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

Кафедра «Прикладная математика и информатика»
(наименование)

09.03.03 Прикладная информатика
(код и наименование направления подготовки, специальности)

Цифровая трансформация бизнеса
(направленность (профиль) / специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему Проект создания информационной системы для привлечения клиентов

Обучающийся Л.И. Громов _____
(Инициалы Фамилия) (личная подпись)

Руководитель Н.Н. Рогова
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Консультант канд. филол. наук, доцент М.В. Дайнеко
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Тольятти 2025

Аннотация

Темой данной выпускной квалификационной работы является «Проект создания информационной системы для привлечения клиентов».

Выпускная квалификационная работа посвящена разработке информационной системы, направленной на привлечение клиентов посредством современных цифровых технологий [12]. В ходе исследования рассматриваются методы автоматизации процессов взаимодействия пользователей с сервисами компании, а также внедрение функционала, стимулирующего лояльность клиентов, включая систему кешбэка.

Бакалаврская работа предусматривает создание мобильного приложения, обеспечивающего удобный и эффективный доступ к услугам, что способствует повышению удовлетворённости пользователей и расширению клиентской базы. Работа включает анализ требований, проектирование архитектуры системы, а также описание этапов её реализации и тестирования. Результаты исследования могут быть использованы для оптимизации маркетинговых стратегий и улучшения качества обслуживания в различных сферах бизнеса.

Работа содержит 44 страницы текста, 16 рисуноков, 5 таблиц и 22 использованных источников.

Abstract

The title of the graduation work is Development Project of an Information System for Customer Acquisition.

The graduation work consists of an introduction, several parts, 16 figures, 5 tables, a conclusion, and a list of 22 references including 12 foreign sources.

The aim of this graduation work is to develop a mobile application that increases customer engagement through the implementation of cashback features for utility bill payments.

The object of the graduation work is the process of interaction between users and their utility services.

The subject of the graduation work is the use of information technologies to enhance the efficiency and attractiveness of this interaction.

The key issue of the graduation work is to automate and incentivize utility payments using a mobile application.

The graduation work may be divided into several logically connected parts which are: analysis of existing solutions based on the "AS IS" model, development of the "TO BE" context model to identify necessary improvements, logical design of the mobile application including use case diagrams, and selection of technologies for development.

The first part describes in detail the analysis of existing systems and the rationale for developing a new solution. We examine current mobile applications in the domain and assess their limitations.

The second part outlines the design and modeling phase, including system architecture, UML diagrams, and functional specifications.

The third part concentrates on the implementation of the application, covering both front-end and back-end development, as well as testing and evaluation of features.

In conclusion we’d like to stress that the proposed mobile application enhances user engagement by offering additional functionality and incentives. The work is of interest for a wide circle of readers interested in software development, mobile applications, and utility service automation.

Оглавление

Введение 5
Глава 1 Анализ предметной области 6
1.1 Характеристика компании «ООО Квартплата 24» 6
1.2 Концептуальное моделирование предметной области 8
1.3 Анализ существующих аналогов 9
1.4 Разработка требований к информационной системе 11
1.5 Разработка и анализ бизнес-модели «Как должно быть» 13
Глава 2 Проектирование мобильного приложения 16
2.1Логическое моделирование мобильного приложения 16
2.1.1 Диаграмма вариантов использования 16
2.1.2 Диаграмма классов мобильного приложения 18
2.2 Разработка логической и физической модели данных 19
2.3 Разработка диаграммы компонентов 22
Глава 3 Разработка мобильного приложения 25
3.1 Выбор технологий и программных средств разработки 25
3.2 Описание функциональности и реализации проекта 27
3.2.1 Пользовательский интерфейс и функционал приложения 27
3.2.2 Реализация backend части мобильного приложения 33
3.3 Тестирование мобильного-приложения 37
3.4 Расчет экономической эффективности 39
Заключение 42
Список используемой литературы и используемых источников 43
Приложение А Зависимости в файле build.gradle.kts (Module App) 45
Приложение Б Код файла TransactionApi.kt 46
Приложение В Код файла RetrofitClient.kt 47
Приложение Г Код файла CashbackRules.kt 48
Приложение Д Метод CashbackCalculator.kt 49
Приложение Е Код файла TransactionRoutes.kt 50

Введение

В текущих реалиях цифровизации и постоянного внедрения информационных технологий в современное общество, ключевыми решениями бизнеса для привлечения клиентов являются системы повышения лояльности и доверия клиентов, что способствует расширению аудитории компании. Одним из перспективных направлений таких решений является интеграция системы лояльности в компанию, которая стимулирует пользователей и клиентов к регулярному взаимодействию с платформой или компанией.

Целью выпускной квалификационной работы является разработка такого решения, которое имело бы в своем функционале такую систему лояльности, путем обеспечения удобной оплаты ЖКУ с функционалом кешбека, что позволит повысить привлекательность и уникальность компании на рынке услуг, и выделить ее на остальном фоне, повысив привлекательность для новых клиентов, а так же укрепить вовлеченность уже текущих пользователей.

Предметом исследования бакалаврской работы является бизнес-процесс и системы оплаты ЖКУ клиентами компании и его улучшения для привлечения клиентов путем добавления такой системы [11].

Для достижения поставленной цели необходимо:

- провести анализ бизнес-процесса оплаты жилищно-коммунальных услуг (ЖКУ) компании «Кварплата24»;
- сравнить и рассмотреть существующие аналоги на рынке;
- выбрать технологии и программные средства для реализации системы;
- разработать клиентскую и серверную части мобильного приложения, включающие систему лояльности.

Глава 1 Анализ предметной области

1.1 Характеристика компании «ООО Квартплата 24»

«Компания Квартплата 24 — ИТ-компания, эффективно решающая задачи расчета и учета платы за жилищно-коммунальные услуги, приема и распределения платежей, востребования долгов в полном соответствии с законодательством для товариществ собственников жилья, управляющих и ресурсоснабжающих организаций, информационно-расчетных центров на всей территории РФ» [7].

Информация о деятельности компании и представлена на рисунке 1.

Рисунок 1 – Экосистема облачных сервисов «Квартплата 24»

Основной вид деятельности компании «Кварплата24» согласно ОКВЭД: Разработка программного обеспечения (ОКВЭД код 62.01). Из дополнительных видов деятельности можно выделить такие как: Консультация и работа в области компьютерных технологий (ОКВЭД код 62.02).

На рисунке 2 представлена организационная иерархическая структура компании.

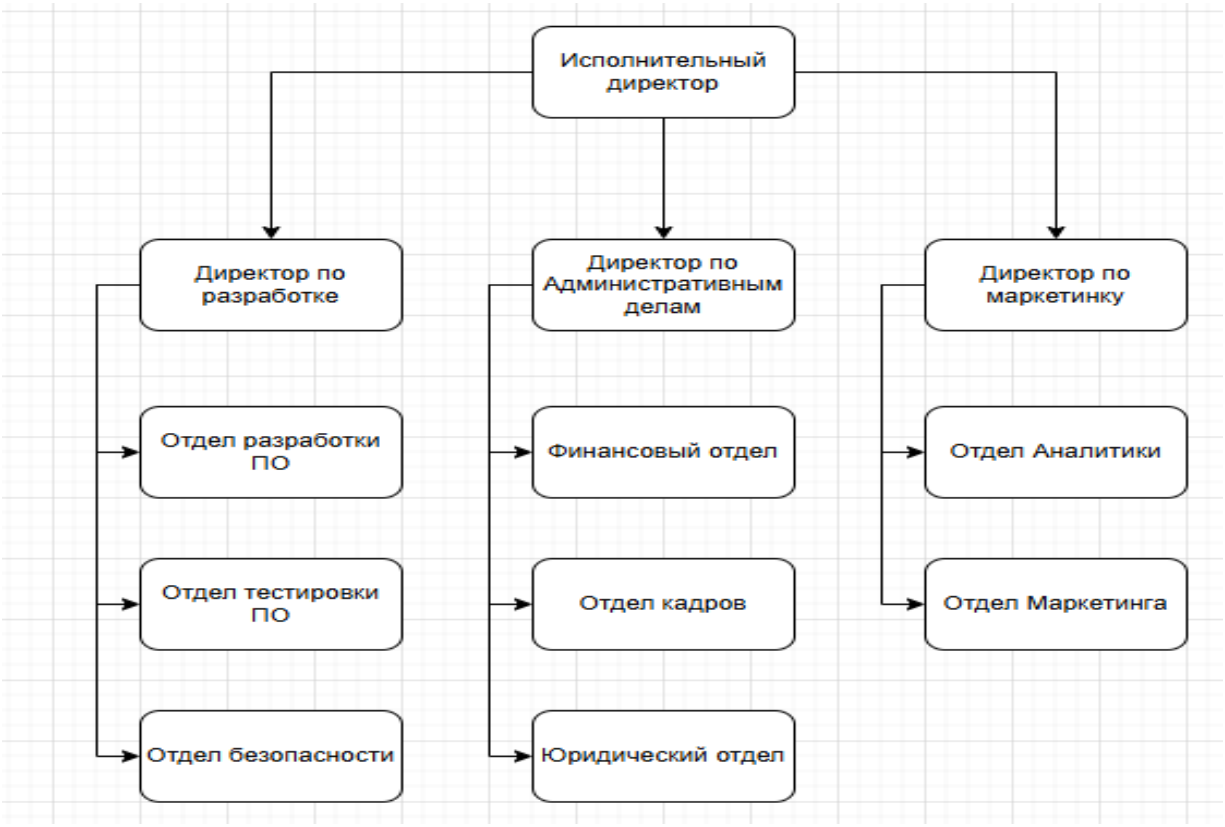

Рисунок 2 – Организационная структура компании «Кварплата24»

Главным в иерархии находится исполнительный директор, который представляет и определяет главное и общее виденье компании, а также ее стратегию и вектор развития. Он занимается анализом и регулированием финансового состояния и устойчивости компании, делегирует задачи между отделами в лицах остальных директоров организационной структуры, которые в свою очередь уже занимаются руководством своих отделов и задач.

1.2 Концептуальное моделирование предметной области

Для моделирования бизнес-процесса и решения поставленной задачи была создана диаграмма «как есть» (AS IS) в нотации BPMN, что позволило детально описать и проанализировать процесс оплаты жилищно-коммунальных услуг с целью его последующего совершенствования и оптимизации.

