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

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

Кафедра «Прикладная математика и информатика»
(наименование)

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

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

# ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему «Разработка ПО для автоматизации сбора показаний приборов учета»

Обучающийся М.Д. Ермилов
(Инициалы Фамилия) (личная подпись)

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

Консультант А.В. Егорова
ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Тольятти 2023

## Аннотация

Тема выпускной квалификационной работы - «Разработка приложения для автоматизации сбора показаний приборов учета»

Актуальность данной работы заключается в необходимости автоматизации процесса сбора и обработки показаний приборов учета для упрощения и ускорения работы организаций.

Объектом исследования данной выпускной квалификационной работы является автоматизация процесса сбора показаний приборов учета в организации ООО «Квартплата 24».

Предметом исследования является разработка приложения для автоматизации сбора показаний приборов учета.

Результаты выполнения квалификационной работы имеют практическую ценность и могут быть использованы организациями для упрощения и автоматизации процесса сбора и обработки показаний приборов учета.

В первой главе приводится краткая характеристика организации, анализ её существующих способов автоматизации сбора показаний приборов учета и формулировка требований к разрабатываемому приложению.

Во второй главе описывается архитектура приложения, описываются возможности его использование, производится функциональное моделирование и выбор платформы для его разработки.

В третьей главе производится выбор инструментов реализации, реализация приложения со всеми требуемыми возможностями и тестирование разработанного приложения.

В заключении описаны результаты выполнения выпускной квалификационной работы.

Бакалаврская работа состоит из трех глав, заключения и списка используемой литературы.

Бакалаврская работа содержит 40 страниц текста, 13 рисунков, 3 таблицы, 30 источников.

## Abstract

The title of the graduate qualification work is «Development of software for automation of meter reading collection».

The relevance of this graduate qualification work is the need to automate the process of collecting and processing of meter readings to simplify and accelerate the work of organizations.

The object of this graduate qualification work is to automate the process of collecting meter readings in the organization of OOO «Kvartplata 24».

The subject of this graduate qualification work is the development of an application to automate the collection of metering devices readings.

The results of this qualification work has practical value and can be used by organizations to simplify and automate the process of collecting and processing of meter readings.

The graduation work may be divided to into several logically connected which are: first chapter gives a brief description of the organization, an analysis of its existing ways to automate the collection of metering devices readings and formulating the requirements to the developed application.

The second chapter describes the architecture of the application, describes the possibilities of its use, the functional modeling and platform selection for its development.

In the third chapter, the choice of implementation tools, implementation of the application with all the required features and testing of the developed application.

The conclusion describes the results of the graduate qualification work.

Graduate qualification work consists of three chapters, the conclusion and a list of references.

The qualification work consists of an explanatory note on 40 pages of text, 13 figures, 3 tables, and the list of 30 references.

**Оглавление**

# Введение

Разработка программного обеспечения для автоматизации сбора показаний приборов учета является актуальной в связи с ростом числа объектов, на которых необходимо производить учет потребления энергоресурсов (вода, газ, электроэнергия и т.д.), а также повышением требований к точности и скорости сбора данных.

Традиционный способ сбора показаний приборов учета включает в себя вызов специалистов, которые производят замеры и записывают показания в журналы. Однако этот подход устарел и неэффективен, так как требует больших временных и финансовых затрат, а также может приводить к ошибкам при записи данных.

Автоматизация сбора показаний приборов учета позволяет существенно сократить время и стоимость процесса учета, а также повысить его точность и надежность. Кроме того, программное обеспечение для автоматизации сбора показаний приборов учета дает возможность создать единую базу данных о потреблении энергоресурсов, которая может быть использована для анализа и управления потреблением. Это позволяет сократить расходы на энергоресурсы и снизить вредное воздействие на окружающую среду.

Таким образом, разработка ПО для автоматизации сбора показаний приборов учета является важным шагом в направлении улучшения учета потребленных ресурсов и увеличения эффективности использования энергоресурсов.

В современном мире использование приборов учета - неотъемлемая часть процесса сбора показаний и контроля за энергопотреблением. Это связано с растущими ценами на энергоресурсы и все большим желанием людей экономить энергию и сокращать свои расходы. Организация ООО «Квартплата 24» использует информацию о показаниях приборов учета - следовательно ей необходимо иметь эффективную систему для сбора и

обработки показаний.

Актуальность данной работы заключается в необходимости автоматизации процесса сбора показаний прибора учета для уменьшения затрат времени на передачу показаний приборов учета и упрощения процесса взаимодействия жильцов и организаций.

Цель данной ВКР - разработка приложения для персональных компьютеров для автоматизации сбора показаний приборов учета. Разработанное приложение повысит эффективность сбора и обработки данных, уменьшить возможность ошибок и искажений в данных, а также обеспечит целостность и надежность хранения и использования информации.

Для достижения данной цели необходимо выполнить следующие задачи:

– изучить способы автоматизации сбора показаний приборов учета в организации ООО «Квартплата 24»;

– сформулировать требования к разрабатываемому приложению;

– разработать приложение, удовлетворяющее требованиям;

– протестировать разработанное приложения.

Объектом исследования данной ВКР является автоматизация сбора показаний приборов учета в организации ООО «Квартплата 24»

Предметом исследования является разработка приложения для автоматизации сбора показаний приборов учета.

Первая глава содержит характеристику организации, анализ существующей в ней системы сбора показания приборов учета и выработку требований к разрабатываемому приложению.

Вторая глава содержит проектирование приложения.

В третьей представлен выбор средств для разработки, реализация и тестирование разработанного ПО.

Таким образом, данная работа имеет практическую ценность и может быть использована организациями для упрощения и автоматизации процесса сбора и обработки показаний приборов учета.

# Глава 1 Характеристика организации и анализ существующих технологий сбора показаний приборов учета

## 1.1 Характеристика организации

ООО «Квартплата 24» – Российская IT-компания, занимающаяся разработкой и сопровождением экосистемы из более чем 10 тесно интегрированных между собой облачных сервисов, решающих задачи расчёта и учета платы за ЖКХ, приема платежей и их распределение, а также взыскание долгов в соответствии с законодательством РФ на всей её территории.

Компания имеет опыт работы в IT сфере ЖКХ с 1996 года и обслуживает свыше 1 миллиона лицевых счетов, имеет более 700 клиентов ТСЖ, УО, РСО в более 50 субъектах РФ.

Квартплата 24 объединяет более 40 банков и платежных систем, обеспечивает более 30 способов электронной оплаты, производит прием платежей по всей территории Российской федерации и за ее пределами, а также является единственным в РФ платежным сервисом, который обеспечивает моментальный прием и расщепление платежей за жилищно-коммунальные услуги на всех этапах прохождения платежа.

