

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет
вычислительной техники

Кафедра
МОиПЭВМ

Направление подготовки 09.03.02 «Информационные системы и технологии»

БАКАЛАВРСКАЯ РАБОТА

на тему

Информационная система ЖКХ на
основе витрины данных

Студент _____ Мельхов Антон Алексеевич
(подпись, дата) (ФИО полностью)

Руководитель _____ Макарычев П.П.
(подпись, дата) (фамилия, инициалы)

Нормоконтролёр _____ Такташкин Д.В.
(подпись, дата) (фамилия, инициалы)

Работа допущена к защите (протокол заседания кафедры от _____ № _____)

Заведующий кафедрой _____ Козлов А.Ю.
(подпись) (фамилия, инициалы)

Работа защищена с отметкой _____ (протокол заседания ГЭК от _____ № _____)

Секретарь ГЭК _____ Попова Н.А.
(подпись) (фамилия, инициалы)

Пенза, 2023

Федеральное государственное бюджетное образовательное учреждение
высшего образования
ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет

Вычислительной техники

Кафедра

МО и ПЭВМ

«Утверждаю»
Заведующий кафедрой
_____ А.Ю. Козлов
«__» _____ 2023 г

**ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
БАКАЛАВРА**

1. Студент Мельхов Антон Алексеевич гр. 19ВИ1 факультета ВТ
<Фамилия, имя, отчество полностью>

2. Тема работы Информационная система ЖКХ на основе витрины данных

Тема утверждена приказом ПГУ № _____ от "____" _____ 20____

2. Руководитель работы Профессор МОиПЭВМ, доктор т.н. Макарычев П.П., ПГУ

3. Задание на работу (назначение разработки, исходные данные и т.п.)

Назначение: информационная система ЖКХ на основе витрины данных

Основные функции информационной системы:

а) хранение информации о пользователях, сотрудниках, значениях показателей

б) предоставление печатной формы квитанции

Технология разработки: RUP, витрина данных

Инструментальные средства: Visual Studio Code, Apache Jmeter, Dbeaver, Docker

Среда исполнения: операционная система Centos 7.8

4. Перечень подлежащих разработке
вопросов _____

- 1) Анализ предметной области
- 2) Анализ функциональных требований
- 3) Проектирование
- 4) Реализация
- 5) Контроль качества информационной системы: тестирование производительности

5. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1	Анализ предметной области и требований к системе	10%	12.04.2023 – 16.04.2023	
2	Проектирование программных средств	20%	17.04.2023 – 26.04.2023	
3	Реализация программных средств	30%	27.04.2023 – 11.05.2023	
4	Тестирование программных средств	20%	12.05.2023 – 21.05.2023	
5	Оформление пояснительной записки	20%	22.05.2023 – 01.06.2023	

Дата выдачи задания «22» декабря 2022г.

Руководитель бакалаврской работы _____
(подпись, дата)

Задание к исполнению принял студент _____
(подпись, дата)

Работу к защите допустить
Декан ФВТ _____ Л.Р. Фионова
(подпись, дата)

Реферат

Пояснительная записка содержит 50 листов, 16 таблиц, 18 рисунков, 8 листингов, 22 использованных источников.

ЖКХ, ВИТРИНА ДАННЫХ, PROSTORE, POSTGRES, DOCKER, APACHE KAFKA, APACHE JMETER, DBEAVER, JDBC DRIVER.

Объект разработки: Информационная система жилищно-коммунального хозяйства.

Цель работы: Разработать информационную систему жилищно-коммунального хозяйства на основе витрины данных.

В процессе работы был произведен анализ требований и условий функционирования разрабатываемых программных средств, определены структура ПС, технология их разработки и тестирования.

Разработана информационная система с использованием технологии витрины данных, которая реализована с помощью компонента обработки запросов и анализа данных Prostore. Использование информационной системы обеспечивает удобное хранение развернутой информации о пользователях, организациях, услугах и предоставление пользователю печатной формы квитанции.

					ПГУ 09.03.02 – 8БР191.14 ПЗ			
Изм.	Лист	№ докум.	Подпись	Дата	Информационная система ЖКХ на основе витрины данных. Пояснительная записка.	Лит.	Лист	Листов
Разраб.		Мельхов А.А.						
Провер.		Макарычев П.П.					4	50
Реценз.						группа 19ВИ1		
Н. Контр.		Такташкин Д.В.						
Утверд.		Козлов А. Ю.						

Содержание

Введение	6
1. Анализ предметной области разработки	7
1.1 Общие положения жилищного коммунального хозяйства	7
1.2 Обзор аналогичных информационных систем	9
1.3 Концептуальная модель предметной области.....	11
1.4 Постановка задачи на проектирование информационной системы	12
2 Анализ требований к программному обеспечению	14
3 Анализ информационных технологий разработки программного обеспечения.....	18
3.1 Методология проектирования информационной системы.....	18
3.2 Средства проектирования и разработки системы.....	18
3.3 Технология витрин данных.....	19
3.4 Анализ языка шаблонизатора Pebble Template	20
4 Проектирование программного обеспечения	22
4.1 Проектирование архитектуры витрины данных.....	22
4.2 Проектирование базы данных в витрине данных.....	23
4.3 Проектирование печатной формы квитанции.....	28
5 Реализация информационной системы.....	29
5.1 Реализация витрины данных.....	29
5.2 Наполнение витрины тестовыми данными	33
5.3 Реализация печатной формы.....	36
5.4. Тестирование производительности	38
6 Планирование и расчёт бюджета разрабатываемой информационной системы.....	42
Заключение	47
Список использованных источников	48

Введение

Жилищно-коммунальное хозяйство – это одна из важнейших сфер жизнедеятельности людей, связанная с обеспечением жилья и комфортного проживания в нем. Для эффективной работы данной отрасли необходимы управление, контроль и анализ информации о состоянии и использовании жилищного фонда, коммунальных услугах и их оплате.

В выпускной квалификационной работе рассматривается разработка информационной системы жилищно-коммунального хозяйства в витрине данных.

В ходе выполнения квалификационной работы были решены следующие задачи:

- выполнен анализ предметной области;
- определены функциональные возможности системы;
- выполнено проектирование структуры базы данных для хранения сведений о пользователях, обращениях и значениях измерительных приборов;
- разработана печатная форма квитанции;
- разработаны sql запросы к витрине данных;
- составлены требования к аппаратным ресурсам для размещения витрины данных;
- выполнено реализация информационной системы;
- протестировано разработанное программное обеспечение.

Для хранения информации о пользователях, сотрудниках и измерениях было выбрано ядро типового программного обеспечения витрины данных НСУД.

1. Анализ предметной области разработки

1.1 Общие положения жилищного коммунального хозяйства

Жилищное законодательство — это совокупность признанных законодательных актов, которые были разработаны для регулирования жилищных отношений между гражданами, юридических лиц, а также субъектов Российской Федерации, муниципальных образований. Жилищные нормативно-правовые акты исходят из необходимости обеспечения органами государственной власти и органами местного самоуправления условий для реализации гражданами своего права на жилище, его безопасности, неприкосновенности и недопустимости произвольного лишения жилища.

Жилищное законодательство регулирует отношения по поводу:

- соответствия жилых помещений установленным санитарным и техническим правилам и нормам, контролю над использованием и сохранностью жилищного фонда и иным требованиям законодательства;
- управления многоквартирными домами;
- внесения платы за жилое помещение и коммунальные услуги, предоставления коммунальных услуг;
- возникновения, изменения, прекращения права владения, распоряжения жилыми помещениями государственного и муниципального жилищных фондов;
- пользования жилыми помещениями частного жилищного фонда, общим имуществом собственников помещений.

Современные нормативно-правовые акты, которые регулируют деятельность жилищно-коммунального хозяйства, состоит из нескольких сотен законодательных актов, которые были приняты на федеральном уровне, а также нескольких тысяч нормативных актов принятых на уровне регионов России [1].

Принятыми нормативными правовыми актами регламентируются:

- техническое обслуживание и ремонтные работы, прочие коммунальные услуги;
- электроснабжение, газоснабжение, отопление, вывоз мусора, обслуживание прилегающих территорий, водоснабжение и водоотведение;
- выделение денежных средств на коммунальные хозяйства;
- общие вопросы жилищно-коммунального хозяйства;
- льготы и компенсации (дотации) по оплате коммунальных услуг.

Можно выделить следующие основные законодательные акты:

- Федеральный закон № 184-ФЗ «О техническом регулировании»;
- Федеральный закон № 190-ФЗ «О теплоснабжении»;
- Постановление Правительства РФ «О порядке осуществления деятельности по управлению многоквартирным домом»;
- Федеральный закон № 210-ФЗ «Об основах регулирования тарифов организаций коммунального комплекса»;
- Федеральный закон № 189-ФЗ «О введении в действие жилищного кодекса»;
- Федеральный закон № 416-ФЗ «О водоснабжении и водоотведении»;
- Постановление Правительства РФ № 406 «О государственном регулировании тарифов в сфере водоснабжения и водоотведения» [2].

Правительство законодательно закрепило право в формировании ценообразования платы на жилье и коммунальные услуги, но субъекты РФ вправе сами устанавливать размер оплаты жилья и коммунальных услуг, которые не могут быть ниже установленных федеральных стандартов.

Главная цель реформы жилищно-коммунального хозяйства – это обеспечение жилищно-коммунальными услугами населения с гарантией качества. Об этом ясно говорится в «Концепции реформы жилищно-коммунального хозяйства в Российской Федерации», одобренной Указом Президента Российской Федерации № 425.

Главным документом, регулирующим качество оказания жилищно-коммунальных услуг населению, является договор на управление многоквартирным домом, заключенный между жильцами дома и управляющей компанией. В договоре фиксируется объем, качество и порядок учета потребляемых услуг.

Управляющая компания или товарищество собственников жилья в свою очередь заключают договоры на предоставление коммунальных услуг с ресурсоснабжающими организациями.

1.2 Обзор аналогичных информационных систем

Информационные системы окружают нас повсеместно, не обошло это стороной и сферу жилищного коммунального хозяйства.

Основным аналогом является Государственная информационная система жилищно-коммунального хозяйства, которая соответствует документу «О государственной информационной системе жилищно-коммунального хозяйства» от 21.07.2014 N-209 ФЗ [3].

На рисунке 1 представлена информация о наполненности данными информационной системы на 20.06.2023.