«Модель «как есть» позволяет описать и систематизировать необходимые и протекающие бизнес-процессы рассматриваемой предметной области [8]. На базе этой модели выявляется необходимость изменения тех или иных бизнес-процессов, которые так же включают в себя объекты в информационной системе. Такую модель называют функциональной, выполняя ее с использованием методологий и графических нотаций»[3].

Бизнес-процесс «Оплата ЖКУ», построенный с использованием методологии BPMN, представлен на рисунке 3.

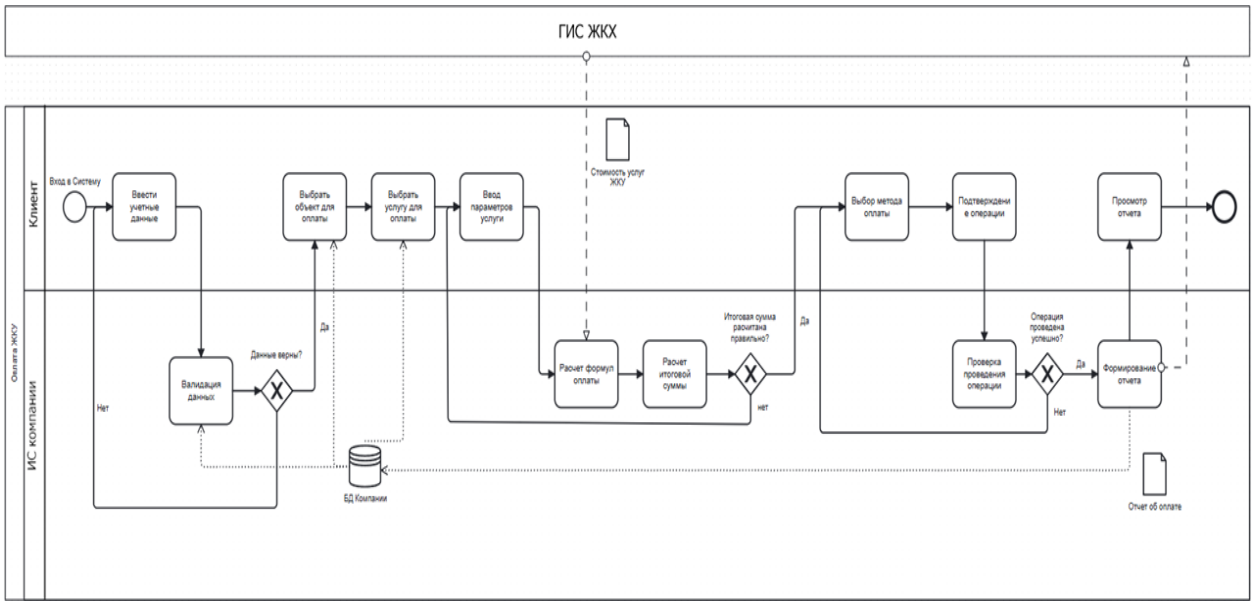

Рисунок 3 - Контекстная диаграмма бизнес-процесса оплаты ЖКУ «как есть»

На диаграмме представлено описание бизнес-процесса оплаты ЖКУ пользователем. Процесс начинается с входа в систему и ввода данных, которые, после валидации данных, позволяют пользователю пройти дальше по пулу. Пользователь выбирает недвижимость, объект для оплаты, данные о которых хранятся в базе данных компании. После этого, он выбирает услугу для оплаты и вводит параметры услуг.

Параметры услуг отправляют эти данные дальше. После этого проводится расчет итоговой суммы с условием на правильный расчет. В случае, если сумма была рассчитана неправильно, то пользователю будет необходимо еще раз ввести данные. После расчета всей суммы, пользователь выбирает метод оплаты, подтверждает операцию, и в случае, если операция была проведена, сформируется отчет, который отправится в БД компании и ГИС ЖКХ. Пользователю будет предоставлен отчет для просмотра, и на этом процесс бизнес оплаты будет завершен.

Расчет итоговой суммы берет свои данные о цене услуг ЖКУ от ГИС ЖКХ, такие как:

- стоимость горячей и холодной воды;
- стоимость электроэнергии;
- другие жилищно-коммунальные услуги.

На данном этапе становится понятно, каким образом протекает бизнес-процесс оплаты ЖКУ, основные действия пользователя, как они взаимосвязаны между собой, поток данных, и какие изменения необходимо выполнить для выполнения поставленной задачи.

1.3 Анализ существующих аналогов

Программных продуктов в сфере ЖКХ и ЖКУ большое количество, однако не всех из них предоставляют обширный функционал, хороший и интуитивно понятный интерфейс, в том числе и возможности кешбека. Были

рассмотрены наиболее популярные приложения.

«Госуслуги.Дом» «официальное Российское единое приложение для оплаты ЖКУ, Госуслуги.Дом. Приложение позволяет подавать показания счетчиков жилищно-коммунальные услуги и оплачивать коммунальные их, видеть свои счета за ЖКХ, транзакции, а также связываться с управляющей компанией своего дома, путем оставления заявок».

«Кварплата+ - это мобильное приложение для оплаты ЖКУ, предоставляемое к скачиванию в различных площадках интернета. Клиенты оплачивают ЖКУ, а также могут настроить уведомления о сроках оплаты или необходимости показания счетчиков, приложение так же включает возможность добавления различных ЖКУ для оплаты по qr коду или реквизитам».

«Мобильное приложение «Платосфера» позволяет оплачивать услуги ЖКУ с мобильного телефона, в том числе счета за электроэнергию и воду. Так же имеет встроенный функционал по добавлению различных услуг ЖКУ для оплаты путем сканирования qr кода или добавления реквизитов для оплаты» [10].

Для оценки возможностей разработки и сравнения мобильного приложения с существующими аналогами на рынке были выделены следующие критерии оценки:

- удобство использования,
- цифровая безопасность,
- интуитивно понятный интерфейс,
- расширенный функционал,
- наличие системы лояльности.

В настоящее время представленные приложения оцениваются по «указанным критериям по пятибалльной шкале, где «1» означает полное несоответствие, а «5» — полное соответствие требованиям. Результаты оценки программных продуктов представлены в таблице 1» [3].

Таблица 1- Анализ существующих на рынке систем

Критерий	Госуслуги.Дом	Квартплата+	Платосфера
Удобство использования	4	4	4
Цифровая безопасность	4	4	4
Интуитивно понятный интерфейс	3	3	3
Расширенный функционал	3	3	2
Система лояльности	0	1	0
Итого	14	16	13

На основании проведённого сравнительного анализа следует вывод, что рассматриваемые программные продукты не соответствуют требованиям, предъявляемым к аналогичным решениям на рынке в рамках решаемой задачи.

1.4 Разработка требований к информационной системе

На основании анализа аналогов, были созданы требования к информационной системе, реализованной как мобильное приложение для оплаты ЖКУ с возможностью расчета и получения скидки на оплату, которая реализована в виде системы лояльности в методологии FURPS+, которые отображены и представлены в таблице 2.

«FURPS+ является классификацией требований к программным системам. Данная методология и данные на основании сравнения аналогов, позволила составить следующие требования, включая основные, требования к удобству, надежности, производительности и поддержки» [4].

Таблица 2– Требования к информационной системе в методологии FURPS+

Вид требований	Содержание Требований
F. функциональные требования к информационной системе	- привязка банковской карты, - добавление услуги для оплаты, - система лояльности, - проверка расчета итоговой стоимости, - изменение формул расчета кешбека, - просмотр за что начисляется кешбек, - редактирование данных клиента, - проверка отчетности транзакций, - расчет кешбека на основании прошлых транзакций, - учет прошлых транзакций, - предоставление отчета об оплате, - выбор услуги и недвижимости для оплаты, - проверка отчета по формированию кешбека, -добавление недвижимости, - подача показаний ЖКУ, - авторизация в приложении, - оплата ЖКУ
U. удобство использование информационной системы	- удобный интуитивно понятный пользовательский интерфейс - возможность автозаполнения форм
R. надежность информационной системы	- проверка введённых показателей ЖКУ
P. производительность информационной системы	- Android 5.1 или более поздние версии
S. поддерживаемость информационной системы	- предоставление приложения на площадках мобильного сегмента
+- ограничения информационной системы	- расчет кешбека проводится только на основе транзакций платежной системы МИРв.

Данные требования соответствуют необходимому функционалу для разработки приложения в рамках поставленной задачи.

1.5 Разработка и анализ бизнес-модели «Как должно быть»

На основе модели «как есть» (AS IS) была разработана диаграмма «как должно быть» (TO BE), которая позволила выявить необходимые изменения и улучшения существующего бизнес-процесса, ранее описанного с помощью модели «как есть» (AS IS).

Диаграмма «как должно быть» (TO BE), выполненная в нотации BPMN, отражает бизнес-процесс расчёта кешбэка системой лояльности и её взаимодействие с процессом оплаты жилищно-коммунальных услуг (рисунок 4).

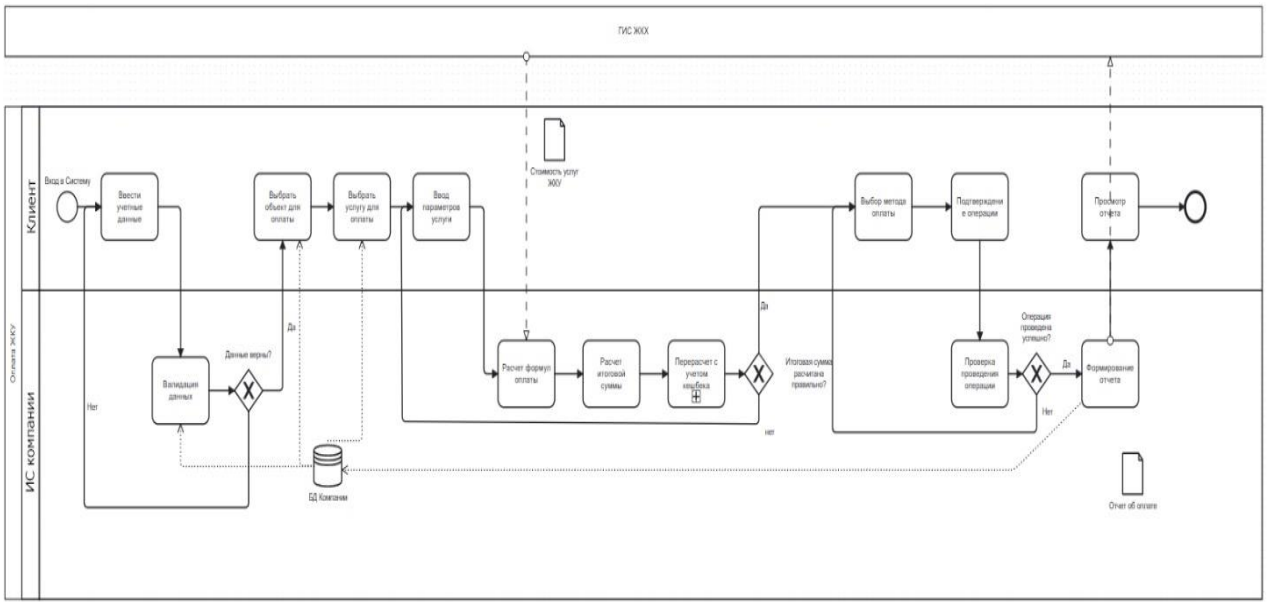

Рисунок 4 – Диаграмма бизнес-процесса «Оплата ЖКУ»

Для подробного понимания, работы системы лояльности, суб-процесс «Перерасчет с учетом кешбека» был декомпозирован и представлен на рисунке 5.