«Квартплата 24» является резидентом технопарка в сфере высоких технологий «Жигулевская долина» и членом НП СРО «Национальный Жилищный Конгресс», а также включена Минстроем России в Банк данных умных городов.

## 1.2 Способы автоматизации сбора показаний в организации ООО «Квартплата 24»

Одним из важных этапов работы организации ООО «Квартплата 24» является сбор показаний приборов учета. В домах, обслуживаемых

организацией Квартплата 24, используются приборы учета, такие как газовые счетчики, электросчетчики, счетчики воды и тепла. Они позволяют точно учитывать потребление ресурсов каждым абонентом и формировать платежные документы.

Автоматизация сбора показаний приборов учета является важным элементом взаимодействия организации и абонентов. Это позволяет сократить время на подготовку платежных документов, снизить риски ошибок в расчетах и упростить процесс взаимодействия. Для ускорения процесса сбора показаний существует два способа его автоматизации в ООО «Квартплата 24», которые представлены не рисунке 1.

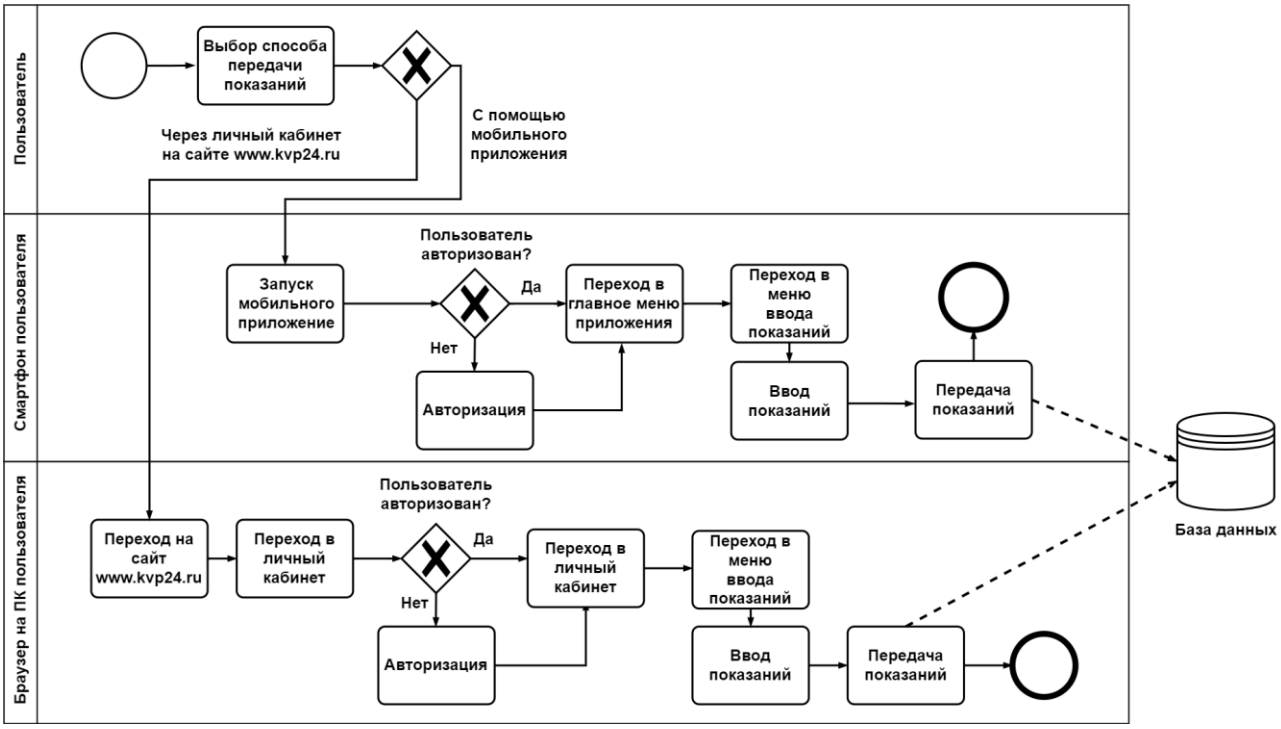


Рисунок 1 – Схема существующих вариантов сбора показаний приборов учета

Они осуществляются при помощи ручного ввода показаний на сайте или в мобильном приложении. Для их реализации требуется выполнить следующие действия:

– зайти в личный кабинет на сайте www.kvp24.ru или в мобильное приложение «Квартплата 24: Личный кабинет»;

– открыть вкладку, в которой производится ввод показаний приборов учета;

– ввести показания приборов учета;

– нажать на кнопку «Отправить» в мобильном приложении или «Передать показания» на сайте.

После этого данные показаний будут сохранены в базе данных организации. Отправка показаний через сайт или мобильное приложение позволяет жильцам удобно и быстро предоставлять показания своих приборов учета. Таким образом автоматизация сбора показаний приборов учета значительно упрощает процесс взаимодействия жильца и организации и сокращает время, затрачиваемое на него. Важно отметить, что для успешной отправки показаний через сайт или мобильное приложение необходимо иметь доступ к интернету.

### 1.3 Требования к проектируемому приложению

Основываясь на информации, приведенной в разделе 1.2 можно сделать вывод, что ускорение процесса взаимодействия жильца и организации имеет важную роль в жизни современного человека, и целью данной выпускной квалификационной работы является разработка приложения для автоматизации сбора показаний приборов учета.

Задание, выданное организацией ООО «Квартплата 24» гласит, что проектируемая технология должна представлять из себя приложение для ПК, у которого имеются такие возможности, как ввод новых показаний приборов учета и просмотр истории показаний.

Так же для формирования дополнительных требований к разрабатываемому приложению воспользуемся классификацией FURPS+. FURPS+ (Functionality, Usability, Reliability, Performance, Supportability, +

Security) - это методология оценки качества программного обеспечения, разработанная в IBM. Она включает в себя шесть атрибутов, которые определяют важные характеристики программного обеспечения: функциональность, удобство использования, надежность, производительность, поддерживаемость и безопасность. Эта методология используется для определения требований к программному обеспечению и оценки его соответствия этим требованиям [2]. Перечень требований представлен в таблице 1, где в столбце «Приоритет», «1» означает наименьший приоритет, а «5» - наивысший.