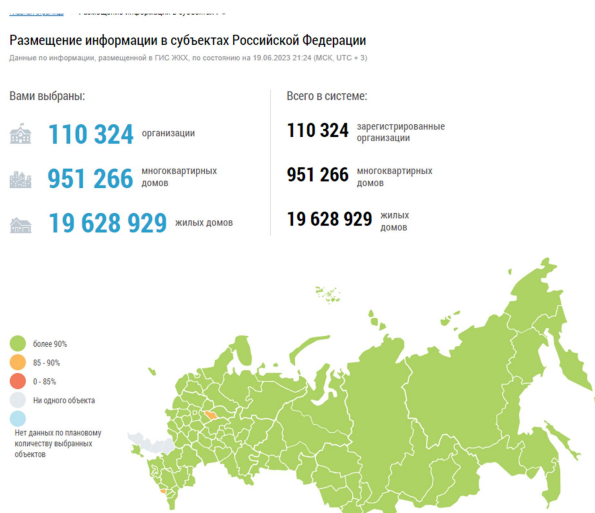


Рисунок 1 – Размещение информации в субъектах РФ

На рисунке 2 представлена информация о возможностях граждан данного аналога.



Рисунок 2 – Возможности граждан

Зачастую застройщики жилищных комплексов предоставляют свои продукты в области обработки обращений, квитанций и показаний счётчиков, например в ПИК комфорт (рисунок 3) [4].

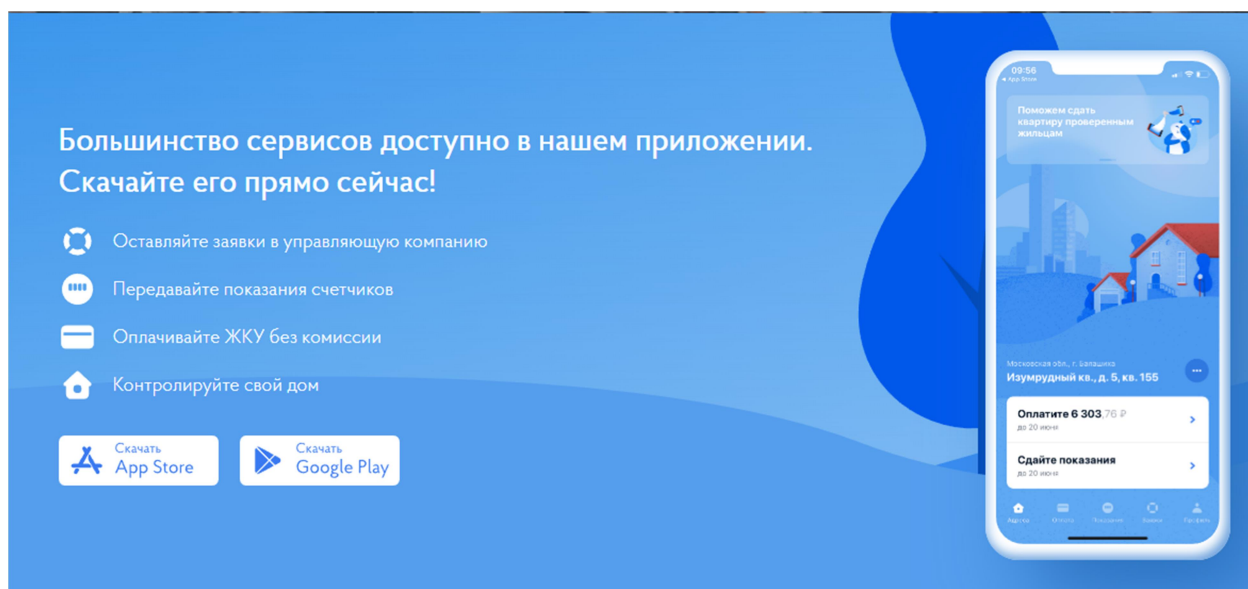


Рисунок 3 – Функционал приложения в ПИК комфорт

Схожий функционал имеет мобильное приложение «Термодом-сервис», представленный на рисунке 4 [5].

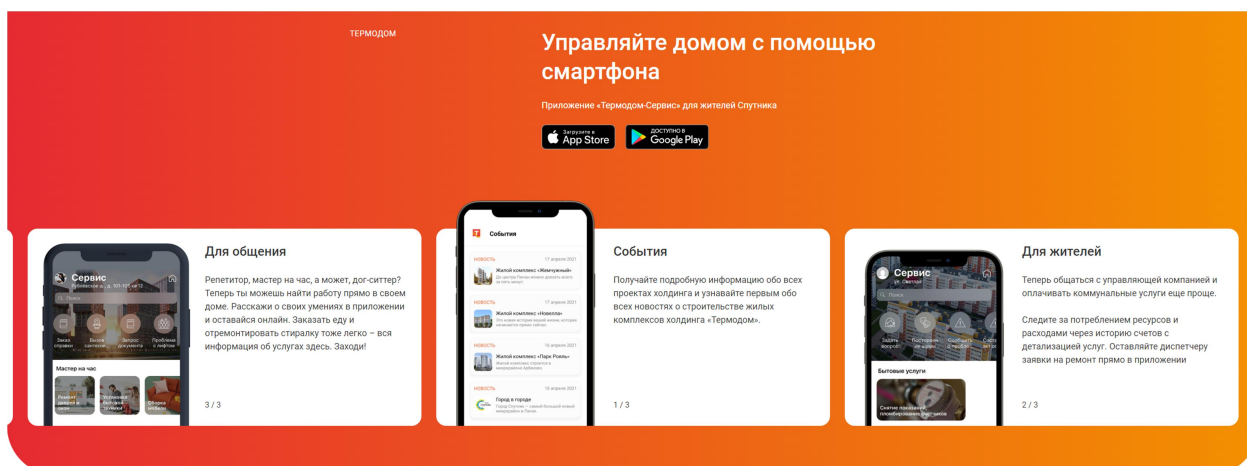


Рисунок 4 – Мобильное приложение «Термомод-сервис»

В рассмотренных аналогах не приведена информация о реализации систем с использованием витрин данных.

1.3 Концептуальная модель предметной области

Предметной областью рассматриваемой задачи является обеспечение информацией граждан.

Для регистрации пользователей необходимо предоставить документы о том, что он проживает по данному адресу.

Для организаций, которые обслуживают помещения, необходимо передать сведения о сотрудниках.

Информационная система должна хранить и обрабатывать данные о:

- пользователях;
- сотрудниках;
- организациях;
- значениях показателей измерительных приборов;
- пожеланиях и обращениях граждан.

На основе вышеизложенного модель предметной области может быть представлена с помощью следующей диаграммы классов (рисунок 5).

Описание сущностей предметной области приведено в таблице 1.

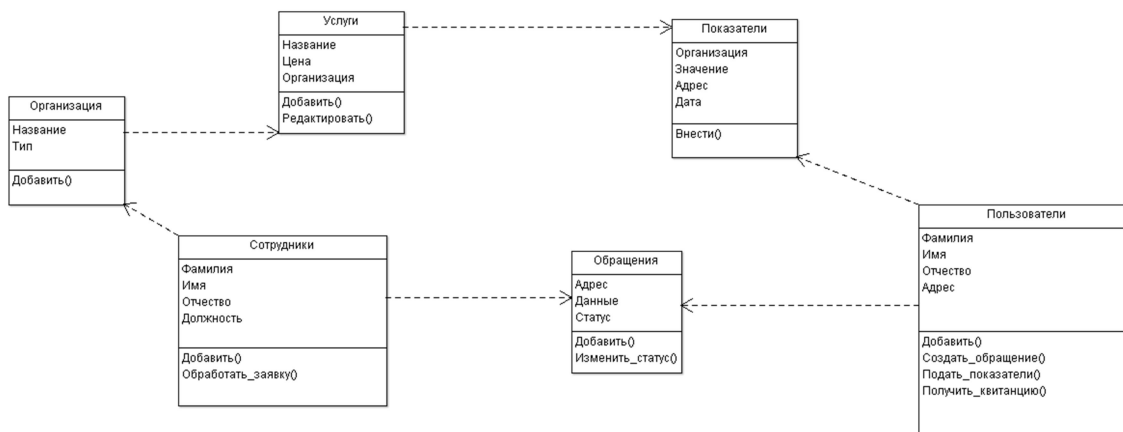


Рисунок 5 – Концептуальная модель предметной области

Таблица 1 – Описание сущностей предметной области

№ п/п	Наименование	Описание сущности	Число экземпляров
1	Услуги	Информация об услуге оказываемой организацией	50
2	Показатели	Значение измерительных приборов	500
3	Обращения	Обращение от пользователя	50
4	Сотрудники	Информация о сотруднике	160
5	Организация	Информация об организации	10
6	Пользователи	Данные пользователя	150

В данном подразделе была построена модель предметной области, позволяющая более точно и полно отразить наиболее существенные для решения поставленной задачи сущности, их свойства и взаимосвязи.

1.4 Постановка задачи на проектирование информационной системы

В данной выпускной квалификационной работе решается задача разработки информационной системы ЖКХ на основе витрины данных. Программное обеспечение должно выполнять размещение, хранение, изменение данных. Предоставлять сотрудникам сведения об обращениях пользователей, чтобы дать возможность обрабатывать их.

Информационная система должна быть реализована с использованием витрины данных, так как предполагает долгосрочную работу, большие объёмы данных, высокую производительность.

2 Анализ требований к программному обеспечению

Функциональные требования к системе. Функциональные требования к системе определяют действия, операции информационной системы, которые она должна выполнять [6]. В соответствии с предъявленными требованиями разрабатываемая система должна предоставлять функционал для пользователей и сотрудников организаций. На основании проведенного анализа предметной области была построена диаграмма вариантов использования информационной системы ЖКХ (рисунок 6).

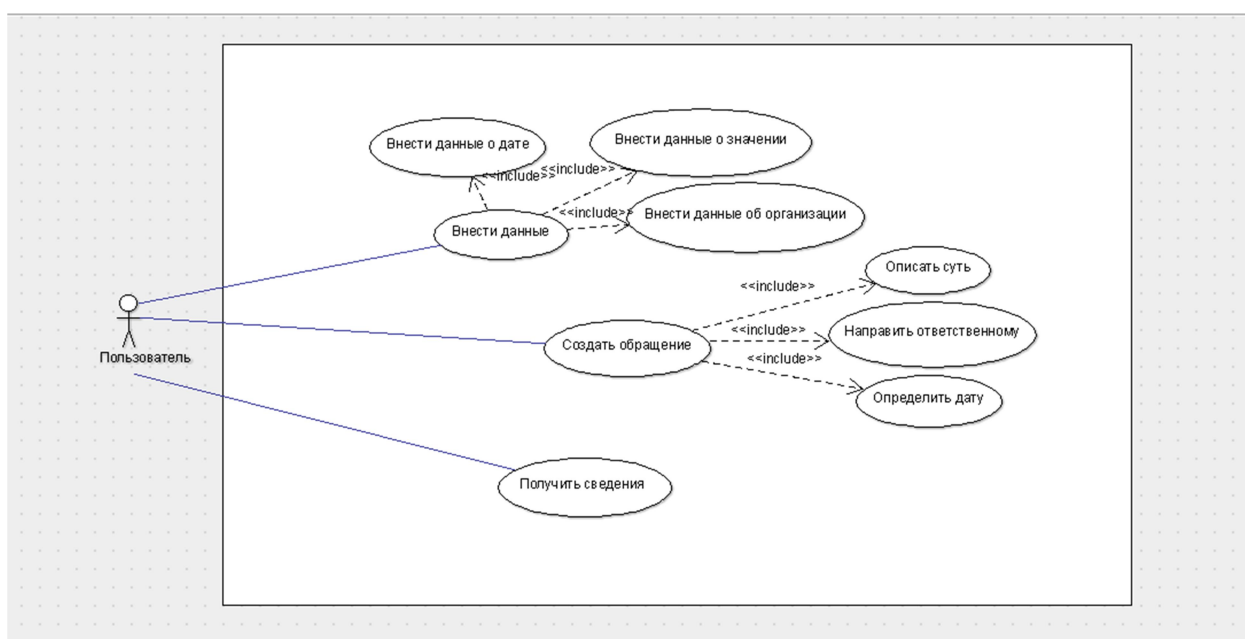


Рисунок 6 – Диаграмма вариантов использования

Основными вариантами использования информационной системы выделены внести данные, создать обращение, получить сведения. Детальное описание основных прецедентов диаграммы (таблицы 2 – 4).