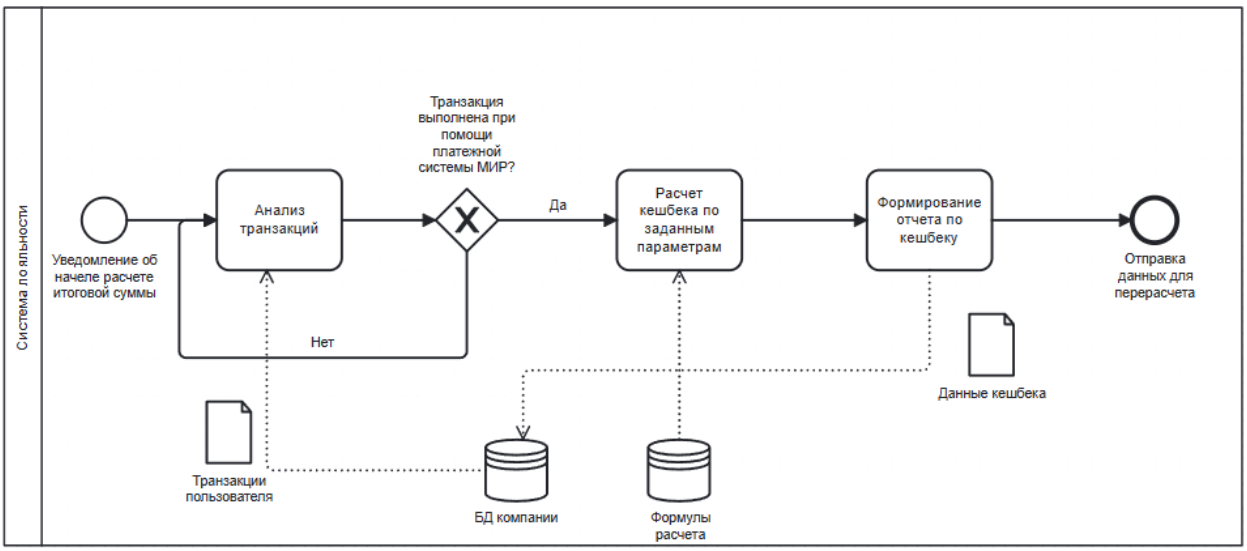

Рисунок 5 – Декомпозиция суб-процесса «Как должно быть»

Система лояльности, получив данные о начале расчете итоговой суммы проводит анализ предыдущих транзакций пользователя, в случае если транзакции были выполнены с использованием платежной системы МИР, на основании этих данных проходит расчет предоставляемого кешбека по заданным в базе данных формул расчета параметров, после проводится формирование отчета о предоставленном кешбеке, и он отправляется в базу данных компании, в виде «Данных кешбека». После отправки, этот отчет отправляется в процесс расчета итоговой суммы, для предоставления пользователю кешбека на основании его транзакций.

Формулы расчета кешбека прописываются администратором, который управляет системой лояльности. На основании предыдущих транзакций при использовании приложения компании, а «Данные кешбека» содержат в себе прямую информацию о начисленных для оплаты средствах.

В ходе моделирования бизнес-процессов при помощи диаграмм «как есть» и «как должно быть» были выявлены необходимые улучшения и модификации текущего бизнес-процесса оплаты, а также необходимые для выполнения поставленной задачи функциональные требования. В этих бизнес-

процессах были определены все необходимые процессы для проведения оплаты ЖКУ с использованием системы лояльности для компании.

Выводы по главе 1

В первой главе был проведен и описан подробный анализ деятельности компании «Квартплата 24». Были выявлены необходимые изменения или улучшения для бизнес-процессов и возможностей системы.

Для полного понимания предметной области и бизнес-процессов были построены контекстные диаграммы «как есть» и «как должно быть» при помощи методологии BPMN. Проведен анализ бизнес-процесса оплаты ЖКУ, рамках выполнения поставленной задачи. И обосновано решение о необходимости создания новой информационной системы в виде приложения, которая бы соответствовала требованиям.

Глава 2 Проектирование мобильного приложения

2.1Логическое моделирование мобильного приложения

Моделирование логики мобильного приложения является одной из важнейших частей разработки продукта. Данный подход обеспечивает формирование чёткой и структурированной логики данных, а также позволяет определить ключевые объекты, необходимые для реализации проекта. При проектировании и разработке мобильного приложения для оплаты жилищно-коммунальных услуг с функцией кешбэка применялся унифицированный язык моделирования UML, который является интуитивно понятным и простым, помогая полноценно сфокусироваться на выполнении поставленной задачи и описать необходимые для реализации приложения.

2.1.1 Диаграмма вариантов использования

Диаграмма вариантов использования описывает уровень доступа функционала каждой группе указанных пользователей. В данном случае основным пользователем мобильного приложения будет являться «Клиент», контролирующим звеном приложения будет являться «Администратор».

Варианты использований актеров «Клиент» и «Администратор» представлены на рисунке 5.

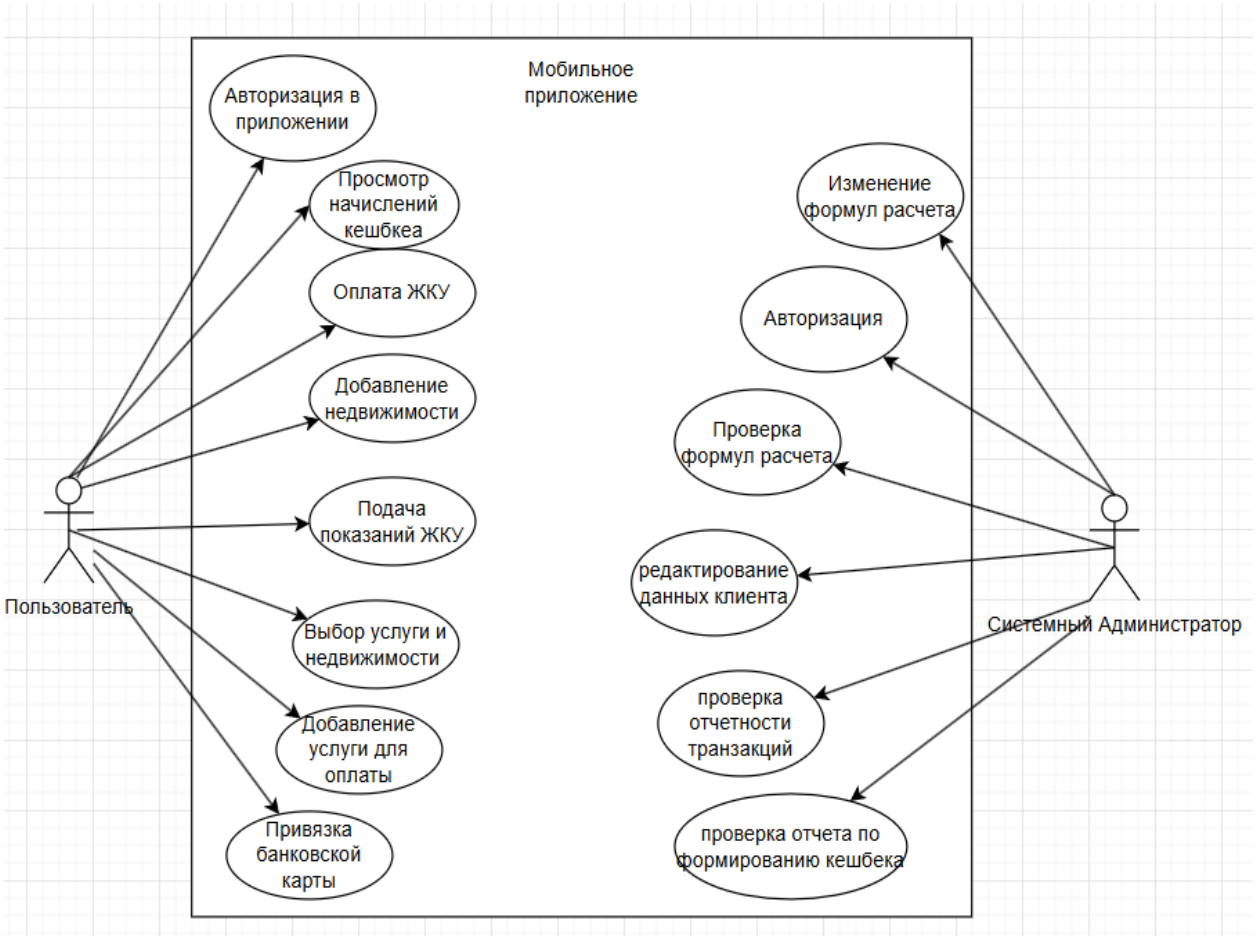

Рисунок 5 - Диаграмма вариантов использований

Клиент (пользователь мобильного приложения):

- авторизация в приложении;
- просмотр начислений кешбека;
- оплата ЖКУ;
- добавление недвижимости;
- подача показаний ЖКУ;
- выбор услуги и недвижимости;
- добавление услуги для оплаты;
- привязка банковской карты.

Администратор:

- изменение формул расчета кэшбека;

– авторизация;
– проверка формул расчета;
– редактирование данных клиента;
– проверка отчетности транзакций;
– проверка отчета по формированию кешбека.

2.1.2 Диаграмма классов мобильного приложения

«Далее было осуществлено проектирование диаграммы классов UML, предоставляющее взаимосвязь объектов в системе, оно иллюстрирует структуру ИС, описывает атрибуты, классы, методы и отношения между ними. Классы же представляют собой абстракцию для объектов, которые содержит система. Каждый класс имеет в себе собственные поля данных и операций, которые определяют поведение этого класса» [5].