Таблица 1 – Требования к разрабатываемому приложению

| | Требование | Приоритет | Описание |
|---|---|---|---|
| Functionality | | | |
| 1 | Передача новых показаний приборов учета | 5 | Приложение должно иметь возможность передачи пользователем показаний приборов учета холодного водоснабжения, горячего водоснабжения, электроэнергии, газа и отопления с последующим внесением их в базу данных. |
| 2 | Просмотр истории показаний приборов учета | 5 | Приложение должно иметь возможность просмотра пользователем показаний приборов учета, внесенных ранее даже при отсутствии подключения к интернету |
| 3 | Возможность передачи показаний приборов учета разных типов | 5 | Приложение должно иметь возможность передачи показаний приборов учета таких типов, как холодное и горячее водоснабжение, электроэнергия, газ, отопление. |
| Usability | | | |
| 1 | Удобство и понятность интерфейса | 5 | Каждое действие в приложении должно быть интуитивно понятным. |

Продолжение таблицы 1

| | | | |
|---|---|---|---|
| 2 | Комфортная цветовая гамма приложения | 3 | Цветовая гамма не должна создавать напряжения для глаз пользователя. |
| 3 | Читабельные и понятные шрифты | 3 | Текст в приложении должен быть удобен для прочтения |
| Reliability | | | |
| 1 | Стабильность | 1 | Приложение должно работать стабильно, без сбоев и ошибок |
| 2 | Проверка на ошибки | 1 | Приложение должно проверять входные данные на наличие ошибок и уметь их обрабатывать |
| 3 | Доступность системы 24/7 | 5 | Приложение должно иметь круглосуточный режим функционирования |
| Performance | | | |
| 1 | Быстродействие приложения | 3 | Время запуска приложение должно составлять менее трех секунд, а время отклика на действия пользователя – менее одной секунды |
| Supportability | | | |
| 1 | Поддержка операционной системы Windows | 5 | Приложение должно корректно работать на персональных компьютерах с операционной системой Windows |

Таким образом проектируемое приложение имеет 2 функциональных требования, которые определяют его основные возможности и приоритеты в их реализации.

### 1.4 Сравнительный анализ используемых аналогов

Для целесообразности разработки приложения необходимо проверить, есть ли среди используемых аналогов те, что подходят под требования к разрабатываемому приложению. Для сравнения были выбраны наиболее

популярные аналоги - «Инфокрафт ЖКХ 365», «Диспетчер24», «Hudson.im».

Основным назначением разрабатываемого приложения является передача показаний приборов учета разного типа, однако в разделе 1.3 были описаны дополнительные возможности, наличие которых требуется в нем - следовательно сравнение выбранных аналогов «Энергетика», «Energy Manager», «Pelican Wireless» и «MeterNet» будет проводиться по всем функциям, наличие которых требуется в разрабатываемом приложении. Сравнение будет производиться по критериям:

- передача показаний приборов учета;
- просмотр истории показаний приборов учета;
- возможность передачи показаний разных типов счетчиков (горячая вода, холодная вода, электроэнергия, газ, отопление).

Оценка будет производиться по шкале от 0 до 5, где 0 - полное несоответствие требованиям, 5 - полное соответствие требованиям.

Таблица 2 – Сравнение аналогов разрабатываемого приложения

| | «Инфокрафт ЖКХ 365» | «Диспетчер24» | «Hudson.im» |
|---|---|---|---|
| Передача показаний приборов учета | 5 | 4 | 4 |
| Просмотр истории показаний приборов учета | 5 | 5 | 3 |
| Возможность передачи показаний разных типов счетчиков (Горячая вода, холодная вода, электроэнергия, газ, отопление) | 5 | 4 | 5 |
| Наличие приложения для ПК | 0 | 0 | 0 |

Оценка возможностей приложений по представленным критериям показала, что во всех выбранных аналогах присутствуют возможности,

требуемые для разработки приложения в данной выпускной квалификационной работе, однако ни у одного из них нет приложения для ПК. Исходя из результатов сравнения можно сделать вывод о том, что достойные аналоги существуют, у них нет приложения для ПК, поэтому разработка приложения в рамках данной работы по выработанным критериям является актуальной и целесообразной.

## Выводы по главе 1

В данной главе были рассмотрены способы сбора показаний приборов учета в организации ООО «Квартплата 24» и приведена её краткая характеристика.

Были определены требования к разрабатываемому как при помощи системы классификации FURPS+, так и исходя из задания заказчика.

Также был проведен сравнительный анализ аналогов разрабатываемого приложения, который показал, что его разработка является актуальной и целесообразной.

# Глава 2 Проектирование приложения

## 2.1 Выбор платформы реализации приложения

Основываясь на данных отчета онлайн-сервиса Statcounter о процентном соотношении используемых операционных систем во всем мире за прошедший год (https://gs.statcounter.com/os-market-share/desktop/worldwide/#monthly-202204-202304-bar), можно сделать выводы о том, что доля в 73.48% приходится на операционную систему Windows, 15.51% на OS X ( macOS), 2.73% на Linux, 2.43% на Chrome OS, 0.01% на FreeBSD и 5.84% на прочие операционные системы. Данная статистика представлена на рисунке 2.