Таблица 2 – Спецификация прецедента «Внести данные»

Имя прецедента	Внести данные
Идентификатор прецедента	ID1
Краткое описание	Пользователь предоставляет данные о пользовании услугами

Продолжение таблицы 2

Актёры, вовлечённые в прецедент (главные)	Пользователь
Состояние системы до прецедента	Отсутствуют данные
Фактические этапы прецедента	<ol style="list-style-type: none"> 1. Ввести дату 2. Ввести значение 3. Выбор организации, по которой предоставляются сведения
Состояние системы после прецедента	Данные доступны в витрине
Альтернативные потоки	Нет

Таблица 3 – Спецификация прецедента «Создать обращение»

Имя прецедента	Создать обращение
Идентификатор прецедента	ID2
Краткое описание	Уведомление ЖКХ о имеющихся проблемах
Актёры, вовлечённые в прецедент (главные)	Пользователь
Состояние системы до прецедента	Отсутствие обращения
Фактические этапы прецедента	<ol style="list-style-type: none"> 1. Пользователь описывает ситуацию 2. Пользователь указывает дату
Состояние системы после прецедента	Обращение представлено в витрине
Альтернативные потоки	Нет

Таблица 4 – Спецификация прецедента «Получить сведения»

Имя прецедента	Получить сведения
Идентификатор прецедента	ID3
Краткое описание	Получение квитанции
Актёры, вовлечённые в прецедент (главные)	Пользователь
Состояние системы до прецедента	Отсутствие квитанции
Фактические этапы прецедента	Система формирует квитанцию для пользователя
Состояние системы после прецедента	Сформирована квитанция
Альтернативные потоки	Нет

Таким образом, в ходе анализа функциональных требований к системе была построена диаграмма вариантов использования и спецификация основных прецедентов.

Требования к производительности информационной системы. Чтобы сформировать требования к информационной системе, были проанализированы обращения пользователей, согласно которым в среднем в день поступает 7 обращений [9]. Для оказания положительного впечатления на пользователя, информационная система должна реагировать на запросы не более 1 секунды, а также поддерживать работоспособность под 30-ю обращениями в секунду.

Требования к надёжности информационной системы. Свойство системы сохранять во времени в установленных пределах значения всех параметров, характеризующих способность выполнять требуемые функции в заданных режимах и условиях эксплуатации, является одним из основных критериев качества. Чем качественнее система, тем она надежнее, и наоборот [10].

Требования к надежности разрабатываемой информационной системы можно сформулировать следующим образом:

- ошибки в представленных данных не должны приводить к критическим нарушениям работы системы или потере работоспособности;
- система должна обеспечить механизм резервного копирования данных;
- система должна контролировать целостность и непротиворечивость информации в базе данных;
- система должна иметь возможность предоставлять данные из архива.

Требования к аппаратно-программным средствам. Основные требования к аппаратным средствам, которые необходимо учитывать при разработке, тестировании и функционировании создаваемого программного продукта: быстродействие, стабильность и надежность.

В качестве операционной системы необходимо использовать дистрибутивы Linux, для возможности использования всех предоставляемых ресурсов и уменьшения себестоимости системы.

В качестве базы данных необходимо использовать Postgres, так как является открытым решением и имеет большое количество пользователей, что позволит быстро решить возникающие проблемы.

3 Анализ информационных технологий разработки программного обеспечения

3.1 Методология проектирования информационной системы

Разработка автоматизированной информационной системы страхового агента будет выполняться в соответствии с методологией Rational Unified Process (RUP). Эта методология RUP опирается на методы анализа, проектирования и разработки программного обеспечения, методы управления проектами [11 – 13].

RUP обеспечивает прозрачность и управляемость процесса, объектно-ориентированную технологию. Моделирование базируется на понятиях объектов, классов и зависимостей между ними. Эти модели в качестве единого стандарта для организации взаимодействия участников проекта используют Unified Modelling Language (UML) – универсальный язык моделирования [14]. Диаграммы UML повышают качество документации проекта и облегчают разработку документации к программной системе. Язык UML может быть применен на всех этапах жизненного цикла анализа бизнес-систем и разработки приложений и создает унифицированное, понятное всем разработчикам описание программных средств на каждом этапе.

3.2 Средства проектирования и разработки системы

В качестве средств проектирования [15] информационных систем следует отметить целый класс программного обеспечения, который работает с языком моделирования UML. К данному классу сред проектирования можно отнести NClass, MS Visio, Altova UModel, Magic Draw, Sparx Enterprise Architect, Visual Paradigm, Sybase Power Designer, семейство Rational, Violet UML Editor, UMLet, StarUML, CADE и другие.

В качестве организации подключения к витрине данных использовался DBeaver Community edition, который позволяет работать с JDBC драйверами.

Для построения диаграммы IDEF1X был задействован Microsoft Visio.

3.3 Технология витрин данных

Если коротко, то витрины (витрина от англ. data mart) – это набор структурированных данных. Обычно это данные по определенной теме или задаче в компании. Например, витрина с данными о заказчиках для отдела маркетинга может содержать подробные данные по договорам, истории заказов и поставок, оплатах, звонках и адресах доставки. Ничего лишнего, только нужные и актуальные очищенные данные, полученные из других ИС предприятия. Таких витрин даже на одном предприятии может быть множество.

Требования, предъявляемые к витринам данных, можно отобразить следующим образом:

Изоляция данных. Обновление данных, например загрузка справочника, может быть растянуто во времени, при этом до окончания загрузки текущая версия справочника должна быть полностью доступна с исключением «грязного чтения» загружаемой версии.

Гарантии атомарности операций при обновлении данных. В случае сбоев и ошибок загрузки данных витрина остаётся в состоянии, которое предшествовало сбойному процессу. Другими словами, или данные обновляются полностью, или не обновляются вовсе, не оставляя следов сбойных операций.

1. Устойчивость к дубликатам изменений. Весьма сложно и дорого реализовывать во всех ИС-источниках данных выгрузку по принципу exactly-once. Наличие дублей одинаковых изменений объектов не должно приводить к нарушению логической целостности состояния витрин.

2. Системная темпоральность. Мало какая реляционная СУБД имеет функцию системной темпоральности «из коробки». Ведение системного времени и версионирование записей по системному времени позволяет сравнивать состояние данных витрины между двумя разными моментами времени или проводить «расследование», основываясь на данных,

которые были в витрине в определенный момент в прошлом. Одним из вариантов обеспечения темпоральности является реализация SCD2 с ведением диапазонов сроков действия для версий записи.

3. Эффективное выполнение различных видов запросов: сравнительно редких и тяжелых аналитических запросов, затрагивающих большой объем данных (OLAP-нагрузка), и множества одновременных простых запросов (OLTP-нагрузка). Как правило, СУБД заточены на какой-то один вариант нагрузки: OLAP или OLTP.

Проанализировав предоставленную информацию, была выбрана отечественная разработка с открытым исходным кодом [16]. Сыграло ещё огромную роль в выборе данного продукта то, что он сопровождается документацией на русском языке [17].

3.4 Анализ языка шаблонизатора Pebble Template

Веб-приложения формируют HTML-документы для показа пользователям. Данные документы динамически формируются на стороне сервера. Некоторые части этих документов не изменяется: навигационное меню, заголовки, подключение внешних файлов и т.д. Другая часть обычно содержит повторяющиеся элементы: списки элементов, таблицы и т.д. Эти данные обычно зависят от тех данных, которыми управляет приложение.

Для того, чтобы выполнять шаг генерации HTML-документов было удобнее, на стороне сервера применяются специальные инструменты — шаблонизаторы. В задачу любого шаблонизатора входит формирование строки на основании шаблона и конфигурации. Конкретные инструменты отличаются друг от друга возможностями и способом описания самих шаблонов, их синтаксисом.

Шаблоны обычно представляют собой обычные файлы на жёстком диске, содержащие текст на целевом языке, а также управляющие конструкции на языке шаблона [18].

Шаблонизаторы предоставляют средства для обработки поступающих данных:

- вставка данных в шаблон;
- обход списков;
- выполнение условной логики в зависимости от данных;
- фильтрация данных.

Множество приложений используют данный шаблонизатор благодаря его возможностям для разработки собственных расширений. Например, на основе этого в сервисе `printable-form-service` реализован механизм направления и выполнения `sql` выражения в Prostore.

4 Проектирование программного обеспечения

4.1 Проектирование архитектуры витрины данных

Проектирование архитектуры осуществляется согласно минимальным требованиям, предъявляемым к витрине данных, изложенными в документации.

А именно, для запуска витрины требуется: 4 логических ядра процессора, 16 GB оперативной памяти и 20 GB памяти на жёстком диске.

Минимальный набор компонентов для запуска:

- Prostore (ядро витрины) 1 экземпляр;
- Status-monitor (сервис мониторинга статусов Kafka) 1 экземпляр;
- Kafka и Zookeeper (брокер сообщений) 1 экземпляр;
- СУБД Postgres (хранилище данных) – 1 экземпляр;
- Kafka-postgres-writer (компонент для загрузки данных в витрину) 1 экземпляр;
- Printable-form-service (компонент для печатной формы) – 1 экземпляр.

В качестве брокера сообщения используется Apache Kafka. Kafka Apache – распределенная система обмена сообщениями между серверными приложениями в режиме реального времени. Благодаря высокой пропускной способности, масштабируемости и надежности применяется в компаниях, работающих с большими объемами данных. Написана на языках Java и Scala.