На рисунке 6 представлена диаграмма классов. Она служит основой для дальнейшей реализации и понять какие именно процессы будут происходить в дальнейшем и какие данные требовать.

.

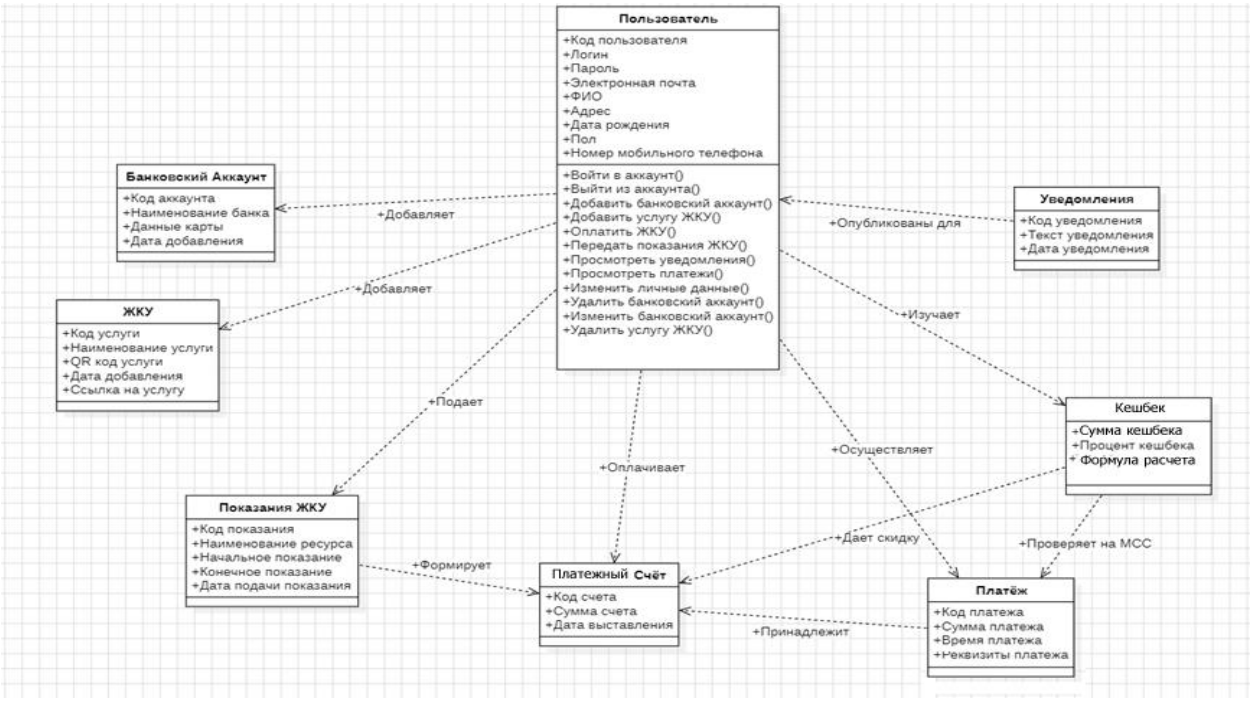

Рисунок 6 – Диаграмма классов модели мобильного приложения

«Диаграмма классов — структурная диаграмма языка моделирования UML, демонстрирующая общую структуру иерархии классов системы, их кооперацій, атрибутов (полей), методов, интерфейсов и взаимосвязей между ними. Широко применяется не только для документирования и визуализации, но также для конструирования посредством прямого или обратного проектирования» [5].

«Диаграмма включает следующие классы: «Пользователь», «Банковский аккаунт», «ЖКУ», «Показания ЖКУ», «Счёт», «Платёж», «Категории Кешбека» и «Уведомления». Так же продемонстрированы отношения между ними. Это позволяет создать более полную картину структуры ИС, и понять как компоненты, которые относятся к ней, будут взаимодействовать друг с другом» [5].

2.2 Разработка логической и физической модели данных

«Логическая модель данных представляет собой структуру данных, которая определяет сущности, их атрибуты и связи между ним без привязки к конкретной реализации. Она является расширением концептуальной модели. С помощью логической модели данных бизнес-аналитики и архитекторы данных могут визуализировать операционные или транзакционные процессы на диаграмме взаимоотношений между сущностями. Логические модели данных определяют, как работают и взаимодействуют объекты данных. Каждая таблица описывает объекты данных и их атрибуты в знакомых для бизнеса терминах. Для всех объектов назначается первичный ключ (PK) для идентификации атрибутов в каждой строке. Некоторые объекты содержат внешние ключи (FK), которые описывают их связь с другими объектами в отношениях «один ко многим»»[6].

Данная модель данных представлена на рисунке 7.

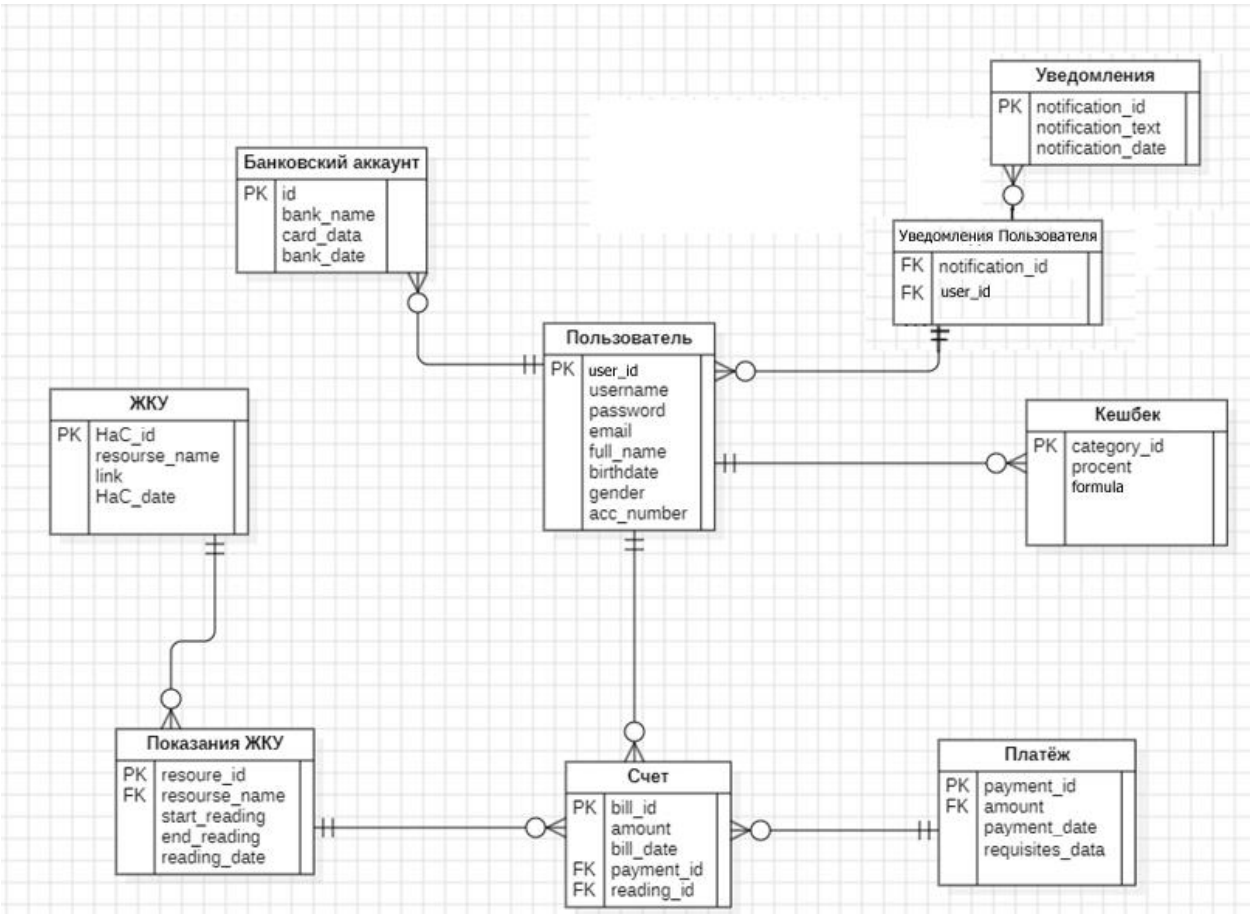

Рисунок 7 – Логическая модель данных мобильного приложения

Логическая модель описывает отношения между классами. Множество банковских аккаунтов может принадлежать одному пользователю, множество счетов могут предоставляться одному пользователю, на одну ЖКУ может быть множество показаний, а множество счетов может быть оплачено одним платежом. В свою очередь множество категорий кешбека предоставлены лишь одному пользователю.

«Физическая модель данных показывает устройство и структуру общей базы данных, показывая их тип, взаимосвязь таблиц, первичные и внешние ключи. Она представляет типы информации и данных, которые лежат в основе физической базы данных приложения» [4].

На рисунке 8 показана физическая модель расчета коммунальных платежей.