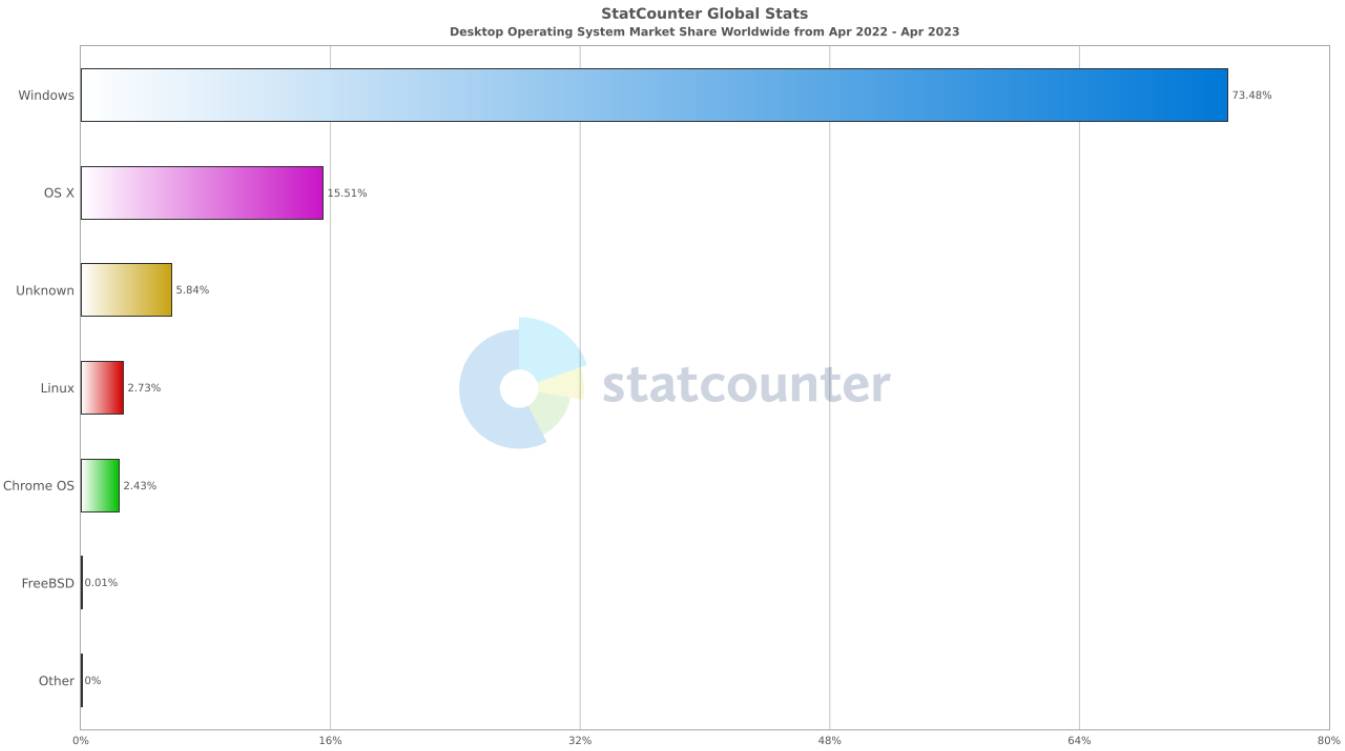


Рисунок 2 – статистика используемых операционных систем

Исходя из представленных данных, выбор операционной системы, для которой будет разрабатываться приложение падает на Windows, так как она

является наиболее популярной ОС у пользователей персональных компьютеров.

### 2.2 Проектирование архитектуры приложения

Приложение для автоматизации сбора показаний приборов учета будет иметь архитектуру «Клиент-сервер». Клиентская часть - это приложение, которое установлено на локальном компьютере пользователя. С помощью этого приложения пользователь может вводить данные о показаниях приборов учета и отправлять их на серверную часть приложения, а также производить другие действия, использующие эти данные. Серверная часть - это веб-сервис, который принимает запросы от клиентской части и производит действия с данными из базы данных. Схема архитектуры проектируемого приложения представлена на рисунке 3.

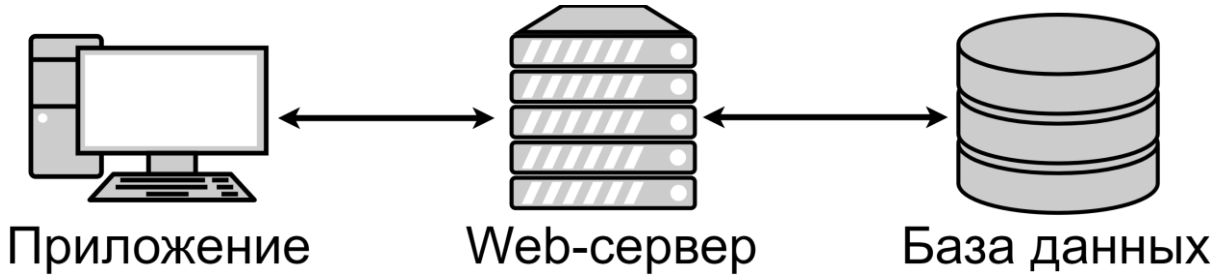


Рисунок 3 – Архитектура проектируемого приложения

Приложение будет состоять из двух компонентов:

- классы управления приложением;
- класс прикладного программного интерфейса базы данных.

Классы управления приложением – отвечают за взаимодействие пользователя с приложением, то есть отображение пользовательского интерфейса, обработку действий пользователя и отправку запросов классу прикладного программного интерфейса. Класс прикладного программного

интерфейса – отвечает за выполнение запросов к базе данных, то есть добавление, получение и редактирование данных в ней. Диаграмма компонентов приложения представлена на рисунке 4.

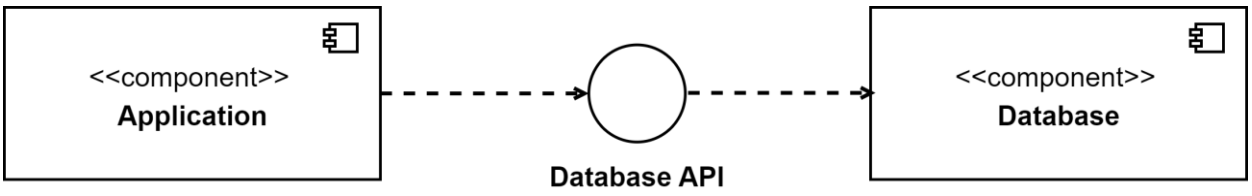


Рисунок 4 – Диаграмма компонентов проектируемого приложения

Таким образом для реализации приложения была выбрана трехзвенная архитектура «Клиент-сервер».

### 2.3 Проектирование логической модели приложения

Логическая модель приложения представляет собой описание объектов, свойств и отношений между ними, которые используются в приложении. Она описывает основную логику приложения, его функции и взаимодействие с другими системами или компонентами [27].

Для начала построим диаграмму вариантов использования проектируемого приложения.

Диаграмма вариантов использования является основным инструментом анализа и проектирования системы, позволяющим установить требования к функциональности системы, ее взаимодействию с пользователями и окружающей средой.

Она позволяет определить основные варианты использования системы, их связи и последовательность взаимодействия, а также выделить роли пользователей и определить их задачи и цели [27]. Диаграмма вариантов использования проектируемого приложения представлена на рисунке 5.