Kafka разработана компанией LinkedIn. В 2011 году разработчик опубликовал исходный код системы. С тех пор платформа развивается и поддерживается как открытый проект в рамках фонда Apache Software Foundation [19].

Размещение компонентов и их взаимодействие представлены на диаграмме развёртывания (рисунок 7).

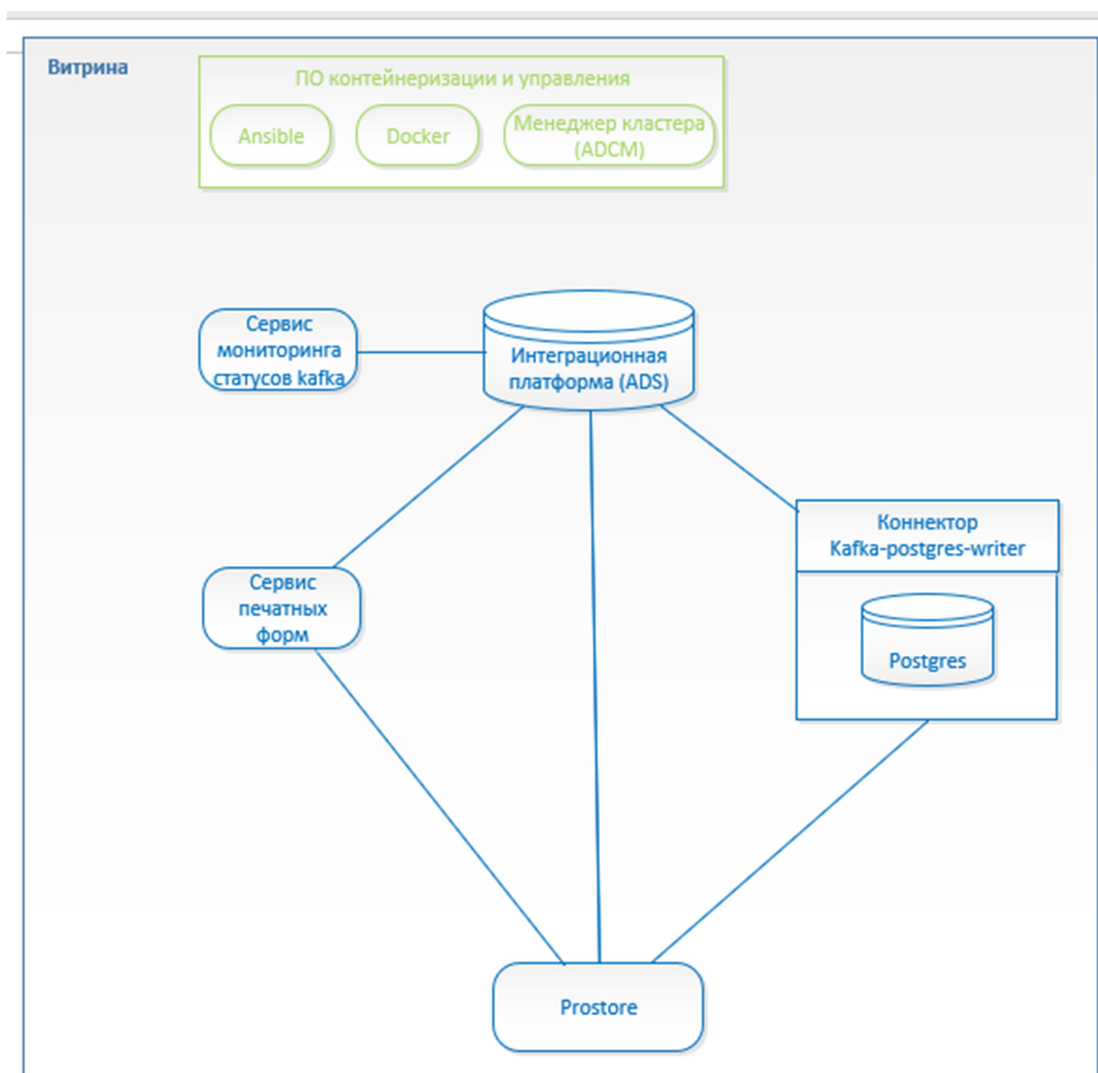


Рисунок 7 – Диаграмма развёртывания

ПО контейнеризации и управления задействовано для запуска компонентов витрины данных.

4.2 Проектирование базы данных в витрине данных

В создании и проектировании информационной системы одну из главных ролей играет проектирование базы данных.

Классическая методология проектирования баз данных представляет собой структурированный подход, включающий в себя специализированные процедуры, технические приемы, инструменты и документацию.

В одной из наиболее используемых методологий проектирования баз данных весь процесс разработки разделяется на три основные фазы:

концептуальное, логическое и физическое проектирование. В общем случае, при проектировании БД на концептуальном уровне мы определяем сущности, на логическом – отношения, а на физическом таблицы.

Концептуальное проектирование базы данных представляет собой процедуру конструирования информационной модели предметной области, не зависящей от каких-либо физических условий реализации [20].

Фаза концептуального проектирования базы данных начинается с создания концептуальной модели данных предметной области, полностью независимой от любых деталей реализации. Модель предметной области приведена на рисунке 6 пункта 1.3.

Фаза логического проектирования базы данных заключается в преобразовании концептуальной модели данных в логическую модель данных предприятия с учетом выбранного типа СУБД.

В процессе создания логической модели предметной области были определены основные типы сущностей. Анализ предметной области показал, что основными являются следующие сущности: «Услуга», «Показатель», «Пользователь», «Обращение», «Сотрудник», «Организация». Описание и особенности использования приведено в таблице 5.

Таблица 5 – Перечень сущностей предметной области

Идентификатор сущности	Описание	Особенности использования
Услуга	Информация о предоставляемой услуге	Стоимость оказания услуги может изменяться
Показатель	Показания значения прибора	Каждый месяц предоставляется информация о показателях
Пользователь	Пользователь ЖКХ	Информация о клиентах ЖКХ. По адресу обслуживания могут находиться несколько клиентов

Продолжение таблицы 5

Идентификатор сущности	Описание	Особенности использования
Обращение	Информация от пользователя	В процессе рассмотрения обращения, информация может дополняться
Сотрудник	Информация о сотруднике организации	У сотрудника может изменяться должность
Организация	Информация об организации	В организации могут работать несколько сотрудников

После выделения сущностей предметной области были определены типы связей между этими сущностями (таблица 6).

Таблица 6 – Типы связей между сущностями предметной области

Тип сущности	Тип связи	Тип сущности
Услуга	1:M	Показатель
Организация	1:M	Сотрудник
Сотрудник	1:M	Обращение
Пользователь	1:M	Показатель
Пользователь	1:M	Обращение

После того, как выявлены все сущности и связи между ними, необходимо определить атрибуты типов сущностей. Атрибут описывает некоторый аспект определенной сущности или связи.

Описание атрибутов типов сущностей для каждой из рассмотренных выше сущностей приведено в таблицах 7 – 11.

Таблица 7 – Атрибуты сущности «Услуга»

Имя атрибута	Формат атрибута	Роль атрибута
Номер услуги	Числовой	РК
Наименование услуги	Символьный	
Стоимость	Числовой	
Наименование измеряемой величины	Символьный	
Номер организации	Числовой	FK

Таблица 8 – Атрибуты сущности «Организация»

Имя атрибута	Формат атрибута	Роль атрибута
Номер организации	Числовой	РК
Тип организации	Символьный	
Наименование организации	Символьный	

Таблица 9 – Атрибуты сущности «Сотрудник»

Имя атрибута	Формат атрибута	Роль атрибута
Номер сотрудника	Числовой	РК
Фамилия	Символьный	
Имя	Символьный	
Отчество	Символьный	
Должность	Символьный	
Номер организации	Числовой	FK

Таблица 10 – Атрибуты сущности «Пользователь»

Имя атрибута	Формат атрибута	Роль атрибута
Номер пользователя	Числовой	РК
Фамилия	Символьный	
Имя	Символьный	
Отчество	Символьный	
Адрес	Символьный	

В сущности «Сотрудник», как и, в сущности «Пользователь», атрибут «ФИО» был разбит на «Фамилия», «Имя», «Отчество», что позволяет выполнить условие атомарности, соответствующее первой нормальной форме.

Таблица 11 – Атрибуты сущности «Показатель»

Имя атрибута	Формат атрибута	Роль атрибута
Номер показателя	Числовой	РК
Значение	Числовой	
Адрес	Символьный	
Дата	Дата	РК
Номер услуги	Числовой	FK

В таблице 12 представлены атрибуты сущности «Обращение».

Таблица 12 – Атрибуты сущности «Обращение»

Имя атрибута	Формат атрибута	Роль атрибута
Номер обращение	Числовой	PK
Содержание	Символьный	
Дата	Дата	
Номер пользователя	Числовой	FK
Номер сотрудника	Числовой	FK
Статус обращения	Символьный	

Фрагмент схемы базы данных в нотации IDEF1X, спроектированной в Microsoft Visio, приведён на рисунке 8.