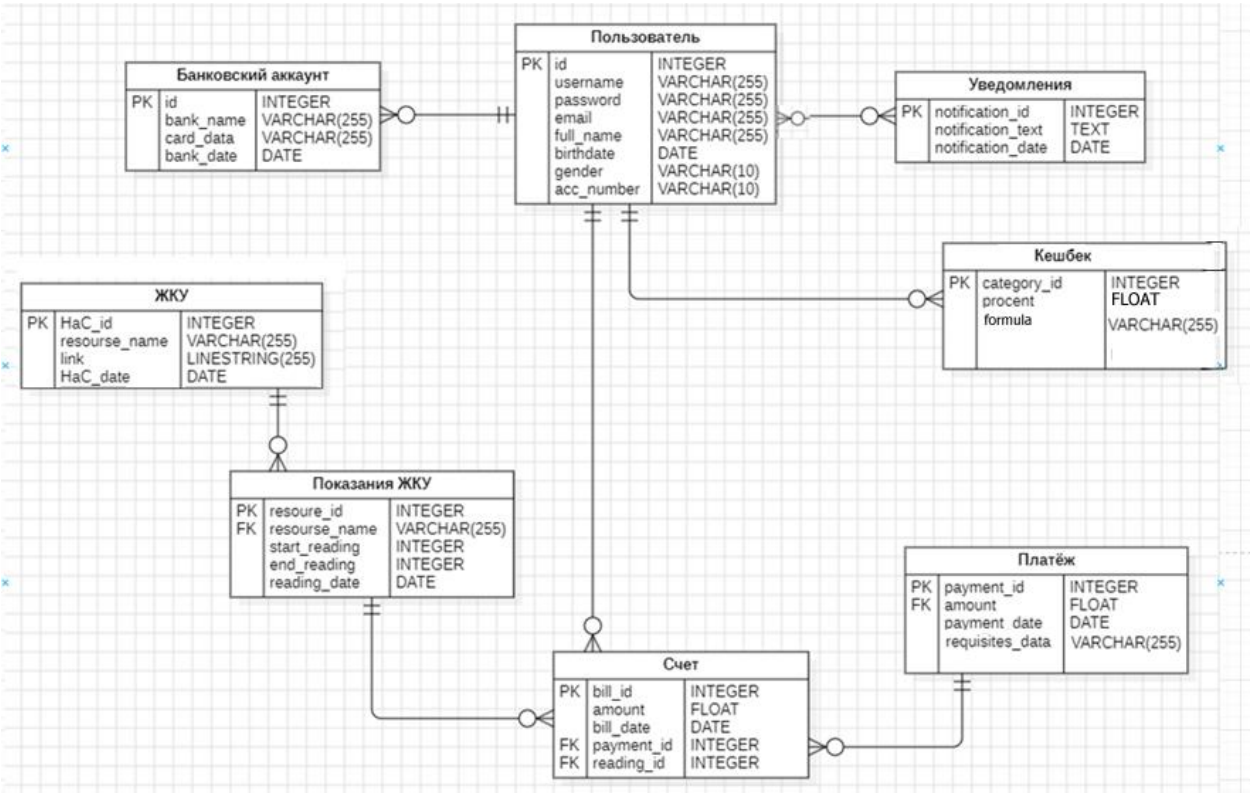

Рисунок 8 – Физическая модель данных

«Физическая модель представляет реализацию логической модели в конкретной СУБД. По сравнению с логической моделью, здесь отсутствуют некоторые технические детали, но сохранены все ключевые сущности и их взаимосвязи. Данная модель была специально адаптирована под ИС и СУБД» [6].

Основное внимание в физической модели уделено связям между таблицами. Например, видно, что платежи связаны с счетом через payment_id, а ЖКУ имеет связь с resource_name в Показания Жку. Отсутствие некоторых атрибутов (по сравнению с логической моделью) может свидетельствовать о проведенной нормализации данных или оптимизации структуры для повышения производительности.

2.3 Разработка диаграммы компонентов

«В ходе выполнения работы, была так же спроектирована диаграмма компонентов мобильного приложения Диаграмма компонентов — это структурная диаграмма языка унифицированного моделирования, она описывает особенности физического представления системы. Диаграмма компонентов позволяет определить архитектуру разрабатываемой системы, установив зависимости между программными компонентами» [14].

Данная диаграмма необходима для описания и дальнейшей реализации ИС в рамках создания бизнес-решения, так как позволяет наиболее точно определить взаимодействие частей системы друг с другом, что помогает в дальнейшем ее проектировании. Диаграмма компонентов представлена на рисунке 9.

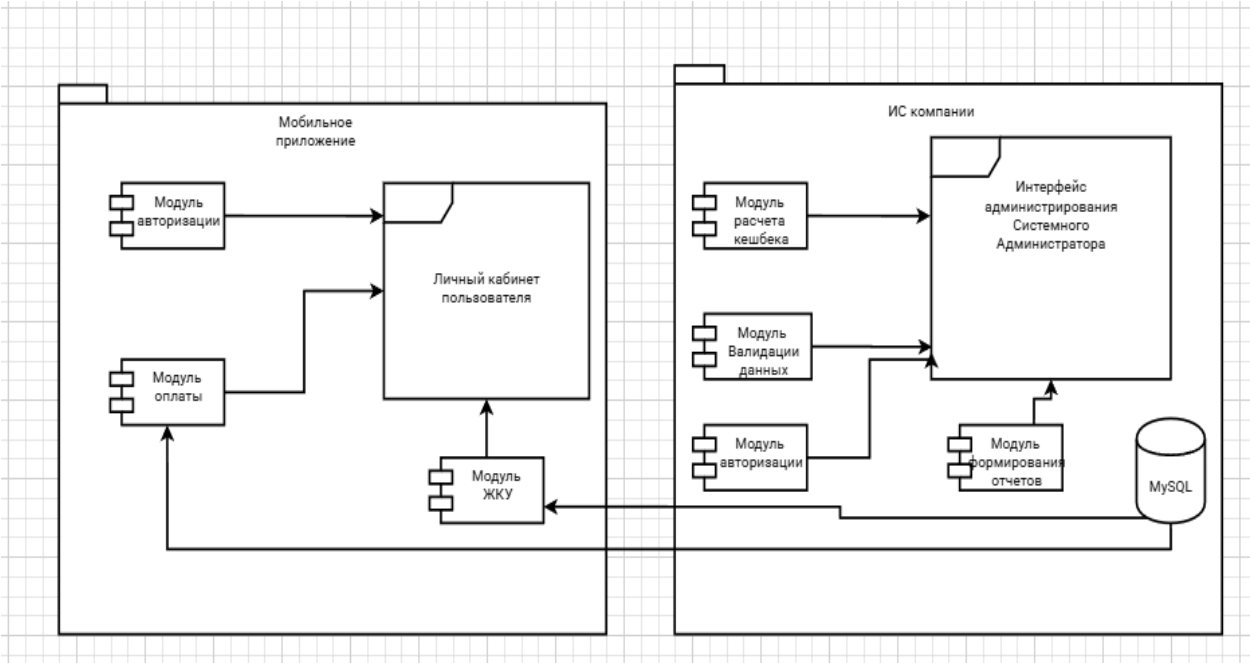

Рисунок 9–Диаграмма компонентов ИС «Оплата ЖКУ с использованием кешбека»

Представленная на данном рисунке архитектура, показывает все возможности и состав данных и компонентов, которые используются в работе сервиса. Данное решение представляет комплексный подход для автоматизации многих процессов, таких как расчет и управления платежами за коммунальные услуги с предоставлением кешбека. Архитектура мобильного приложения были спроектированы для предоставления интуитивно понятного интерфейса как для пользователей, так и для сотрудников управляющих компаний. В основе системы лежит принцип прозрачности и доступности всех операций, что позволяет минимизировать ошибки в начислениях и сократить время на обработку запросов.

Личный кабинет жильца предоставляет полноценный комплексный функционал через персональный мобильный-интерфейс. В личном кабинете реализованы следующие ключевые возможности: просмотр текущих начислений с детализацией по видам услуг, доступ к архиву платежей за предыдущие периоды, получение электронных квитанций установленного образца. Интерфейс личного кабинета адаптирован для различных устройств и включает инструменты для анализа потребления коммунальных ресурсов с возможностью построения сравнительных графиков. Все финансовые операции защищены многоуровневой системой авторизации и протоколируются для обеспечения прозрачности расчетов.

Для сотрудников управляющих компаний система предлагает набор инструментов, позволяющих оперативно работать с данными. Это включает в себя проверку корректности начислений, формирование отчётов для бухгалтерии, проверку категорий кешбэков и их настройку. Автоматизированные алгоритмы расчётов сокращают время на рутинные операции, а встроенные механизмы проверки данных помогают избежать ошибок.

Надёжность системы обеспечивается за счёт использования современной базы данных MySQL, регулярного резервного копирования и многоуровневой защиты персональных данных. Автоматические проверки

расчётов и встроенные механизмы анализа помогают выявлять несоответствия на ранних этапах. Система также поддерживает интеграцию с платёжными сервисами, что позволяет жильцам оплачивать счета без необходимости покидать личный кабинет.

Выводы по главе 2

Во второй главе было показано логическое проектирование мобильного приложения в рамках поставленной задачи. Были построены модели данных, которые включают в себя физические и логические, диаграмма вариантов использования, диаграмма классов, диаграмм компонентов. Были подробно описаны основные способы и возможности взаимодействия актеров в лицах пользователя, администратора и приложения. Были построены диаграммы классов, которые помогут более четко определить рамки разработки и необходимый функционал и бекенд приложения. Так же была описана необходимая для физической разработки архитектура приложения, которая бы в полной мере могла бы реализовать заявленный и описанный функционал мобильного приложения.

Глава 3 Разработка мобильного приложения

3.1 Выбор технологий и программных средств разработки

Для разработки приложения, выбрана среда разработки (IDE), инструмент, упрощающий работу над приложением и содержащий в себе все необходимые для этого инструменты.

В качестве среды разработки было выбрано программное обеспечение Android Studio. Этот выбор основан на сравнительном анализе с другими средами, такими как Visual Studio Code, JetBrains IntelliJ IDEA и Eclipse. Результаты сравнения средств разработки представлены в таблице 3 [16].

Таблица 3 – Сравнение средств разработки

Критерии	AndroidStudio	VisualStudio	Intelij IDEA	Eclipse
Удобство использования	+	+	+	-
Функциональность	+	+	+	+
Поддержка языков	Kotlin, Java	C#, C++	Kotlin, Java	Java, C++
Модель распространения	бесплатно	бесплатно/ платно	бесплатно/ платно	бесплатно
Итог	4	3	3	2

Для работы с API банковских транзакций был выбран Retrofit, библиотека для работы с REST API, которая позволяет отправлять HTTP-запросы и обрабатывать ответы в удобном виде. С ее помощью можно интегрировать внешние сервисы, получать данные о транзакциях и анализировать их для расчета максимального кешбэка, что позволит разработать приложение легче, чем в аналогичных ей библиотеках. Retrofit поддерживает асинхронные запросы, благодаря чему она может выполнять сетевые операции без задержки интерфейса пользователя.

Для организации работы с удаленным MySQL-сервером в мобильном приложении используется специально разработанный REST API, который обеспечивает безопасный обмен данными между клиентом и сервером. Все запросы к базе данных выполняются через этот API, что позволяет минимизировать нагрузку на мобильное устройство и гарантировать целостность данных. Для реализации сетевых запросов применяется библиотека Retrofit, которая обеспечивает стабильное соединение и удобную обработку ответов сервера. MySQL был выбран в качестве основной СУБД благодаря своей надежности, производительности и возможности масштабирования. На сервере развернута нормализованная база данных, включающая таблицы пользователей, платежей, бонусных начислений и транзакций. Для защиты данных реализована система шифрования конфиденциальной информации и регулярного резервного копирования.