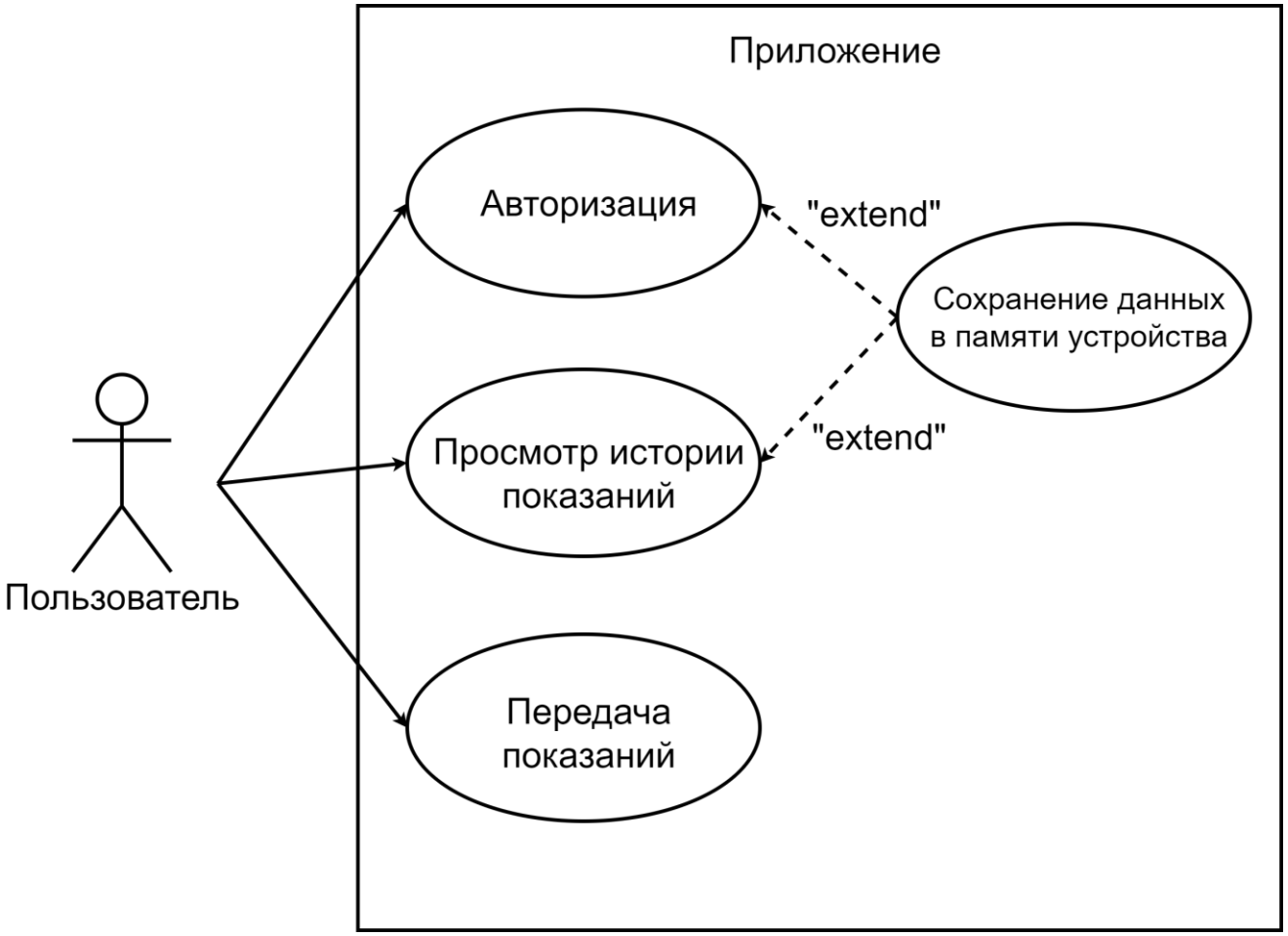


Рисунок 5 – Диаграмма вариантов использования

На диаграмме вариантов использования видно, что с приложением взаимодействует единственный пользователь, следовательно - все действия будут выполняться именно им посредством пользовательского интерфейса. Исходя из имеющихся данных построим диаграмму классов приложения.

Для реализации требуемых функций необходимо построить диаграмму последовательности.

Самой значимой функцией проектируемого приложения является передача показаний приборов учета, поэтому диаграмма последовательности действий будет реализована именно для неё.

Для того, чтобы передать показания приборов учета, требуется выполнить следующий алгоритм действия:

– пользователь переходит в меню ввода показаний приборов учета;

– приложение выводит список приборов учета;
– пользователь вводит показания приборов учета;
– пользователь нажимает на кнопку передачи показаний;
– приложение передает показания в базу данных.

Процесс передачи показаний приборов учета представлен на рисунке 6.

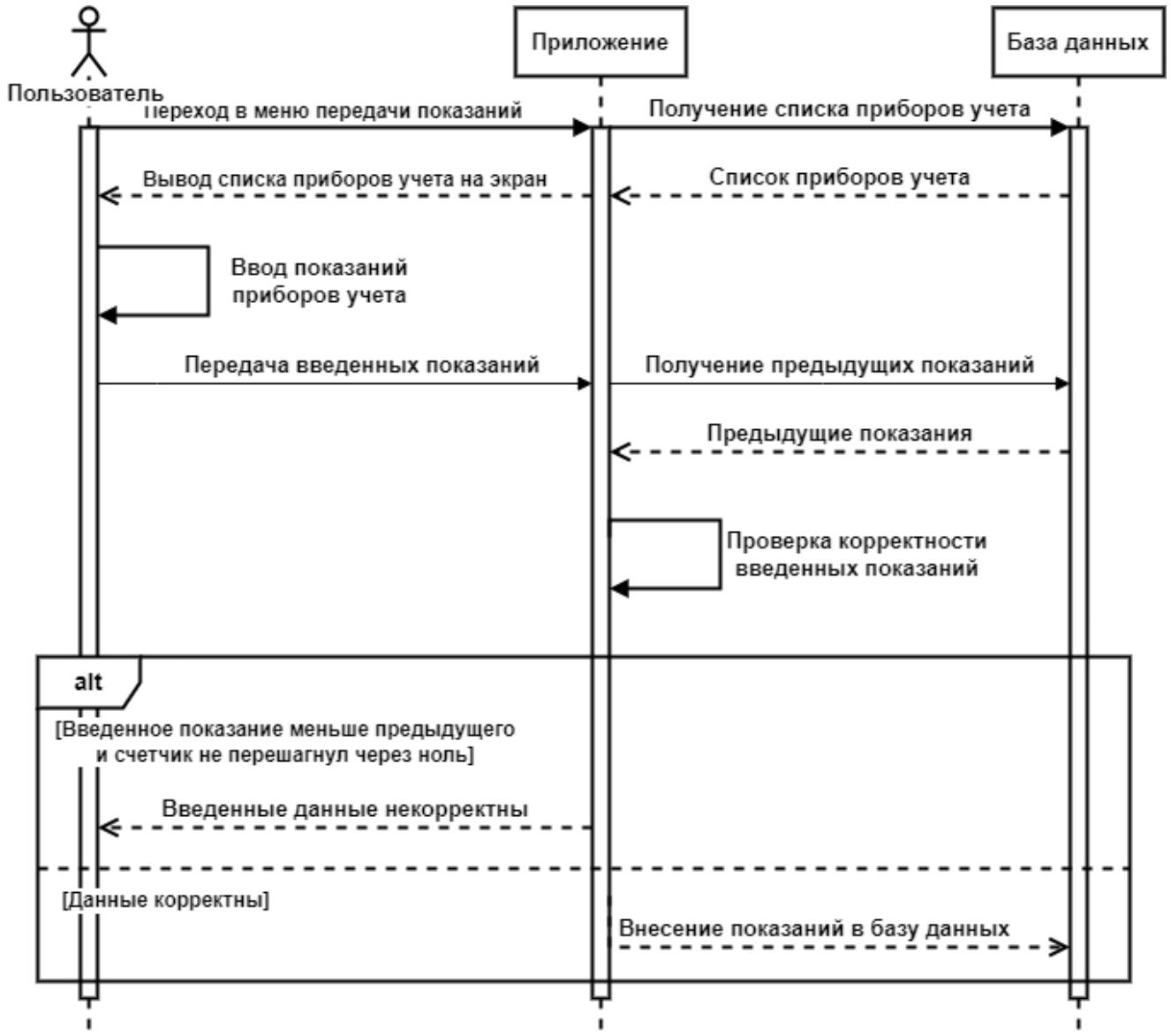


Рисунок 6 - Диаграмма последовательности действий при передаче показаний приборов учета

Диаграмма классов приложения – это графическое изображение

структуры классов в программном приложении. Она отображает классы и их отношения, а также атрибуты и методы, которые определены в каждом классе.