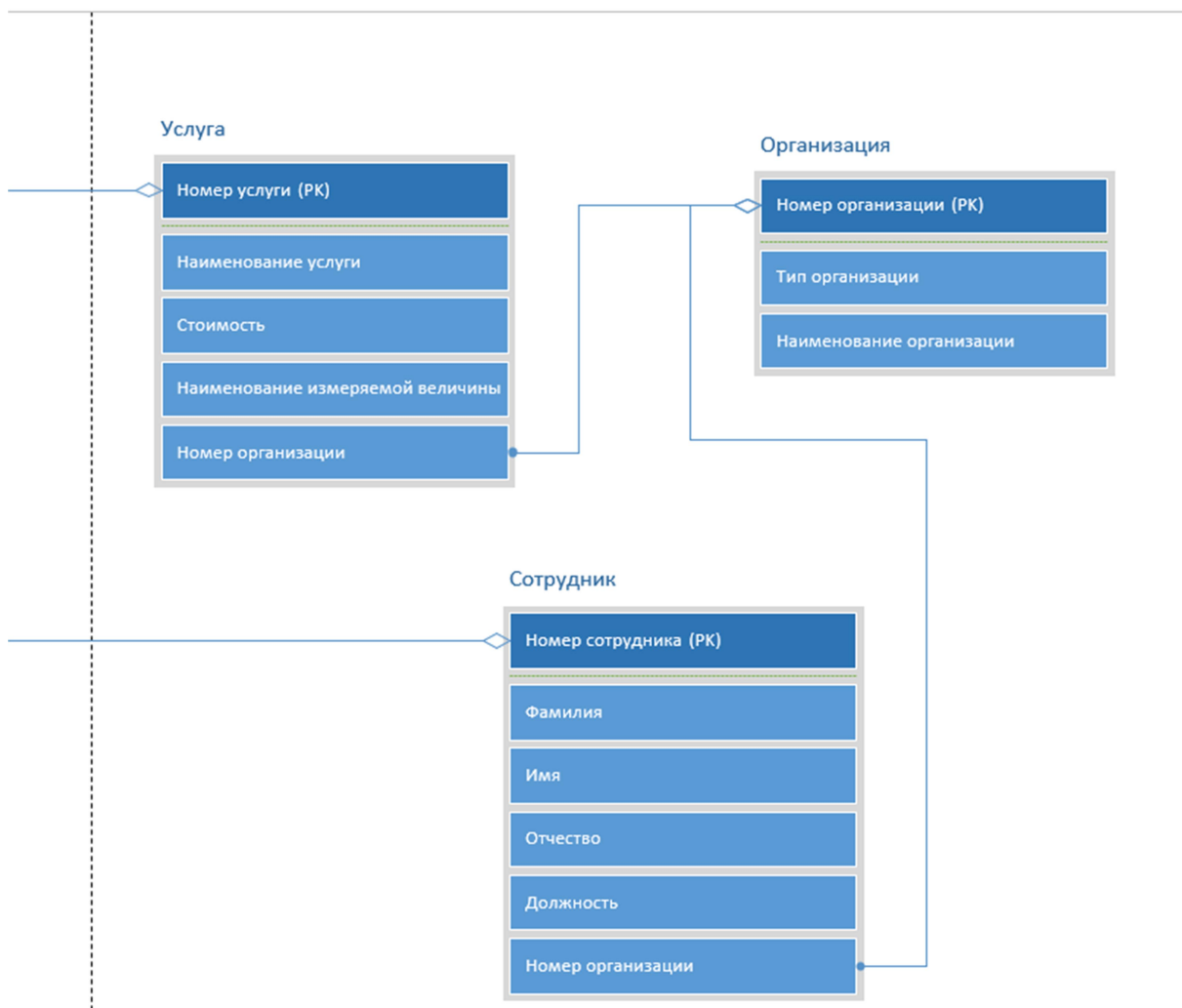


Рисунок 8 – Фрагмент схемы базы данных

Полученный результат упростит процесс создания таблиц в витрине данных.

4.3 Проектирование печатной формы квитанции

В печатной форме квитанции необходимо отобразить такие сведения, как: наименование организации, наименование предоставляемой услуги, показатели, адрес оказания услуги, а также фамилия, имя и отчество плательщика.

Основной способ предоставления информации в квитанции – pdf документ, в качестве альтернативы – xml документ.

5 Реализация информационной системы

5.1 Реализация витрины данных

Для запуска витрины необходимо наличие на сервере пакетов для поддержки контейнеризации. Процесс установки пакетов на ОС Centos 7 представлен на листинге 1.

Листинг 1 – Установка docker

```
sudo yum install -y yum-utils
sudo yum-config-manager --add-repo
https://download.docker.com/linux/centos/docker-ce.repo
sudo yum install docker-ce docker-ce-cli containerd.io docker-
buildx-plugin docker-compose-plugin
sudo systemctl start docker
```

После установки компонентов, необходимо запустить контейнеры, предварительно настроив их, согласно документации. Пример запуска контейнера представлен на листинге 2.

Листинг 2 – Запуск контейнера

```
sudo docker run --env-file /opt/prostore/.envfile prostore:5.8.0
```

Чтобы проверить корректность установки, необходимо создать подключение к Prostore в dbeaver. Dbeaver позволяет подключаться к хранилищам данным, СУБД хранилищам и другим. Зачастую данные подключения реализуются с использованием jdbc драйверов, которые идут в комплекте. Драйвер Prostore отсутствует, как штатный, но его можно добавить в ручном режиме.

Чтобы добавить jdbc драйвер и подключиться к витрине, выполним следующие шаги:

1. Перейдём на вкладку «База данных», затем «Управление драйверами» представлено на рисунке 9.

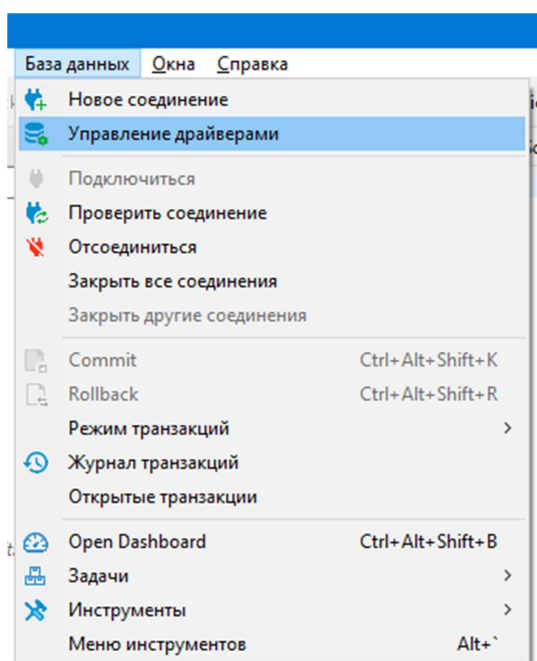
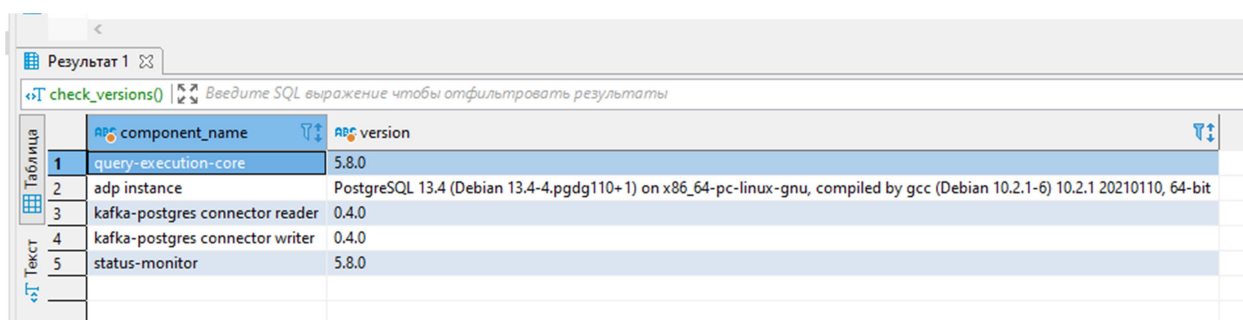


Рисунок 9 – Вкладка с управление драйверами

2. Открывается окно, в котором необходимо указать следующие данные:
3. Имя драйвера: Prostore
4. Имя класса: ru.datamart.prostore.jdbc.Driver
5. Шаблон url: jdbc:prostore://{host}:{port}
6. Переходим на вкладку «Библиотеки», добавляем файл драйвера.
7. После нажимаем «Ок», данными действиями добавили jdbc драйвер в dbeaver.
8. Перейдём к шагу создания соединения. Для этого добавим соединение, в качестве драйвера выбираем созданный с названием «Prostore».
9. Указываем хост для подключения, а также порт 9090 на котором слушается ядро витрины данных.
10. На созданном соединении проверяем работоспособность, выполнив sql выражение «CHECK_VERSIONS();». Пример результат выполнения представлен на рисунке 10.



	component_name	version
1	query-execution-core	5.8.0
2	adp instance	PostgreSQL 13.4 (Debian 13.4-4.pgdg110+1) on x86_64-pc-linux-gnu, compiled by gcc (Debian 10.2.1-6) 10.2.1 20210110, 64-bit
3	kafka-postgres connector reader	0.4.0
4	kafka-postgres connector writer	0.4.0
5	status-monitor	5.8.0

Рисунок 10 – Результат исполнения выражения

Данными шагами было создано соединение, а также проверено.

После проделанной работы необходимо создать витрину, а затем таблицы.

Витрина создаётся следующим sql выражением представленным в листинге 3.

Листинг 3 – Создание витрины

```
Create database jkh;
```

Затем требуется создание таблиц с помощью sql выражений представленных в листинге 4.

Листинг 4 – Создание таблиц

```
create table jkh.service (
    id_service int,
    name_of_service varchar,
    price int,
    name_of_value varchar,
    id_organization int,
    primary key (id_service)
) distributed by (id_service);

create table jkh.organization (
    id_organization int,
    type_organization varchar,
    name_of_organization varchar,
    primary key (id_organization)
) distributed by (id_organization);

create table jkh.worker (
    id_worker int,
```

```

        secondname varchar,
        firstname varchar,
        patronymic varchar,
        positions varchar,
        id_organization int,
        primary key (id_worker)
    ) distributed by (id_worker);

create table jkh.users (
    id_user int,
    secondname varchar,
    firstname varchar,
    patronymic varchar,
    adress varchar,
    primary key (id_user)
) distributed by (id_user);

create table jkh.index (
    id_index int,
    i_value int,
    adress varchar,
    d_date date,
    id_user int,
    id_service int,
    primary key (id_index)
) distributed by (id_index);

create table jkh.appeal (
    id_appeal int,
    text varchar,
    d_date date,
    id_user int,
    id_worker int,
    status_of_appeal varchar,
    primary key (id_appeal)
) distributed by (id_appeal);

```

После создания таблиц, в поле таблицы отображается созданная структура. (рисунок 11). Во время создания таблицы, внутри СУБД хранилища создаются физические таблицы в автоматическом режиме, что позволяет не думать о том, создадутся ли они корректно. Также витрина данных дополнительно создаёт служебные индексы для работы системы. Контроль за целостностью таблиц также выполняет витрина данных.

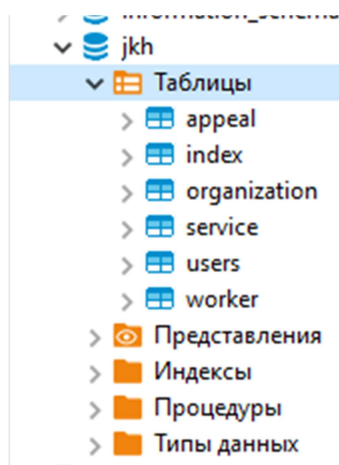


Рисунок 11 – Созданные таблицы

Описанными выше шагами, было выполнено следующие:

- развёрнута витрина данных;
- создано подключение в dbeaver;
- созданы таблицы в витрине данных.

5.2 Наполнение витрины тестовыми данными

Витрина данных поддерживает работу с данными в режиме дельт. Дельта – целостная совокупность изменений в логической базе данных. Дельты нумеруются по порядку в рамках логической базы данных, начиная с нуля.

Дельту можно открыть, закрыть, отменить или удалить.

Дельта, которую открыли и еще не закрыли, называется открытой или горячей, а дельта, которую закрыли, – закрытой. Для каждой логической базы данных одновременно может быть открыто не более одной дельты.

Обращаясь за данными в витрину, можно указывать из какой конкретной дельты получать результат, если не указана, используется последняя закрытая дельта.