Для обеспечения безопасности и авторизации пользователей в мобильном приложении будет использоваться Firebase Authentication который предоставляет удобные инструменты для управления учетными записями пользователей, аутентификации через электронную почту, Google-аккаунты и другие способы входа. Она будет использоваться для защиты и верификации данных с сервером.

Для создания пользовательского интерфейса использован Jetpack Compose современный язык программирования и метод разработки, направленный на создание пользовательского интерфейса, UI в Android, он позволяет создавать динамические интерфейсы с использованием декларативного подход, обеспечивая более плавную и простую разработку мобильного ПО.

Для интеграции с платежными системами и расчета кешбэка на основе проведенных через приложение транзакций с использованием платежной системы МИР, в приложении будет использоваться система обработки транзакций с динамическим анализом данных. Данный механизм позволит

анализировать начисления кешбэка и предоставлять пользователю информацию о наиболее выгодных вариантах возврата средств.

По итогам описания и анализа выбранных инструментов и средств разработки, а также на основе таблицы номер 3, был сделан вывод, что программное обеспечение и библиотеки в виде Android Studio, MySQL, Retrofit, Room, Firebase и Jetpack Compose позволят создать мобильное приложение, которое соответствует требованиям выпускной квалификационной работы

3.2 Описание функциональности и реализации проекта

3.2.1 Пользовательский интерфейс и функционал приложения

Была так же рассмотрена разработка мобильного приложения с точки зрения дизайна пользовательского интерфейса и его функций. В качестве программы для разработки дизайна, фронтэнда и бекенда приложения была использована среда разработки (IDE) Android Studio.

Был реализован функционал авторизации пользователя в мобильном приложении. На базовом уровне, она представляет собой два поля, которые заполняются пользователем для получения доступа в свой личный кабинет: Номер лицевого счета или логин и пароль. В случае, если пользователь забыл пароль, он может нажать на гиперссылку «Забыли пароль?», которая перенаправит его на сайт с поддержкой компании. После заполнения авторизации клиент может войти в личный кабинет, путем нажатия кнопки «Войти». Экран авторизации в приложение представлен на рисунке 10.

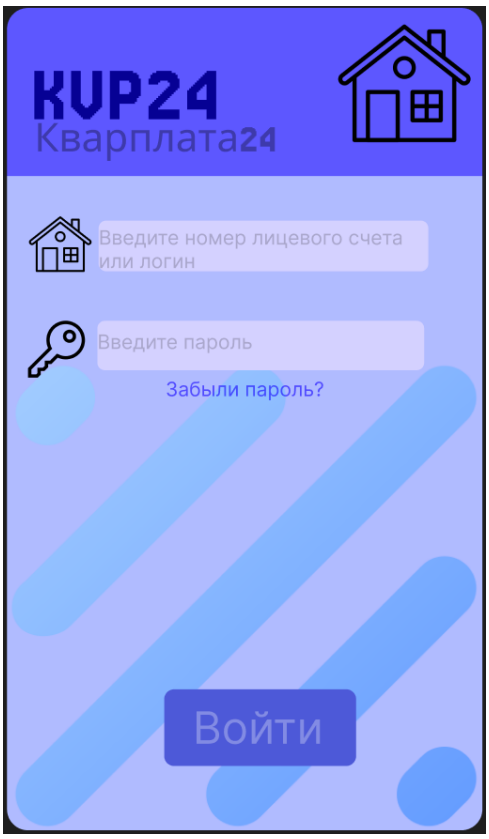

Рисунок 10 – Экран авторизации

Дизайн сочетает в себе понятный интуитивный функционал, а также воплощают в себе необходимые функции для работы приложения.

После успешной авторизации, пользователь попадает на главную страницу приложения, где представлены его личные данные и краткие сведения, такие как: «Лицевой счет»; «Управляющая организация»; «Адрес»; «Привязанный к аккаунту e-mail», краткое описание сроков подачи новых показаний и необходимая сумма к оплате за текущий период времени и кнопка оплаты привязанной в приложении картой. Так же отдельной кнопкой выделены настройки приложения. Панель снизу представляет собой кнопки навигации, которые перемещают пользователя по функциям и страницам мобильного приложения (рисунок 11).

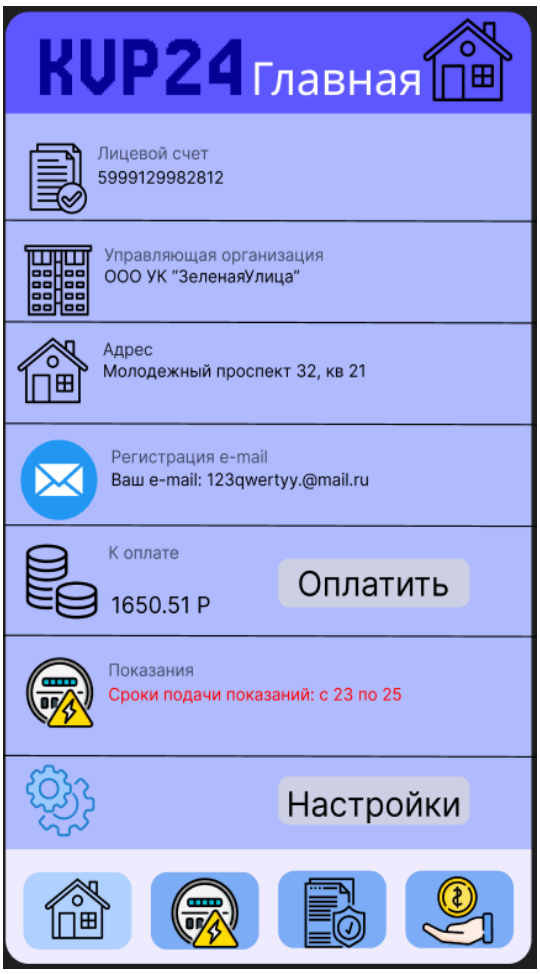

Рисунок 11 – Главная страница мобильного приложения

При использовании функциональных кнопок в приложении, пользователь переместится на страницу подачи показаний, где может ввести новые показания за услуги ЖКУ, такие как холодная вода, горячая вода, электроэнергия, а также добавить любую другую услугу ЖКУ, путем ввода реквизитов, или сканирования QR-кода, путем нажатия на кнопку, которая представлена слева от поля ввода реквизитов. Нажатием на кнопку «Сохранить» пользователь сохраняет вписанные им новые данные.

При переходе на третью страницу, пользователь увидит все начисления по тарифам за текущий и предыдущие месяцы, что представлено на рисунке 12.

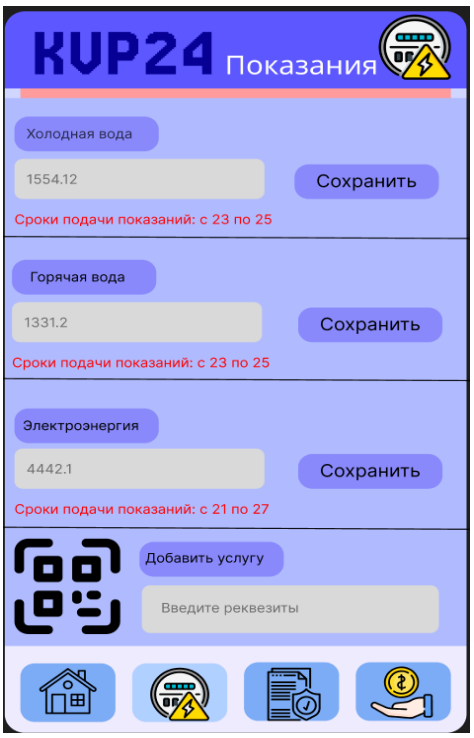

Рисунок 12 – Страница ввода показаний счётчиков

Страница «Платежи», которая представлена на рисунке 13, позволяет пользователю увидит информацию по всем платежам.

Рисунок 13– Страница начислений

Отдельно стоит заметить, что сумма не всегда равна платежу, так как часть суммы оплачивается кешбеком самого сервиса, следственно из-за чего клиент и получает скидку в оплате ЖКУ, что представлено на рисунке 14.

Рисунок 14 – Страница платежей

Страницы оплаты и кешбека приложения, представленные на рисунках 15 и 16, пользователь сможет ввести или изменить данные своей карты, которая по умолчанию используется для оплаты внутри приложения. После привязки карты, приложение получает данные о транзакциях и предоставляет сводку о том, какое количество кешбека было начислено пользователю.

Рисунок 15 – Страница «Оплата»

Последней реализованной страницей, является «Кешбек». Страница устроена достаточно просто, пользователь может наглядно посмотреть за какие транзакции и от каких компаний ему пришел кешбек, который отображается над транзакциями под картой. В случае необходимости, пользователь может скачать файл с условиями начисления и списком партнеров сервиса, для получения подробной информации.

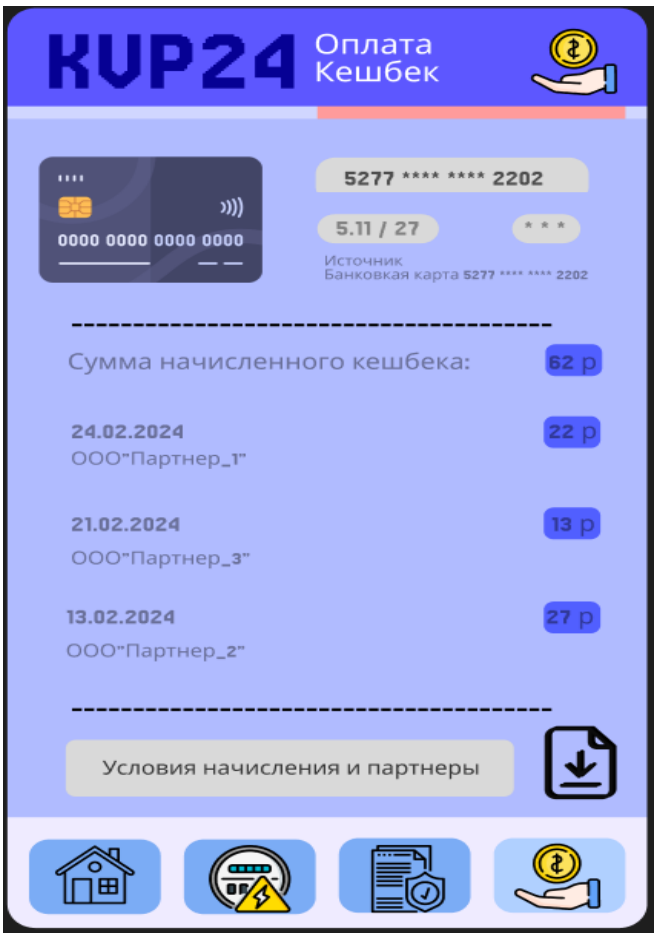

Рисунок 16– Страница кешбек

Данный интерфейс предоставляет необходимый и интуитивно понятный дизайн для реализации и работы мобильного приложения. Он также показывает отличие приложения от других существующих на рынке, и выглядит уверенным с точки зрения предоставляемого функционала.