Диаграмма классов используется для документирования и понимания архитектуры приложения, а также для облегчения коммуникации между разработчиками и другими заинтересованными сторонами [7], [8]. Построенная диаграмма классов приложения представлена на рисунке 7.

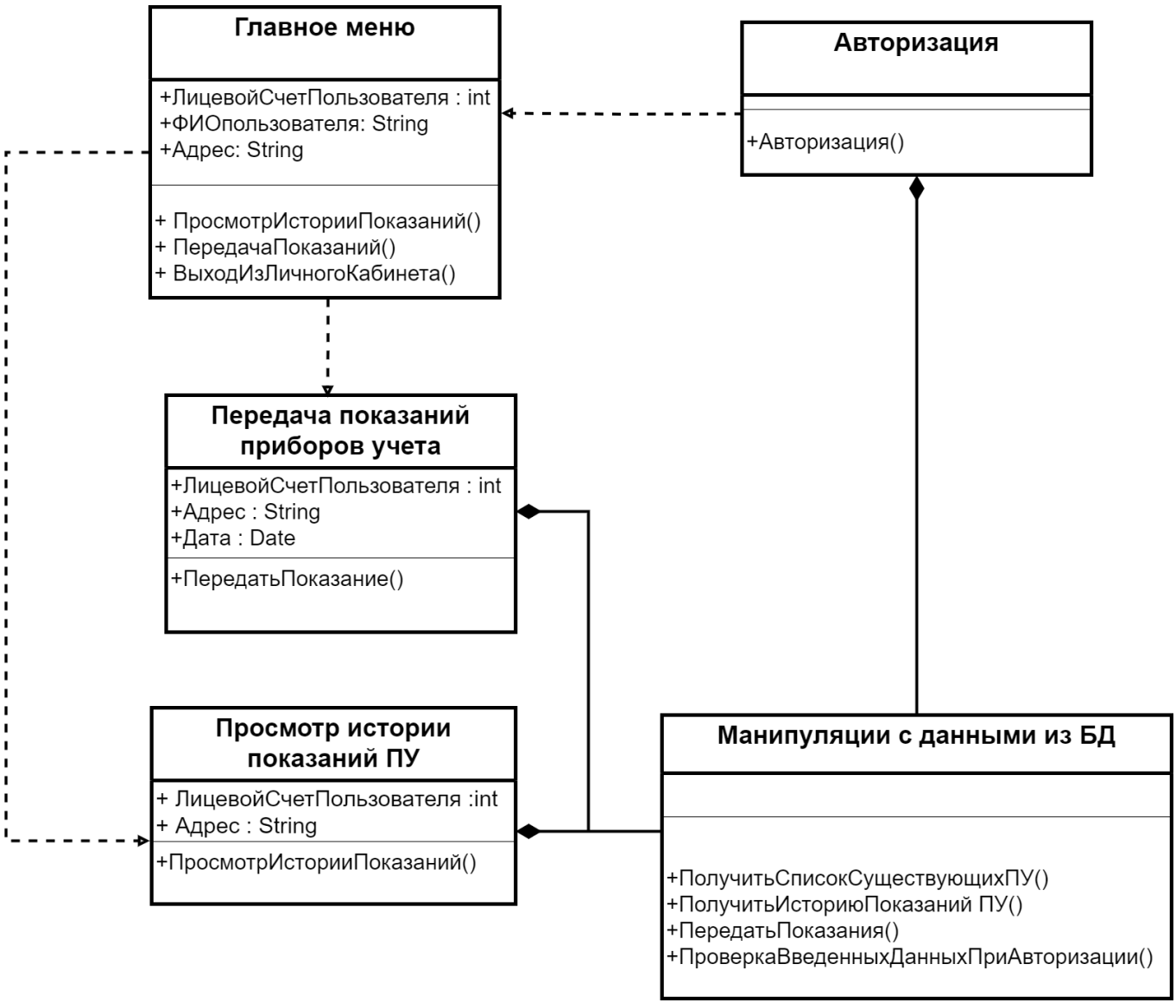


Рисунок 7 – Диаграмма классов приложения

## Выводы по главе 2

Вторая глава посвящена описанию архитектуры и моделированию разрабатываемого приложения.

В данной главе на основании данных об использовании операционных систем персональных компьютеров во всем мире было принято решение разрабатывать проектируемое приложение для ОС Windows.

Для реализации приложения была разработана архитектура «Клиент-сервер».

Также для реализации функционала приложения были реализованы диаграмма вариантов использования, диаграмма классов и диаграмма последовательности действий при передаче показаний приборов учета, описывающие функциональные возможности приложения.

# Глава 3 Разработка приложения для автоматизации сбора показаний приборов учета

## 3.1 Выбор средств реализации приложения

В задании от организации ООО «Квартплата 24» требуется использовать язык программирования Java - следовательно для реализации приложения будет использоваться именно он.

Java - это объектно-ориентированный язык программирования, разработанный компанией Sun Microsystems в 1995 году. Он стал очень популярным в короткие сроки из-за своей стабильности, надежности и кроссплатформенности. Так же он имеет широкий спектр применения и используется для разработки приложений на сервере, веб-сайтов, мобильных приложений, настольных приложений и игр [17].

Java является сильно типизированным языком, что означает, что каждый объект имеет свой тип данных. Это очень полезно при работе с большими программами, потому что это позволяет избежать ошибок, связанных с типами данных. Java также предоставляет возможность для создания GUI-приложений, используя пакет Swing, которая может реализовываться различными способами [17], [18], [19].

В качестве базы данных для тестирования будет использоваться PestgreSQL. PostgreSQL - это объектно-реляционная система управления базами данных (ORDBMS), которая использует язык SQL для работы с данными. Она была разработана в 1986 году в Университете Беркли в США и является бесплатным программным обеспечением с открытым исходным кодом. PostgreSQL предоставляет множество расширенных функций, таких как поддержка геоданных, масштабирование, поддержка транзакций и т.д. Он широко используется для создания профессиональных приложений в различных сферах, включая банковское дело, здравоохранение, науку, веб-разработку и другие области [16].

