить имя, фамилию, отчество и так далее.

Для создания и изменения подключенных к объекту недвижимости услуг, также опишем логику работы активностей, работает все аналогично, единственное отличие, это количество текстовых полей, их всего два, и взаимосвязь между услугами и объектами для правильного отображения. В итоге все эти активности формируют полноценное приложение, удовлетворяющее всем требованиям.

3.4 Тестирование мобильного приложения

Перейдем к непосредственному тестированию созданного приложения, для этого скомпилируем установочный файл и загрузим приложение на телефон, также помимо телефона тестирование проводилось на эмуляторе Android встроенном в среду разработки. Первым делом необходимо проверить работоспособность системы авторизации и регистрации, в случае нажатия на кнопку без заполненных полей или заполненным только одним полем выводится Toast сообщающий пользователю о том, что данные введены не корректно, если такой логин уже занят выведется соответствующее сообщение, тоже самое для регистрации.

Далее зарегистрируем два разных аккаунта и создадим для них объекты недвижимости, при смене аккаунта объекты меняются что говорит о том, что отображение и создание данных работает исправно, в целом приложение не вылетает и работает исправно, но по ходу эксплуатации было выявлено несколько недостатков, которые можно будет исправить в будущих обновления приложения:

- смена пароля и логина происходит без какой-либо верификации, это может быть не безопасно в случае если злоумышленник получит доступ к аккаунту, предлагается добавить отдельные экраны для смены пароля и почты с подключение функции двойной аутентификации и отправной проверочного письма на почту;
- при вводе формулы для расчета услуги пользователю может быть не очень комфортно вводить все формулы с нуля, предлагается добавить всплывающее окно-подсказку в которой будут прописаны некоторые распространенные формулы расчетов, чтобы пользователь мог выбрать шаблонный вариант и не вписывать все вручную;

– на итоговом экране отображаются все услуги и их итоговая стоимость, можно расширить функциональность активности и добавить окно выбора объекта чтобы посчитать сумму расходов не только по всем объектам сразу, но и по какому-то одному конкретному.

Подводя итоги можно сделать вывод, что приложение функционирует без сбоев, и соответствует обозначенным требованием, также приложение имеет потенциал в расширении функционала и интерпретации в различные системы.

Выводы по главе 3

В данной главе был произведен выбор средств и инструментов разработки и обоснование из применения. Была построена навигационная карта приложения и разработан интерфейс, также было разработан непосредственный прототип мобильного приложения, после чего протестирован.

По итогам тестирования было выявлено что приложение соответствует обозначенным требования, а также определены возможные расширения приложения.

Заключение

Бакалаврская работа посвящена разработке мобильного приложения для расчета стоимости коммунальных услуг.

Во время выполнения выпускной квалификационной работы был произведен анализ деятельности компании ООО «Квартплата 24», а также экосистемы сервисов компании. В работе были исследованы аналоги приложения, которое было необходимо разработать, установлены требования для мобильного приложения и поставлена задача на разработку.

После постановки задачи на разработку были построены и обоснованы логические и физические модели мобильного приложения, была описана основная навигационная структура, структура базы данных и физическая схема компонентов серверного приложения.

В ходе работы были определены средства разработки мобильного приложения, с помощью которых был реализован интерфейс и логическая составляющая приложения.

Результатом выпускной квалификационной работы является рабочее мобильное приложение для расчета стоимостей коммунальных услуг, функционал которого включает:

- возможность регистрации, авторизации, и выхода из аккаунта;
- возможность ввода и обработки показаний для расчёта стоимостей различных услуг;
- занесение данных о несколько объектах недвижимости, удаление и редактирование данных;
- возможность составления отчета по всем услугам для всех объектов недвижимости.

Описанный выше функционал постность соответствует обозначенным в работе требованиям, а проведенное тестирование показало, что приложение работает исправно и имеет потенциал в последующем обновлении, дополнении и интерполирования в различные сервисы.

Список используемой литературы

1. Блоха, И. О. Основы программирования на языке Java: учебное пособие. М.: Высшая ш,кола, 2018. 352 с.
2. Глинский, А. П. Разработка мобильных приложений на Android: учебное пособие. СПб.: Питер, 2019. 384 с.
3. Дейтел, П., Дейтел, Х. Программирование на языке Java: учебный курс. 10-е изд. М.: Вильямс, 2017. 1376 с.
4. Жирков, В. В. Базы данных: теория и практика. СПб.: БХВ-Петербург, 2018. 576 с.
5. Иванов, С. И., Петров, А. В. Системы управления базами данных: учебное пособие. М.: МИФИ, 2017. 424 с.
6. Климов, Д. В. Безопасность мобильных приложений: учебное пособие. СПб.: Питер, 2020. 320 с.
7. Кочуров, А. И. Введение в теорию информационных систем: учебное пособие. М.: Юрайт, 2018. 416 с.
8. Лебедев, А. И. Введение в информационные системы и базы данных. М.: Альфа, 2017. 368 с.
9. Литвин, М. В. Программирование мобильных приложений на платформе Android. М.: ДМК Пресс, 2019. 464 с.
10. Мельников, И. И., Семёнов, П. А. Основы сетевых технологий: учебное пособие. М.: Эксмо, 2018. 352 с.
11. Назаров, С. А. Программирование на языке Java: лабораторный практикум. СПб.: Питер, 2019. 432 с.
12. Панкратов, В. А. Информационные системы: теория и практика. М.: Наука, 2017. 488 с.
13. Пермяков, Д. П., Логинов, И. В. Технологии разработки программного обеспечения: учебное пособие. СПб.: Питер, 2020. 512 с.
14. Смирнов, В. В., Тихонов, Ю. Н. Объектно-ориентированное программирование на языке Java. М.: КУДИЦ-ОБРАЗ, 2018. 320 с.

15. Солнцев, Е. И., Федорова, Л. В. Разработка клиент-серверных приложений на Java. М.: ДМК Пресс, 2017. 408 с.

16. Blanchard, S. Java Programming: Mastering Advanced Features. New York: Apress, 2019. 452 p.

17. Brown, C. Mobile Application Development with Android: The Complete Guide. London: Packt Publishing, 2018. 500 p.

18. Davis, M. Beginning Android Programming with Java. 3rd ed. Hoboken: Wiley, 2017. 704 p.

19. Fitzgerald, J. Building Mobile Applications with Java: Comprehensive Guide. San Francisco: No Starch Press, 2020. 480 p.

20. Lee, K. J. Database Systems: Design, Implementation, & Management. 13th ed. Boston: Cengage Learning, 2019. 784 p.