3.2.2 Реализация backend части мобильного приложения

Целью этой части выпускной квалификационной работы было создание системы, которая позволила бы пользователем пользоваться системой лояльности и использовать такой функционал, как авторизация, оплата ЖКУ, просмотр расчета кешбека, оплата ЖКУ кешбеком, привязка банковской карты для оплаты, добавлять услуги ЖКУ для оплаты, передавать данные ЖКУ, чтобы оплатить их в дальнейшем.

Backend мобильного приложения реализован при помощи языка Kotlin с использованием Ktor в качестве веб-фреймворка [20]. База данных хранится и работает при помощи MySQL что позволяет добиться необходимого функционала и легкости в использовании для приложения [21].

Сама суть разработки backend-части этого приложения заключается в его модульности. Так, каждый логически независимый блок выделен в отдельный модуль, что упрощает процесс изменения данных внутри системы для интеграции или изменения каких-либо данных. В процессе разработки данного проекта для реализации данной концепции были выделены следующие модули:

- auth – отвечает за регистрацию и авторизацию пользователей с использованием Firebase Authentication.
- profile – управление профилем пользователя, включая изменение личных данных и просмотр истории кешбэка.
- transactions – обработка и хранение информации о банковских транзакциях с использованием Room (MySQL).
- api – модуль для взаимодействия с внешними API банковских транзакций через Retrofit.
- ui – слой пользовательского интерфейса, разработанный на Jetpack Compose, который отвечает за отображение информации и взаимодействие с пользователем.

Приложение построено на архитектурном паттерне MVVM (Model-View-ViewModel), который разделяет логику работы с данными, бизнес-логику и пользовательский интерфейс:

- Model (Модель) – в свою очередь управляет данными. При разработке приложения используется Room Database который нужен для локального хранения транзакций, а также Firebase Firestore для хранения информации о самих пользователях.

- ViewModel (Модель представления) – управляет логикой отображения данных. Используется Jetpack ViewModel, который получает данные из базы и передает их во View.
- View (Представление) – отвечает за отображение данных. В проекте используется Jetpack Compose, который позволяет создавать декларативный UI с минимальными усилиями.»

Для хранения данных был выбран MySQL. Благодаря этому объекты Kotlin могут управлять данными, не прибегая к написанию SQL-запросов вручную.

- Entity (TransactionEntity.kt) – в свою очередь показывает структуру таблицы в базе данных.
- DAO (TransactionDao.kt) – интерфейс для выполнения запросов к базе данных (добавление, получение, удаление транзакций).
- Database (AppDatabase.kt) – главный класс базы данных, который управляет её созданием и взаимодействием с DAO.

Для интеграции с банковскими API был использован Retrofit, который обеспечивает удобный способ отправки запросов и обработки ответов в формате JSON [22].

- TransactionApi.kt – интерфейс, содержащий методы для получения и отправки данных о транзакциях.
- RetrofitClient.kt – singleton-класс, создающий экземпляр Retrofit и настраивающий подключение к API.

Для аутентификации пользователей и хранения их данных используется Firebase Authentication и Firestore. Это позволяет легко управлять пользователями и их настройками, обеспечивая безопасность хранения данных [15].

- FirebaseAuthManager.kt – отвечает за регистрацию и авторизацию пользователей через Firebase.

– FirebaseFirestoreManager.kt – управляет сохранением и загрузкой данных пользователей.

Пользовательский интерфейс

UI был разработан на Jetpack Compose, что делает его гибким и удобным для модификации.

– MainScreen.kt – основной экран приложения, включающий форму ввода транзакции и отображение информации о кешбэке.
– ProfileScreen.kt – экран профиля, где пользователь может изменить личные данные.
– TransactionHistoryScreen.kt – экран истории транзакций с фильтрацией по дате и сумме.

Для управления зависимостями и сборки проекта используется Gradle Kotlin DSL (build.gradle.kts) [19]. В проекте настроены зависимости для Retrofit, Room, Firebase, Jetpack Compose и MySQL, обеспечивая стабильную работу всех компонентов [17].

Для создания виртуального окружения были установленна AndroidStudio, программная среды, с совместимостью версии API 24, а также build language в виде Kotlin DSL, необходимо установить зависимости для будущей реализации приложения. Зависимости для создания проекта прописываются в файле build.gradle.kts (ModuleL: app), они представлены в приложении А (рисунок А.1).

Для реализации получения и расчета кешбека, необходимо создать такие файлы как TransactionApi.kt, RetrofitClient.kt CashbackCalculator.kt и CashbackRules.kt, которые представлены в приложениях Б, В, Г, Д.

TransactionApi.kt, код которого представлен в приложении Б (рисунок Б.2), управляет возможностью получения данных из API самой транзакции, передавая такие данные как amount – сумму, date – дату и код транзакции –, который сообщает о соответствии транзакции через платежную систему МИР.

RetrofitClient.kt в свою же очередь, отвечает за прямой запрос самому API на получение данных о транзакциях, после чего, метод, указанный выше, получает из него данные для передачи в следующие два метода, код данного файла представлен в приложении В (рисунок В.3).

В приложении Г (рисунок Г.4) представлен файл CashbackRules.kt, который отвечает за регулирование правил кешбека.

В приложении Д (рисунок Д.5), показана формула расчета в файле CashbackCalculator.kt, который берет данные из файла CashbackRules.kt и рассчитывает на их основе и на базе данных из транзакций, которые указаны в двух предыдущих методах сам кешбек.

Так же нужно отметить необходимость создания файла TransactionRoutes.kt, который бы возвращал сумму кешбека в само приложение и давал возможность использовать её в качестве оплаты, код файла представлен в приложении Е (рисунок Е.6).

3.3 Тестирование мобильного-приложения

Для проверки функциональности, производительности и соответствия поставленным требованиям приложения, было выполнено тестирование методом «Черного ящика».

«Тестирование чёрного ящика или поведенческое тестирование — метод тестирования функционального поведения объекта с точки зрения внешнего мира, при котором не используется знание о коде тестируемого объекта. [9] Стратегия поведенческого теста исходит из технических требований и их спецификаций. Из достоинств данного метода можно выделить то, что он помогает найти ошибки, которые невозможно обнаружить методом «белого ящика»; − «Черный ящик» позволяет быстро выявить ошибки в функциональных спецификациях (в них описаны не только входные значения, но и то, что мы должны в итоге получить); − в виду того, что тесты

проводятся с позиции пользователя; − составлять тест-кейсы можно сразу после подготовки спецификации.

Недостатки метода: − основным недостатком метода «черного ящика» является возможность пропуска границ и переходов, которые не очевидны из спецификации самого приложения» [2].

Данные тестирования представлены в таблице 4.

Таблица 4 – Реализация тестирования мобильного приложения

Тест	Действие	Результат
информирующие сообщения	авторизация в приложении	успешная авторизация
сообщение об успешной операции	проверка входа в кабинет	вход выполнен успешно, переход на главную страницу
восстановление пароля	проверка функции восстановления пароля	письмо с инструкциями отправлено, пароль изменен
добавление услуг ЖКУ	проверка добавления услуг	услуги успешно добавлены
внесение показаний	проверка внесения показаний счетчиков	показания успешно внесены и сохранены в базе данных
управление оплатами	проверка отображения детализации сумм к оплате	суммы корректно отображены, детализация верна
выбор услуг недвижимости	проверка выбора услуг и недвижимостей	выбор услуг успешен
адаптивность интерфейса	проверка отображения на различных устройствах	интерфейс корректно отображается не на всех устройствах
ошибка на стороне клиента	проверка обработки ошибок, вызванных пользователями	ошибки корректно обработаны, уведомления отображены
ошибка на стороне сервера	проверка обработки ошибок сервера	ошибки сервера не выявлены

Результатами тестирования данным методом, было доказано, что мобильное приложение работает вполне корректно. Основополагающие функции, которые были описаны в требованиях, а также расчет кешбека, работали в соответствии с требованиями и спецификациями.

3.4 Расчет экономической эффективности

«Для оценки стоимости реализации и экономической эффективности ИС для привлечения клиентов в виде приложения определены следующие категории затрат:

- оплата труда сотрудников, которые участвуют в реализации приложения,
- расходы на хостинг, поддержку и администрирование приложения,
- расходы на оплату электроэнергии.

В таблице 5 были представлены заработные платы сотрудников.

Таблица 5 – Расчет суммы оплаты труда специалистов, привлечённых к разработке приложения для компании «Квартплата24»

Должность	Величина оплаты труда специалиста за 1 час. руб	Величина на трудозатраты	Величина заработной платы, руб
Программист	800	102,5	82000
Специалист компании (Маркетолог)	350	80	28000
Директор	460	40	18400
Экономист или бухгалтер	200	5	1000
Итого			129400

Величина денежных вложений, связанных с оплатой труда специалистов, привлеченных к разработке приложения для компании «Квартплата 24», составили 129,4 тыс. руб. С учетом отчислений страховых взносов, тариф которых составляет 30,2%, величина затрат принимает значение:

$$Z = 128.4 * 1.302 = 168.2 \text{ тыс. руб.} \quad (1)$$

«Расчет затрат, связанных с использование компьютерного оборудование, необходимого при разработке приложения для компании «Квартплата 24», производится через расчёт доли времени разработки системы относительно срока амортизации, составляющего для данной категории оборудования, а именно сервера для хостинга ИС, 5 лет»[13].

$$S_A = \frac{2000}{60} = 3333 \text{ тыс. руб. в месяц} \quad (2)$$

«Затраты, связанные с оплатой тарифа электроэнергии, составляющего в 6 руб./кВт*ч с учетом 180 часов машинного времени, 0,7 кВт полезной мощности используемого оборудования, составят: $S_E = 6 * 180 * 0.7 = 756$ руб.

Таким образом, экономический эффект от внедрения системы лояльности в качестве проекта по привлечению клиентов, по предварительным расчетам эффект за первый год составит 301,2 тыс. руб.