На рисунке 12 показана последовательность дельт, где дельта 3 является открытой, а все предыдущие – закрытыми.

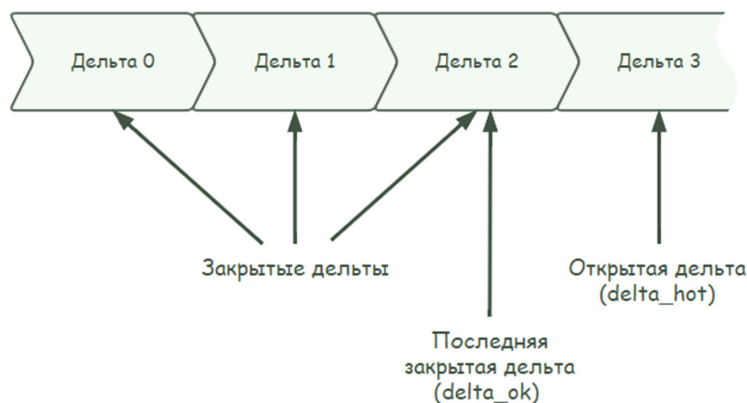


Рисунок 12 – Открытая и закрытая дельта

Загрузка может осуществляться двумя способами: через брокер сообщений или через sql выражение.

Для загрузки через брокер необходимо создать внешнюю таблицу, в листинге 5 приводится пример.

Листинг 5 – Создание внешней таблицы

```
Create upload external table jkh.index_ext (
    id_index int,
    i_value int,
    adress varchar,
    d_date date,
    id_service int

) location 'kafka://zookeeper:2181/JKH_INDEX_EXT' format 'AVRO';
```

Для осуществления загрузки необходимо направить сообщение в формате avro в топик 'JKH_INDEX_EXT'. Пример оформления сообщения представлен на рисунке 13. У направляемого сообщения ключ может иметь любое значение. Порядок следования полей в avro сообщении должен совпадать с порядком следования в логических таблицах. Типы данных также должны совпадать и соответствовать правилам конвертации.

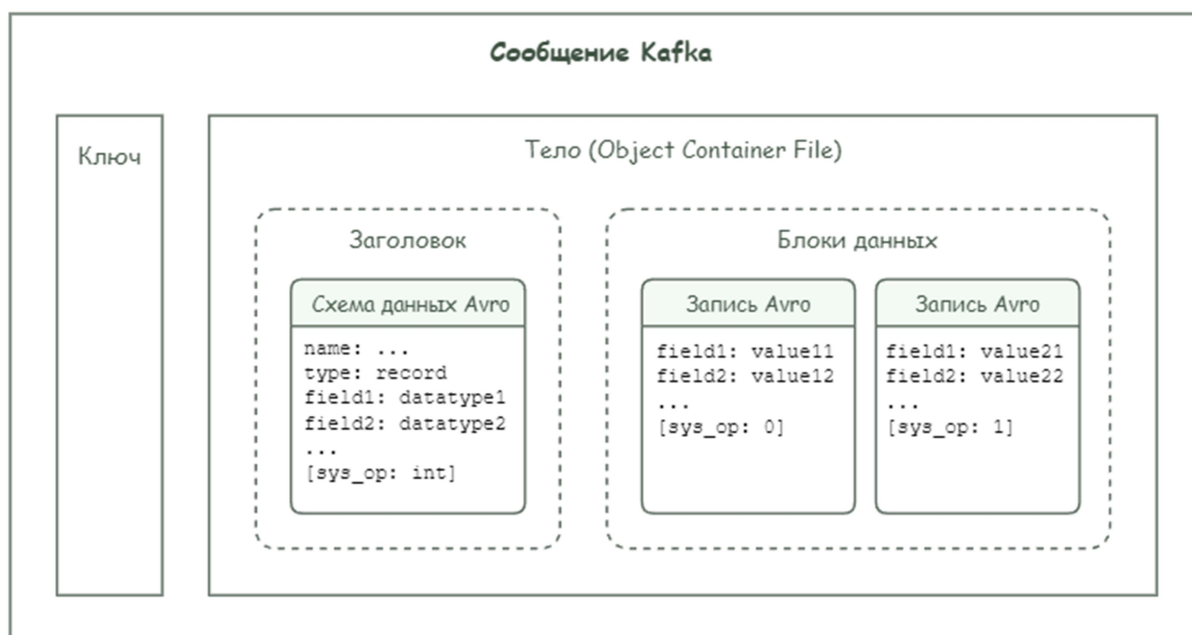


Рисунок 13 – Структура загружаемых сообщений

После загрузки необходимо открыть дельту, выполнить вставку в логическую таблицу, а затем зафиксировать дельту. Пример операций представлен на листинге 6.

Листинг 6 – Загрузка в логическую таблицу

```
Begin delta;
Insert into jkh.index select * from jkh.index_ext;
Commit delta;
```

Проверить то, что загрузка осуществилась успешно, можно выполнив sql выражение: «select * from jkh.index».

Альтернативный метод загрузки информации представляет из себя sql конструкцию «INSERT INTO [db_name.]table_name VALUES (value_list_1), (value_list_2);». Где db_name – название витрины, table_name – название логической таблицы, value_list_1, value_list_2 – набор данных, представленных списком через запятую. Пример такой вставки данных доступен в листинге 7.

Листинг 7 – Загрузка данных с помощью sql конструкции

```
Begin delta;
Insert into jkh.index values (1, 500, 'г. Пенза, ул. Пушкина, д.
4, кв. 5', date 2023-03-07, 1)
Commit delta;
```

Во время наполнения витрины данных был задействован второй метод размещения информации.

5.3 Реализация печатной формы

Для создания печатной формы будет задействован компонент printable-form-service, который позволяет получить данные из витрины, а также сформировать xml и pdf документ.

Шаблон получения данных представлен на листинге 8.

Листинг 8 – Получение данных

```
{# Предоставление информации о оказанных услугах #}
{# Входящие параметры #}

{% var id_user %}{{ params[0] }}{% endvar %}          {#
Идентификатор пользователя - Обязательно к заполнению #}
{% var date %}{{ params[1] }}{% endvar %}              {# Дата
обращения - Обязательно к заполнению #}
{% var patronymic %}{{ params[2] }}{% endvar %}        {#
Отчество - Обязательно при наличии у застрахованного лица #}

{# -----
----- #}
{# Получение данных #}
{% var GetDataQuery %} {# 1:1 #}
SELECT
    user.secondname,
    user.firstname,
    user.patronymic,
    user.adress,
    index.i_value,
    service.name_of_service,
    service.price,
    service.name_of_value,
    index.i_value * service.price as full_price,
    organization.name_of_organization
FROM
```

```

    jkh.index ind
    inner join jkh.users user on user.id_user = ind.id_user
    inner join jkh.service service on service.id_service =
ind.id_service
    inner join jkh.organization organization on
organization.id_organization = service.id_organization
WHERE
    index.i_date = {{ date }} and
    user.id_user = {{ id_user }}
{% endvar %}

{{ sql("Data", GetDataQuery) }}

{# формируем json #}
{% var json_data %}
{
    {% if rows.Data is empty %}
        "nodata": true
    {% else %}
        "nodata": false,
        "data": [
            {% for item in rows.Data %}
                {% if loop.first %}
                    {% else %}
                        ,
                    {% endif %}
                    {
                        "secondname": "{{ item.secondname }}",
                        "firstname": "{{ item.firstname }}",
                        "patronymic": "{{ item.patronymic }}",
                        "adress": "{{ item.adress }}",
                        "i_value": "{{ item.i_value }}",
                        "name_of_service": "{{ item.name_of_service }}",
                        "price": "{{ item.price }}",
                        "name_of_value": "{{ item.name_of_value }}",
                        "full_price": "{{ item.full_price }}",
                        "name_of_organization": "{{
item.name_of_organization }}"
                    }
                {% endfor %}
            ]
        {% endif %}
    }
{% endvar %}

{# выведем полученный json в неэкранированной форме #}
{{ json_data | raw }}

```

В нём используется расширение `sql`, которое позволяет обратиться к витрине данных за информацией, после строится `json` объект с данными.

На основе полученной информации строится в зависимости от выбора xml или pdf объект.

5.4. Тестирование производительности

Современное программное обеспечение просто обязано бесперебойно работать под колоссальными нагрузками. Любого рода проблемы, связанные с плохой производительностью, могут стать причиной отказа клиентов от использования вашего ПО. В связи с этим, проведение качественного нагрузочного тестирования должно стать обязательным, для обеспечения стабильности работы ваших приложений.

Нагрузочное тестирование (Load Testing) или тестирование производительности (Performance Testing) – это автоматизированное тестирование, имитирующее работу определенного количества бизнес пользователей на каком-либо общем (разделяемом ими) ресурсе [21].

Нагрузочное тестирование производилось с помощью утилиты Apache Jmeter, который позволяет создавать профили тестирования. Для получения результата использовалось хранилище метрик InfluxDB 1.8x. Значения для подстановки брались из Redis, предварительно наполнив его.

Результат, который удалось получить в разы превышает заявленный для корректной работы информационной системы, а именно 2,500 запросов в секунду с средним временем выполнения 40 миллисекунд.

Длительность теста составляла 5 минут.

Количество потоков увеличивалось от 2000 до 2500 каждые 30 секунд.

1 поток соответствует действиям 1 пользователям.

Описание тестового окружения представлено в таблице 13.

Таблица 13 – Тестовое окружение

Текущая версия системы	5.8.0
Сервер	Srv-dtm-lite
Драйвер	dtm-jdbc-5.8.0.jar
Пул коннектов драйвера	0 (Пул не ограничен)
Количество записей в БД	2_317_300

Продолжение таблицы 13

Запрос	<pre> SELECT user.secondname, user.firstname, user.patronymic, user.adress, index.i_value, service.name_of_service, service.price, service.name_of_value, index.i_value * service.price as full_price, organization.name_of_organization FROM jkh.index ind inner join jkh.users user on user.id_user = ind.id_user inner join jkh.service service on service.id_service = ind.id_service inner join jkh.organization organization on organization.id_organization = service.id_organization WHERE index.i_date = '\${UC01_i_date}' and user.id_user = '\${UC01_id_user}' </pre>
Объем данных в МБ	485(data) + 63 (indexes)
Конфигурация стенда	12 CPU 64 RAM 63 GiB

После проведения нагрузочного тестирования, получили метрики работы информационной системы, которые показаны на рисунках 14 – 16.