Важным этапов в реализации приложения является выбор интегрированной среды разработки. Интегрированная среда разработки (IDE) - это программное обеспечение, предназначенное для упрощения процесса разработки приложений. Она предоставляет программистам все необходимые инструменты для создания, редактирования, отладки и компиляции программного кода в одном окне.

IDE включает в себя текстовый редактор с подсветкой синтаксиса, автодополнением и функцией проверки ошибок, компиляторы, отладчики, средства управления проектом и другие инструменты, которые помогают повысить производительность программистов и ускорить процесс разработки. Среди популярных IDE для разработки приложений на языке Java существуют такие, как Eclipse, Intellij IDEA, NetBeans.

Eclipse - это самая популярная среда разработки для Java. Она бесплатна и обладает широкой базой пользователей и множеством плагинов.

IntelliJ IDEA - это платная среда разработки для Java, но она обладает мощным набором инструментов и функций, которые делают ее очень популярной среди профессионалов.

NetBeans - это бесплатная среда разработки и интегрированная среда разработки для Java EE. Она обладает открытым исходным кодом и интегрированной системой управления проектами.

Для того, чтобы выбрать подходящую среду для разработки приложения, проведем сравнение по следующим критериям:

– производительность – на сколько быстро работает IDE;

– совместимость плагинов и библиотек – на сколько корректно работают плагины и библиотеки в IDE;

– обновление документации – на сколько своевременно происходит обновление документации IDE;

– опыт использования – личный опыт работы в IDE.

Результаты субъективного сравнительного анализа сред разработки по выбранным критериям приведены в таблице 3.

Таблица 3 – Сравнительный анализ сред разработки по выбранным критериям

| Наименование IDE | Intellij IDEA | NetBeans | Eclipse |
|---|---|---|---|
| Производительность | 9 | 7 | 8 |
| Совместимость плагинов и библиотек | 10 | 7 | 8 |
| Обновление документации | 9 | 6 | 7 |
| Опыт использования | 6 | 0 | 0 |
| Итого: | 34 | 20 | 23 |

На основании приведенного сравнения было принято решение выбрать интегрированную среду Intellij IDEA для разработки приложения. IntelliJ IDEA – это интегрированная среда разработки (IDE), которая предназначена для создания приложений на языках программирования Java, Kotlin и др. Она разработана компанией JetBrains и является одной из наиболее популярных IDE для Java-разработки в мире [20].

IntelliJ IDEA предоставляет широкий набор инструментов и функций для упрощения и ускорения процесса разработки. Среди них:

- автодополнение кода;
- подсветка синтаксиса;
- рефакторинг кода;
- графический интерфейс для работы с Git;
- встроенная поддержка для работы с базами данных;
- компиляция и запуск приложений внутри IDE;
- инструменты для тестирования кода и отладки.

Одной из наиболее полезных функций IntelliJ IDEA является функция Live Templates, которая позволяет создавать шаблоны для часто используемого кода, такого как циклы, условия и подобное. Также, IntelliJ

IDEA предоставляет возможность установки и использования плагинов, которые расширяют функциональность IDE, делая ее еще более мощной и удобной для разработчиков [20].

### 3.2 Реализация приложения

Важным этапом в разработке приложений является создание пользовательского интерфейса. Пользовательский интерфейс (UI - от англ. User Interface) - это набор элементов и инструментов, которые позволяют пользователю взаимодействовать с компьютерной программой, устройством или другим техническим устройством. UI включает в себя все элементы, которые пользователь видит на экране или использует для управления приложением. К UI относятся элементы управления, такие как кнопки, текстовые поля, полосы прокрутки, выпадающие списки, чекбоксы, радиокнопки и т.д. UI также включает в себя различные формы, диалоговые окна, меню, иконки, сочетания клавиш и многие другие визуальные и функциональные элементы [30].

Цель UI заключается в том, чтобы обеспечить удобный и интуитивный интерфейс для пользователей. Хороший UI должен быть простым в использовании и надежным, удобным и безопасным для пользователя. Лучший UI является максимально интуитивным, поддерживает минимальное количество щелчков и нажатий, и обеспечивает быстрый и легкий доступ к необходимой информации [30].

UI разрабатывается на основе требований пользователей, индустрии, отрасли и требований, поставленных перед проектом. Он может быть разнообразным и сложным, но всегда должен быть легко использовать и понятен для пользователей. В языке программирования Java существуют несколько библиотек для создания графического интерфейса, и вот некоторые из них:

JavaFX - библиотека для создания интерактивных пользовательских

интерфейсов, которая была введена в Java в версии 8. Эта библиотека широко используется в различных приложениях Java и является стандартным инструментом для создания интерфейсов.

Swing - устаревшая библиотека для создания графических интерфейсов, которая была введена в Java в версии 1.2. Она все еще используется, но JavaFX сейчас является более современным и эффективным инструментом.

AWT (Abstract Window Toolkit) - это базовая библиотека для создания графического интерфейса, включенная в стандартную библиотеку Java. Она используется для создания простых приложений.

Библиотека JavaFX - это набор инструментов для создания графических пользовательских интерфейсов (GUI) для настольных приложений на языке Java. С JavaFX можно создавать приложения с отличным пользовательским интерфейсом и обеспечивать богатый пользовательский опыт. JavaFX была разработана для замены устаревшей и неудобной библиотеки Swing, и имеет уникальные возможности, такие как 3D графика, встроенные функции анимации, поддержку мультимедиа, интеграцию с языком макетов FXML и многие другие [25]. Нетрудно догадаться, что библиотека JavaFX является лучшим решением для разработки интерфейса приложений на Java, поэтому она была выбрана для создания приложения для данной выпускной квалификационной работы. Так же для создания интерфейса будет использоваться язык разметки FXML.

FXML (аббревиатура от "FXML Markup Language") - это язык разметки, используемый в JavaFX для описания графического интерфейса пользователя (GUI). Он позволяет создавать пользовательские интерфейсы на основе виджетов JavaFX, таких как кнопки, поля ввода, таблицы, списки и другие [25]. FXML-файлы представляют собой текстовые файлы в формате XML, в которых определяются иерархические структуры элементов интерфейса, а также их свойства и обработчики событий. Они могут быть созданы с помощью редакторов, таких как Scene Builder, который позволяет

визуально создавать UI и генерировать FXML-код. FXML также поддерживает использование специальных аннотаций, для внедрения элементов интерфейса, созданных в контроллере JavaFX, в соответствующие элементы интерфейса, созданные в файле FXML. Это повышает удобство обработки элементов управления и упрощает кодирование и обнаружение ошибок. Файлы FXML могут быть загружены и использованы в Java-коде с использованием класса FXMLLoader, который позволяет загружать FXML-файлы в виде объектов иерархии узлов Node, с возможностью добавления их в сцену JavaFX [25].