Срок окупаемости приложения для компании «Квартплата 24» составит» [13]:

$$T_{OK} = \frac{202886}{301200} * 12 = 8.1 \text{ мес.} \quad (3)$$

Окупаемость разработки приложения составила 8.1 месяцев.

Вывод по главе 3

В результате рассмотрения третьей главы можно сделать вывод, что на этапе разработки мобильного приложения был осуществлен тщательный выбор технологий и программных средств, обеспечивший эффективную реализацию функционала. Использование Jetpack Compose для создания пользовательского интерфейса, базы данных MySQL и языка Kotlin для разработки серверной и клиентской частей позволило реализовать ключевые компоненты приложения, включая авторизацию, управление профилем, внесение показаний счетчиков, оплату услуг ЖКУ, расчет и использование кешбэка, а также добавление новых услуг [18].

Проведённое тестирование подтвердило корректную работу всех модулей, соответствие требованиям по производительности и функциональности, что свидетельствует о достижении поставленных целей разработки.

Заключение

Было разработано мобильное приложение с подсистемой лояльности для расчета кешбека на основе банковских транзакций для предоставления скидки для оплаты ЖКУ. Основное внимание уделено созданию удобного и функционального инструмента, хорошего интуитивно понятного графического дизайна.

В ходе работы были выполнены следующие задачи:

- проведен анализ предметной области и изучены существующие методы расчета кешбека и его работы. Это позволило определить ключевые требования к функционалу системы, а также необходимые нюансы при реализации.
- разработана структура базы данных на основе MySQL, обеспечивающая хранение информации.
- разработан интерфейс мобильного-приложения, включающий формы для ввода данных, страницы с предоставлением информации о платежах, а также кешбеке и о его получении с транзакций.
- проведено тестирование работы системы, в ходе которого были устранены ошибки и оптимизированы механизмы обработки данных и начисления кешбека.

В результате проделанной работы было создано мобильное приложение, которое автоматизирует и предоставляет кешбек для оплаты ЖКУ [1]. Благодаря использованию базы данных MySQL обеспечивается надежное хранение информации, а благодаря инструменту разработки в лице AndroidStudio, качественная работоспособность продукта.

Разработанная система соответствует поставленной задаче, как ИС для привлечения клиентов, реализованная в виде мобильного приложения с системой лояльности.

Все поставленные задачи были решены и достигнуты, в следствии чего, цель выпускной квалификационной работы так же выполнена.

Список используемой литературы и используемых источников

1. Государственная информационная система жилищно-коммунального хозяйства [Электронный ресурс]. – URL: Государственная информационная система жилищно-коммунального хозяйства. (Дата обращения: 25.11.2024).
2. Квартплата+ [Электронный ресурс]. – URL: Платежный кабинет Квартплата+ системы «Город» - любые оплаты онлайн банковской картой. (Дата обращения: 25.11.2024).
3. Квартплата24 [Электронный ресурс]. URL: https://www.kvp24.ru/about/ (дата обращения: 15.11.2024).
4. Лавров, С. Н. Моделирование экономических систем / С. Н. Лавров. – Москва: Экономика, 2024. – 351 с.
5. Логические и физические модели данных [Электронный ресурс]. – URL: https://aws.amazon.com/ru/compare/the-difference-between-logical-and-physical-data-model/ (дата обращения: 12.11.2024).
6. Майерс, Д. Социальная психология / Д. Майерс. – Санкт-Петербург: Питер, 2024. – 688 с.
7. Мескон, М. Х. Основы менеджмента / М. Х. Мескон, М. Альберт, Ф. Хедоури. – Москва: Вильямс, 2024. – 672 с.
8. Модель «Как есть» [Электронный ресурс]. – URL: https://piter-soft.ru/knowledge/glossary/process/as-is-model.html (дата обращения: 07.12.2024).
9. Особенности тестирования «Черного Ящика» [Электронный ресурс]. URL: https://quality-lab.ru/blog/key-principles-of-black-box-testing/ (дата обращения: 07.11.2024).
10. Платосфера [Электронный ресурс]. – URL: Принимайте платежи от клиентов в Платосфере. (Дата обращения: 25.11.2024).
11. Портер, М. Конкурентная стратегия: Методика анализа отраслей и конкурентов / М. Портер. – Москва: Альпина Паблишер, 2011. – 456 с.

12. Проектирование баз данных для автоматизированных систем. [Электронный ресурс]. – URL: https://db-design.ru/methods (дата обращения: 12.04.2025).

13. Райзберг, Б. А. Современный экономический словарь / Б. А. Райзберг, Л. Ш. Лозовский, Е. Б. Стародубцева. – 6-е изд., перераб. и доп. – Москва: ИНФРА-М, 2024. – 512 с.

14. Робинсон, С. Современное имитационное моделирование / С. Робинсон. – Москва: Перо, 2024. – 400 с.

15. Firebase Documentation. (n.d.). Firebase Authentication [Электронный ресурс]. URL: https://firebase.google.com/docs/auth (дата обращения: 15.11.2024).

16. Google. (n.d.). Android Studio [Электронный ресурс]. URL: https://developer.android.com/studio (дата обращения: 15.11.2024).

17. Gradle Build Tool. (n.d.). Documentation [Электронный ресурс]. URL: https://gradle.org/documentation/ (дата обращения: 15.11.2024).

18. Jetpack Compose. (n.d.). Documentation [Электронный ресурс]. URL: https://developer.android.com/jetpack/compose (дата обращения: 15.11.2024).

19. Kotlin Programming Language. (n.d.). [Электронный ресурс]. URL: https://kotlinlang.org/ (дата обращения: 15.11.2024).

20. Ktor Framework. (n.d.). [Электронный ресурс]. URL: https://ktor.io/ (дата обращения: 15.11.2024).

21. Mysql (n.d.). [Электронный ресурс]. URL: https://metanit.com/sql/mysql/ (дата обращения: 15.11.2024).

22. Retrofit Library. (n.d.). [Электронный ресурс]. URL: https://square.github.io/retrofit/ (дата обращения: 15.11.2024).

Приложение А

Зависимости в файле build.gradle.kts (Module App)

```
dependencies {
    // Jetpack Compose
    implementation("androidx.compose.ui:ui:1.4.3")
    implementation("androidx.compose.material:material:1.4.3")
    implementation("androidx.compose.ui:ui-tooling-preview:1.4.3")

    // Retrofit + Gson
    implementation("com.squareup.retrofit2:retrofit:2.9.0")
    implementation("com.squareup.retrofit2:converter-gson:2.9.0")

    // Room (база данных)
    implementation("androidx.room:room-runtime:2.5.1")
    kapt("androidx.room:room-compiler:2.5.1")

    // SQLite
    implementation("androidx.sqlite:sqlite:2.3.0")

    // Firebase
    implementation(platform("com.google.firebase:firebase-bom:32.2.0"))
    implementation("com.google.firebase:firebase-auth-ktx")
    implementation("com.google.firebase:firebase-firestore-ktx")
    implementation("com.google.firebase:firebase-database-ktx")

    // Логирование
    implementation("com.jakewharton.timber:timber:5.0.1")
}
```

Рисунок А.1 – Зависимости в файле build.gradle.kts (Module App)

Приложение Б

Код файла TransactionApi.kt

```kotlin
package com.example.kvp24.api

import retrofit2.http.Body
import retrofit2.http.GET
import retrofit2.http.POST
import retrofit2.http.Path

data class TransactionRequest(val userId: Int, val amount: Double, val date: String, val mccCode: String)
data class TransactionResponse(val message: String, val cashback: Double)

interface TransactionApi {
    @GET("transactions/{userId}")
    suspend fun getTransactions(@Path("userId") userId: Int): List<TransactionRequest>

    @POST("transactions")
    suspend fun addTransaction(@Body transaction: TransactionRequest): TransactionResponse
}
```

Рисунок Б.1 – Код файла TransactionApi.kt

Приложение В

Код файла RetrofitClient.kt

```
package com.example.kvp24.api

import retrofit2.Retrofit
import retrofit2.converter.gson.GsonConverterFactory

object RetrofitClient {
    private const val BASE_URL = "https://api_of_transaction"

    val api: TransactionApi by lazy {
        Retrofit.Builder()
            .baseUrl(BASE_URL)
            .addConverterFactory(GsonConverterFactory.create())
            .build()
            .create(TransactionApi::class.java)
    }
}
```

Рисунок В.1 – Код файла RetrofitClient.kt

Приложение Г

Код файла CashbackRules.kt

```kotlin
package cashback

object CashbackRules {
    val MCC_CATEGORIES = mapOf(
        "5411" to 0.02,  // Продукты - 2%
        "5812" to 0.05,  // Рестораны - 5%
        "5541" to 0.03,  // АЗС - 3%
        "4900" to 0.01   // ЖКУ - 1%
    )

    fun getCashbackPercent(mccCode: String): Double {
        return MCC_CATEGORIES[mccCode] ?: 0.005 // По умолчанию 0.5%
    }
}
```

Рисунок Г.1 – Код файла CashbackRules.kt

Приложение Д

Метод CashbackCalculator.kt

```kotlin
package cashback

import models.Transaction

class CashbackCalculator {
    fun calculate(transaction: Transaction): Double {
        val percent = CashbackRules.getCashbackPercent(transaction.mccCode)
        return transaction.amount * percent
    }
}
```

Рисунок Д.1 – Метод CashbackCalculator.kt

Приложение Е

Код файла TransactionRoutes.kt

```kotlin
package services

import cashback.CashbackCalculator
import database.entities.Transactions
import models.Transaction
import org.jetbrains.exposed.sql.*
import org.jetbrains.exposed.sql.transactions.transaction

class TransactionService {
    private val cashbackCalculator = CashbackCalculator()

    fun getUserTransactions(userId: Int): List<Transaction> = transaction {
        Transactions.select { Transactions.userId eq userId }
            .map {
                Transaction(
                    id = it[Transactions.id],
                    userId = it[Transactions.userId],
                    amount = it[Transactions.amount].toDouble(),
                    date = it[Transactions.date].toString(),
                    mccCode = it[Transactions.mccCode]
                )
            }
    }

    fun addTransaction(transaction: Transaction): Double = transaction {
        Transactions.insert {
            it[userId] = transaction.userId
            it[amount] = transaction.amount.toBigDecimal()
            it[date] = org.joda.time.DateTime.parse(transaction.date)
            it[mccCode] = transaction.mccCode
        }

        // Расчёт кешбека
        val cashbackAmount = cashbackCalculator.calculate(transaction)
        cashbackAmount
    }
}
```

Рисунок Е.1 – Код файла TransactionRoutes.kt