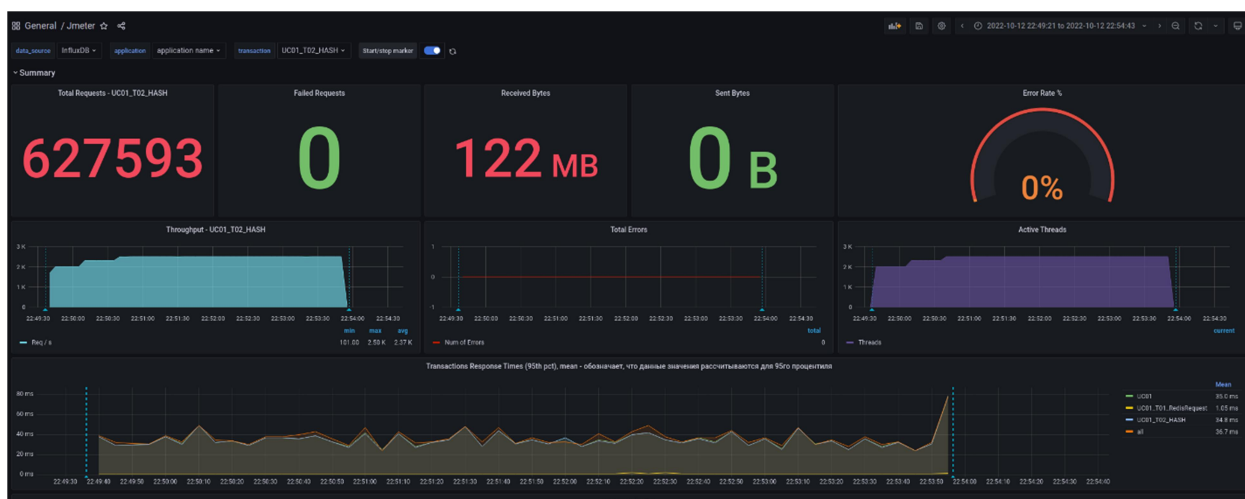


Рисунок 14 – График RPS и времени ответа

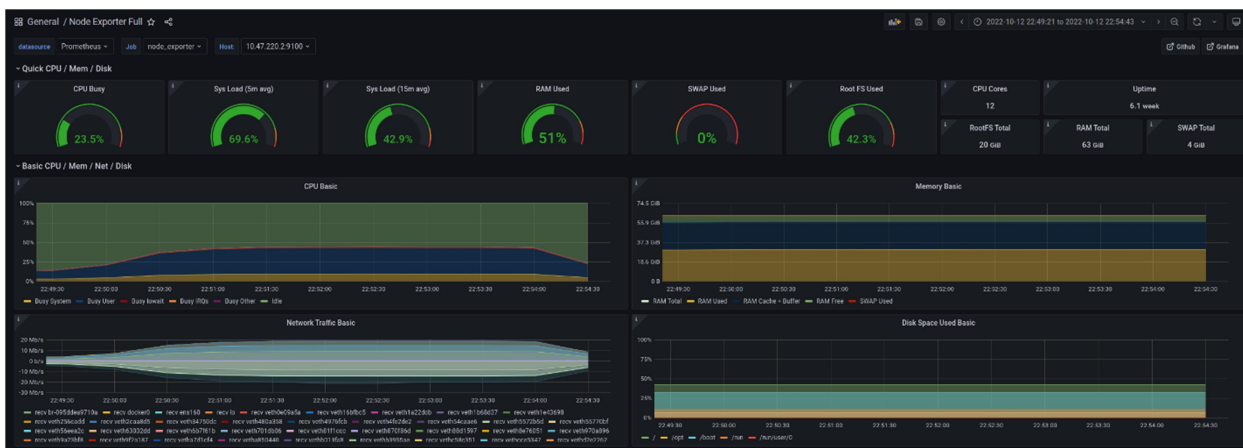


Рисунок 15 – Графики потребления аппаратных ресурсов

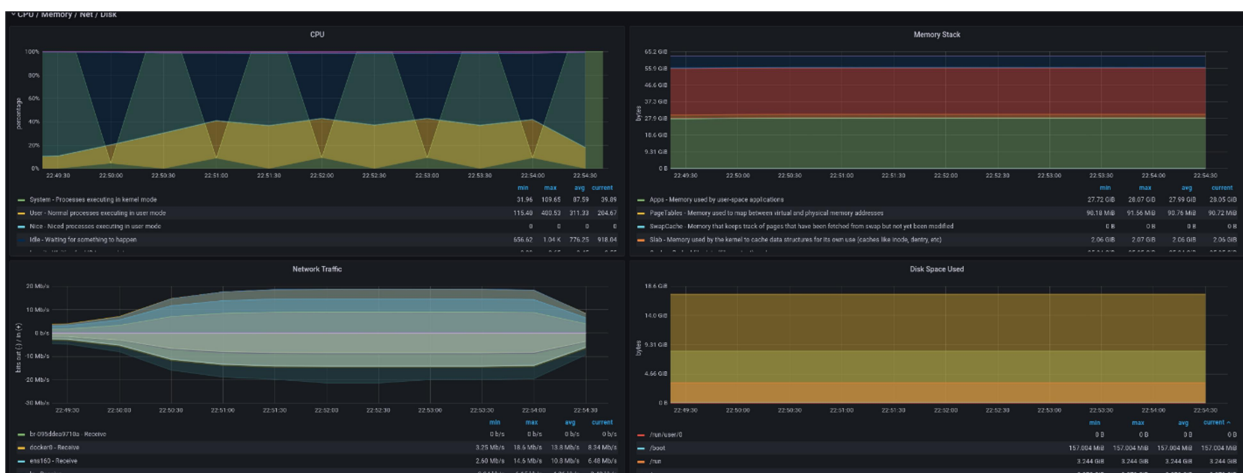


Рисунок 16 – Графики потребления ресурсов

Утилизация CPU на сервере до 34%, RAM до 51%.

Использование ресурсов диска представлено на рисунке 17.

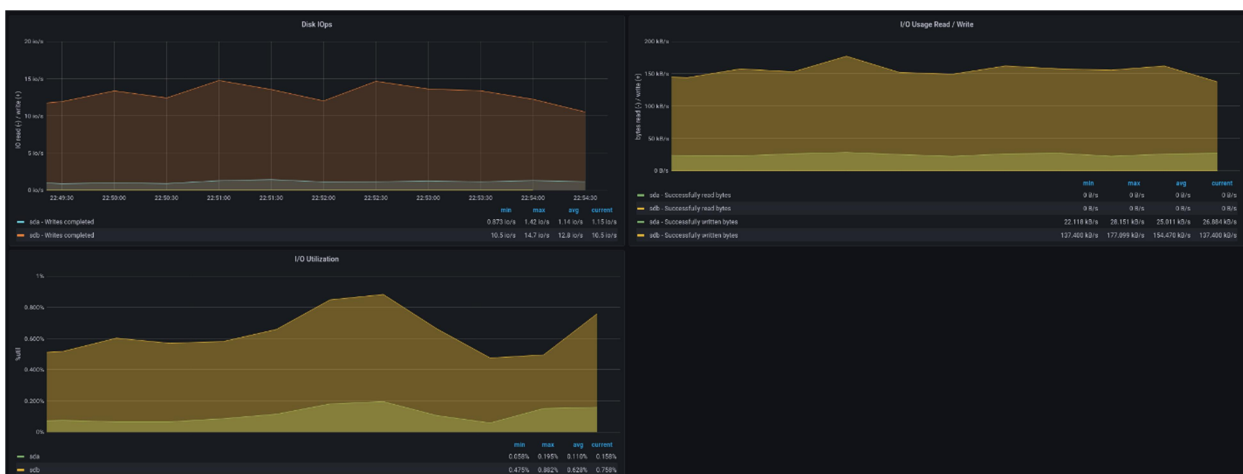


Рисунок 17 – График потребления ресурсов

В ходе выполнения нагрузочного тестирования витрина данных показала положительный результат.

6 Планирование и расчёт бюджета разрабатываемой информационной системы

Оценка бюджета разработки мобильного приложения является важным этапом при планировании проекта. Она помогает определить финансовые ресурсы, необходимые для создания и запуска приложения, и обеспечивает понимание общей стоимости разработки.

При оценке бюджета разработки мобильного приложения следует учитывать несколько ключевых факторов.

Функциональность приложения: чем больше функциональных возможностей и сложных компонентов требуется в приложении, тем больше затрат на разработку. Необходимо провести детальный анализ требований и определить, какие функции приложения являются наиболее важными и необходимыми для первоначального выпуска, а какие могут быть добавлены позже в виде обновлений.

Дизайн и пользовательский интерфейс: Привлекательный и интуитивно понятный дизайн является важным фактором успеха мобильного приложения. Однако профессиональное создание дизайна и разработка пользовательского интерфейса требуют дополнительных затрат на дизайнеров и UX/UI-специалистов.

Платформы и устройства: если планируется выпуск приложения для нескольких платформ (например, iOS и Android), то необходимо учесть затраты на разработку и адаптацию приложения под каждую из платформ. Кроме того, существуют различные устройства с разными размерами экранов, и их поддержка может потребовать дополнительных усилий.

Тестирование и обеспечение качества: критически важно провести тщательное тестирование приложения, чтобы обнаружить и исправить ошибки и неполадки. Поддержка и обновление приложения также могут потребовать финансовых ресурсов.

Масштабирование и поддержка: если планируется дальнейшее развитие и масштабирование приложения после его запуска, следует учесть дополнительные расходы на поддержку, обновления и улучшения функциональности.

Для планирования процесса разработки и расчета бюджета можно использовать диаграмму Ганта.

Диаграмма Ганта – это визуальное представление графика работ, построенное согласно плану проекта. На ней отражены задачи и последовательность их выполнения.

График работ состоит из ряда отрезков, размещённых вдоль временной оси. Каждый из них соответствует отдельной задаче или подзадаче. Начало и конец отрезка соответствуют моменту начала и завершения работы по задаче. Длина отрезка – продолжительность работ [22].

Для построения диаграммы необходимо выделить задачи, а также сроки их выполнения. Задачи проекта представлены в таблице 14.

Таблица 14 – Задачи проекта

Название задачи	Длительность задачи в днях	Дата начала	Дата окончания
Начало проекта	0	12.04.2023	12.04.2023
Постановка задачи и анализ аналогов разрабатываемого приложения	4	12.04.2023	16.04.2023
Анализ аналогов	2	12.04.2023	14.04.2023
Анализ функциональных требований	1	14.04.2023	15.04.2023
Выбор инструментальных средств проектирования и разработки	1	15.04.2023	16.04.2023
Проектирование информационной системы	9	17.04.2023	26.04.2023
Проектирование архитектуры системы	4	17.04.2023	21.04.2023

Продолжение таблицы 14

Название задачи	Длительность задачи в днях	Дата начала	Дата окончания
Проектирование печатной формы	5	21.04.2023	26.04.2023
Реализация информационной системы	14	27.04.2023	11.05.2023
Настройка оборудования	11	27.04.2023	08.05.2023
Реализация печатной формы	3	08.05.2023	11.05.2023
Тестирование приложения	4	12.05.2023	16.05.2023
Тестирование производительности	4	12.05.2023	16.05.2023
Создание пояснительной записки	16	17.05.2023	01.06.2023
Написание пояснительной записки	9	17.05.2023	26.05.2023
Оформление и печать пояснительной записки	7	26.05.2023	01.06.2023
Конец проекта	0	01.06.2023	01.06.2023

На основе выделенных задач и сроков была построена диаграмма Ганта, представленная на рисунке 18.

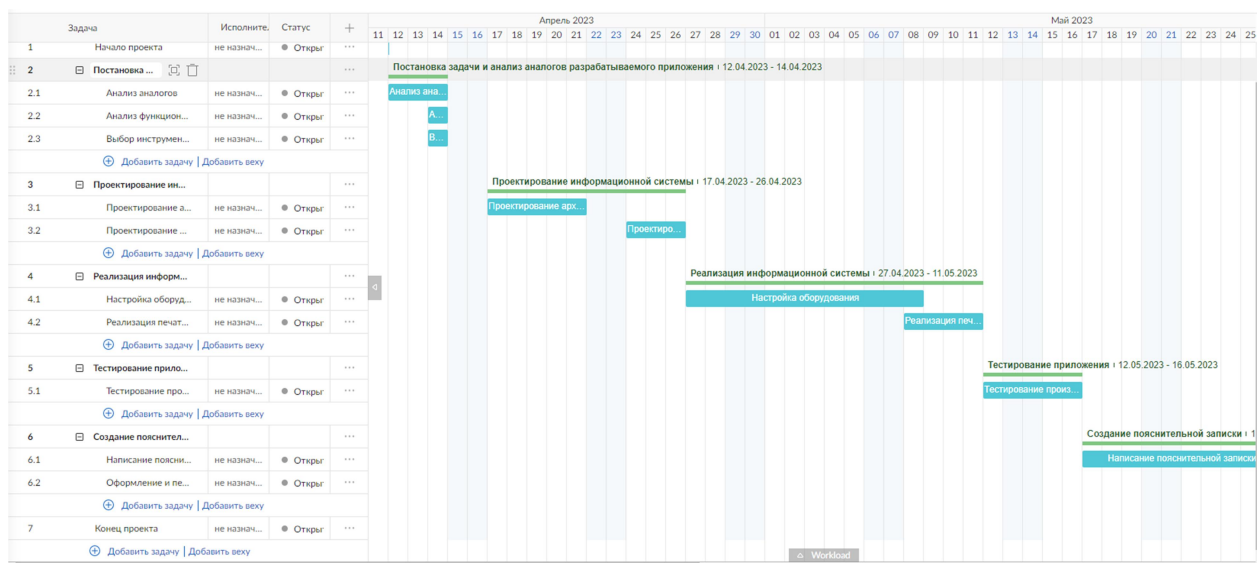


Рисунок 18 – Диаграмма Ганта

Для подсчета затрат на разработку необходимо определить ресурсы проекта. Выделенные ресурсы приведены в таблице 15.

Таблица 15 – Ресурсы проекта

Название ресурса	Тип	Стандартная ставка	Начисление
Аналитик	Трудовой	100,00 Р/ч	Пропорциональное
Проектировщик	Трудовой	200,00 Р/ч	Пропорциональное
Инженер	Трудовой	300,00 Р/ч	Пропорциональное
Дизайнер	Трудовой	150 Р/ч	Пропорциональное
Тестировщик	Трудовой	100 Р/ч	Пропорциональное
Бумага	Материальный	500,00 Р	В начале
Принтер	Материальный	5 000,00 Р	В начале

Для описанных ранее задач, на основе ресурсов, была рассчитана стоимость, необходимая для выполнения каждого этапа разработки. Результат представлен в таблице 16.

Таблица 16 – Задачи и затраты проекта

Название задачи	Длительность задачи в днях	Название ресурса	Затраты
Начало проекта	0	Компьютер, Смартфон, Бумага, Принтер	70 500,00 Р
Постановка задачи и анализ аналогов разрабатываемой информационной системы	4		3 200,00 Р
Анализ аналогов	2	Аналитик	1 600,00 Р
Анализ функциональных требований	1	Аналитик	800,00 Р
Выбор инструментальных средств проектирования и разработки	1	Аналитик	800,00 Р
Проектирование информационной системы	9		20 400,00 Р

Продолжение таблицы 16

Название задачи	Длительность задачи в днях	Название ресурса	Затраты
Проектирование информационной системы	4	Проектировщик	6 400,00 Р
Проектирование печатной формы	5	Проектировщик, Дизайнер	14 000,00 Р
Реализация информационной системы	14		32 400,00 Р
Настройка оборудования	11	Инженер	26 400,00 Р
Реализация печатной формы	3	Аналитик, Дизайнер	6 000,00 Р
Тестирование приложения	4		3 200,00 Р
Тестирование производительности	4	Тестирующий	3200,00 Р
Создание пояснительной записки	16		12 800,00 Р
Написание пояснительной записки	9	Аналитик	7 200,00 Р
Оформление и печать пояснительной записки	7	Аналитик	5 600,00 Р
Конец проекта	0		

Таким образом, по результатам оценки бюджета разработки мобильного приложения была составлена диаграмма Ганта, рассчитано время необходимое на полный цикл разработки проекта, которое составляет 47 дней, на основании необходимых ресурсов рассчитан бюджет проекта, который равен 142 500 рублей.

Заключение

В ходе выполнения выпускной квалификационной работы был проведён анализ предметной области, составлена её модель, выделены основные варианты использования, произведена постановка задач и анализ требований к разрабатываемой информационной системе ЖКХ. Спроектированная витрина данных была протестирована на производительность, которые показали соответствие заявленным требованиям. Результат обработки составил 2500 запросов в секунду.

Список использованных источников

1. Приказ Госстроя РФ от 29.12.1997 N 17-142 (ред. от 23.02.1999) "Об утверждении Методики проведения аудита тарифов в организациях, оказывающих жилищно-коммунальные услуги" // Справочная правовая система «КонсультантПлюс». [Электронный ресурс]. URL: <https://www.consultant.ru> (дата обращения 10.06.2023).
2. Постановление Правительства РФ от 13.08.2006 N 491 (ред. от 29.06.2020) "Об утверждении Правил содержания общего имущества в многоквартирном доме и правил изменения размера платы за содержание жилого помещения в случае оказания услуг и выполнения работ по управлению, содержанию и ремонту общего имущества в многоквартирном доме ненадлежащего качества и (или) с перерывами, превышающими установленную продолжительность"// Справочная правовая система «КонсультантПлюс». [Электронный ресурс]. URL: <https://www.consultant.ru> (дата обращения 10.06.2023)
3. Государственная информационная система жилищно-коммунального хозяйства [Электронный ресурс]. URL: <https://dom.gosuslugi.ru/#!/implementation-map> (Дата обращения 10.04.2023)
4. ПИК Комфорт [Электронный ресурс]. URL: <https://pik-comfort.ru/> (Дата обращения 10.04.2023)
5. Строительный холдинг «Термодом» // Город Спутник. [Электронный ресурс]. URL: <https://penza-sputnik.ru/> (Дата обращения 10.04.2023)
6. ГОСТ 34.602-89 «Техническое задание на создание автоматизированной системы» // Комплекс стандартов на автоматизированные системы. Москва: Стандартинформ, 1990.
7. Ермаков, А.А. Основы надежности информационных систем: учебное пособие / А.А. Ермаков. – Иркутск : ИрГУПС, 2006. – 151 с.

8. StarUML. Руководство разработчика. [Электронный ресурс]. URL: [http://staruml.sourceforge.net/docs/developer-guide\(ru\)/developer-guide.pdf](http://staruml.sourceforge.net/docs/developer-guide(ru)/developer-guide.pdf) (дата обращения: 20.05.2023)
9. Официальный портал Правительства Ростовской области [Электронный ресурс]. URL: <https://www.donland.ru/result-report/1220> (Дата обращения 10.04.2023)
10. Леоненков, А. В. Самоучитель UML 2 / А.В. Леоненков. – СПб.: БХВ-Петербург, 2007. – 576 с.
11. Сергеев С.Ф., Падерно П.И., Назаренко Н.А. Введение в проектирование интеллектуальных интерфейсов: Учебное пособие. – СанктПетербург: СПбГУ ИТМО, 2011. – 108 с.
12. UML 2 и унифицированный процесс. Практический объектноориентированный анализ и проектирование, 2-е издание. – Пер. с англ. – СПб: Символ-Плюс, 2007. – 617 с.
13. Кролл П., Крачтен Ф. Rational Unified Process – это легко. Руководство по RUP для практиков. Пер. с англ. – М.: КУДИЦ-ОБРАЗ, 2004. – 432 с.
14. Крачтен Ф. Введение в Rational Unified Process Издательство: М.: Вильямс; Издание 2-е, 2002. – 240 с.
15. Заботина Н. Н. Проектирование информационных систем: Учебное пособие / Н.Н. Заботина. – М.: НИЦ ИНФРА – М, 2022 – 331 с.
16. Репозиторий типового ПО Витрины данных. [Электронный ресурс]. URL: https://g.info.gov.ru/typical_datamart_vd/dtm-prostore/-/tree/main/ (Дата обращения: 10.05.2023)
17. ООО «Датамарт» // Информационно-методический справочник. [Электронный ресурс]. URL: https://prostore.datamart.ru/docs_prostore/ (Дата обращения: 10.05.2023)
18. А. М. Васильев Использование шаблонизатора Pebble // Разработка веб-приложений [Электронный ресурс]. URL:

<https://lms.crafted.su/web-app-development/2023-1-ivt-2/docs/course/01-basics/05-using-pebble-templates/> (Дата обращения: 20.04.2023)

19. Skillfactory media Kafka Apache // Kafka Apache - что это и где применяется, основы технологии [Электронный ресурс]. URL: <https://blog.skillfactory.ru/glossary/kafka-apache/> (Дата обращения: 20.04.2023)

20. Шустова Л. И. Базы данных: Учебник / Шустова Л.И., Тараканов О.В. – М.: НИЦ ИНФРА-М, 2016. – 304 с.

21. ProTesting.ru Нагрузочное тестирование // Нагрузочное тестирование [Электронный ресурс]. URL: <http://www.protesting.ru/automation/performance.html> (Дата обращения: 20.04.2023)

22. Разбираем диаграмму Ганта – инструмент, который должен знать каждый менеджер // Skillbox. [Электронный ресурс] URL: <https://skillbox.ru/media/management/razbiraem-diagrammu-ganta-instrument-kotoryy-dolzhen-znat-kazhdyy-menedzher/> (дата обращения: 10.06.2023).