Таким образом для реализации графического интерфейса приложения будут использоваться FXML файлы. Для наглядной демонстрации будет приведен пример фрагмента кода создания главного меню приложения. Код FXML файла для создания интерфейса главного меню представлен на рисунке 8.

```
<Button layoutX="190.0" layoutY="211.0" mnemonicParsing="false" prefHeight="63.0"
        prefWidth="279.0" style="-fx-background-color: #FFFFFF; -fx-border-color:
        #140461;" text="Просмотр истории показаний">
   <font>
      <Font name="Arial Bold" size="16.0" />
   </font>
</Button>
<Button layoutX="190.0" layoutY="297.0" mnemonicParsing="false" prefHeight="63.0"
        prefWidth="279.0" style="-fx-background-color: #FFFFFF; -fx-border-color:
        #140461;" text="Передача показаний">
   <font>
      <Font name="Arial Bold" size="16.0" />
   </font>
</Button>
<Button layoutX="190.0" layoutY="383.0" mnemonicParsing="false" prefHeight="63.0"
        prefWidth="279.0" style="-fx-background-color: #FFFFFF; -fx-border-color:
        #140461;" text="Выход из личного кабинета">
   <font>
      <Font name="Arial Bold" size="16.0" />
   </font>
```

Рисунок 8 – Фрагмент содержимого FXML-файла на примере создания главного меню

Здесь описываются элементы(кнопки), находящиеся на ней, а также их параметры, такие как цвет, размер, шрифт и размер текста, расположение, и действия при нажатии. Всего создано две кнопки:

– «Просмотр истории показаний приборов учета»;

– «Передача показаний приборов учета».

При нажатии на каждую из них, на экране будут появляться другие меню с возможностями, соответствующими их назначению, например - при нажатии на кнопку передачи показаний приборов учета откроется меню с возможностью откроется меню с соответствующей возможностью.

Чтобы при нажатии на одну из кнопок выполнялось какое-либо действие, требуется создать специальный класс-контроллер. Классы-контроллеры в JavaFX являются главными компонентами для управления в пользовательском интерфейсе. Они используются для связи представления и модели, и управления элементами управления на форме.

Каждый класс-контроллер обрабатывает определенную форму, которая отображает пользовательский интерфейс. Он содержит логику, необходимую для обработки пользовательских действий, и предоставляет информацию о данных, отображаемых на форме.

Для создания класса-контроллера нужно добавить аннотацию @FXML перед методом-обработчиком для каждого элемента управления на форме. Это позволяет связать метод с соответствующим элементом управления и событием действия пользователя.

Когда пользователь выполняет действие на форме, соответствующий метод-обработчик срабатывает и выполняет необходимые действия.

Фрагмент кода класса-контроллера с методами-обработчиками для главного меню приложения представлен на рисунке 9.

```
1 usage
@FXML
public void handleCheckReadingHistoryButtonClick(ActionEvent event) {
    changeMenuAfterHandlingButtonClick( FXMLFileName: "checkMeterReadingsHistoryMenu.fxml",event);
}
1 usage
@FXML
public void handleSendMeterReadingsButtonClick(ActionEvent event) {
    changeMenuAfterHandlingButtonClick( FXMLFileName: "sendMeterReadingsMenu.fxml",event);
}
2 usages
public void changeMenuAfterHandlingButtonClick(String FXMLFileName, ActionEvent event){
    try{
    FXMLLoader loader = new FXMLLoader(getClass().getResource(FXMLFileName));
    Stage stage = (Stage) ((Node) event.getSource()).getScene().getWindow();
    stage.setScene(new Scene(loader.load()));
    stage.show();
} catch(
IOException e)
{
    e.printStackTrace();
}
```

Рисунок 9 – Фрагмент кода обработчиков событий в классе-контроллере

Здесь представлены два метода-обработчика и каждый из них выполняется после нажатия на определенную кнопку в интерфейсе главного меню. Так как в каждом из них выполняются одинаковые действия – открытие нового меню, то был создан еще один метод, куда передается название FXML-файла, где прописан код соответствующей сцены, которая должна появляться на экране приложения после нажатия соответствующей кнопки. Для остальных меню данного приложения интерфейс будет так же создан при помощи вышеописанного способа.

Функционал разрабатываемого приложения основан на взаимодействии с базой данных – передаче и получении информации, содержащейся в ней, и этот процесс будет происходить при помощи SQL-запросов. SQL (Structured Query Language) - язык запросов, который используется для работы с базами данных. SQL-запросы позволяют выбирать, изменять, добавлять и удалять данные из таблиц базы данных [6].

Также для доступа пользователя к функциям приложения будет использована технология авторизации с помощью пароля.

Технология авторизации при помощи логина и пароля является одним из наиболее распространенных механизмов защиты данных в Интернете. Она позволяет пользователям пройти процедуру аутентификации на веб-сайте или в приложении, используя уникальную комбинацию логина и пароля. Когда пользователь вводит логин и пароль на странице авторизации, эти данные отправляются на сервер для проверки.

В случае, если введенная информация соответствует записанным в базе данных учетным данным, пользователь получает доступ к защищенной зоне веб-сайта или приложения [26].

Ключевым моментом в технологии авторизации при помощи логина и пароля является безопасность.

Так как логин и пароль - уникальная идентификационная комбинация, то хранилище учетных данных должно быть достаточно надежным, чтобы ограничить доступ к ним со стороны злоумышленников.

Технология авторизации при помощи логина и пароля является одним из основных механизмов защиты данных в Интернете и позволяет пользователям получить доступ к защищенным ресурсам.

Увеличение безопасности используемых паролей поможет снизить риски несанкционированного доступа к учетным данным.

### 3.3 Тестирование разработанного приложения

Тестирование разработанного приложения будет проводиться при помощи метода функционального тестирования.

Функциональное тестирование приложения - это процесс проверки функциональности программного обеспечения для того, чтобы убедиться, что оно работает корректно и соответствует требованиям пользователя и спецификации продукта.

В рамках функционального тестирования тестируются все функции и свойства приложения, включая его взаимодействие с пользователем, работу с данными и т.д.

В процессе функционального тестирование будет произведено тестирование всех основных функций приложения:

- авторизация;
- добавление нового показания;
- просмотр истории показаний.

При открытии приложения пользователю требуется пройти процедуру авторизации. Для этого в соответствующие поля нужно ввести номер лицевого счета пользователя и пароль от личного кабинета.

Меню авторизации в приложении представлено на рисунке 10.

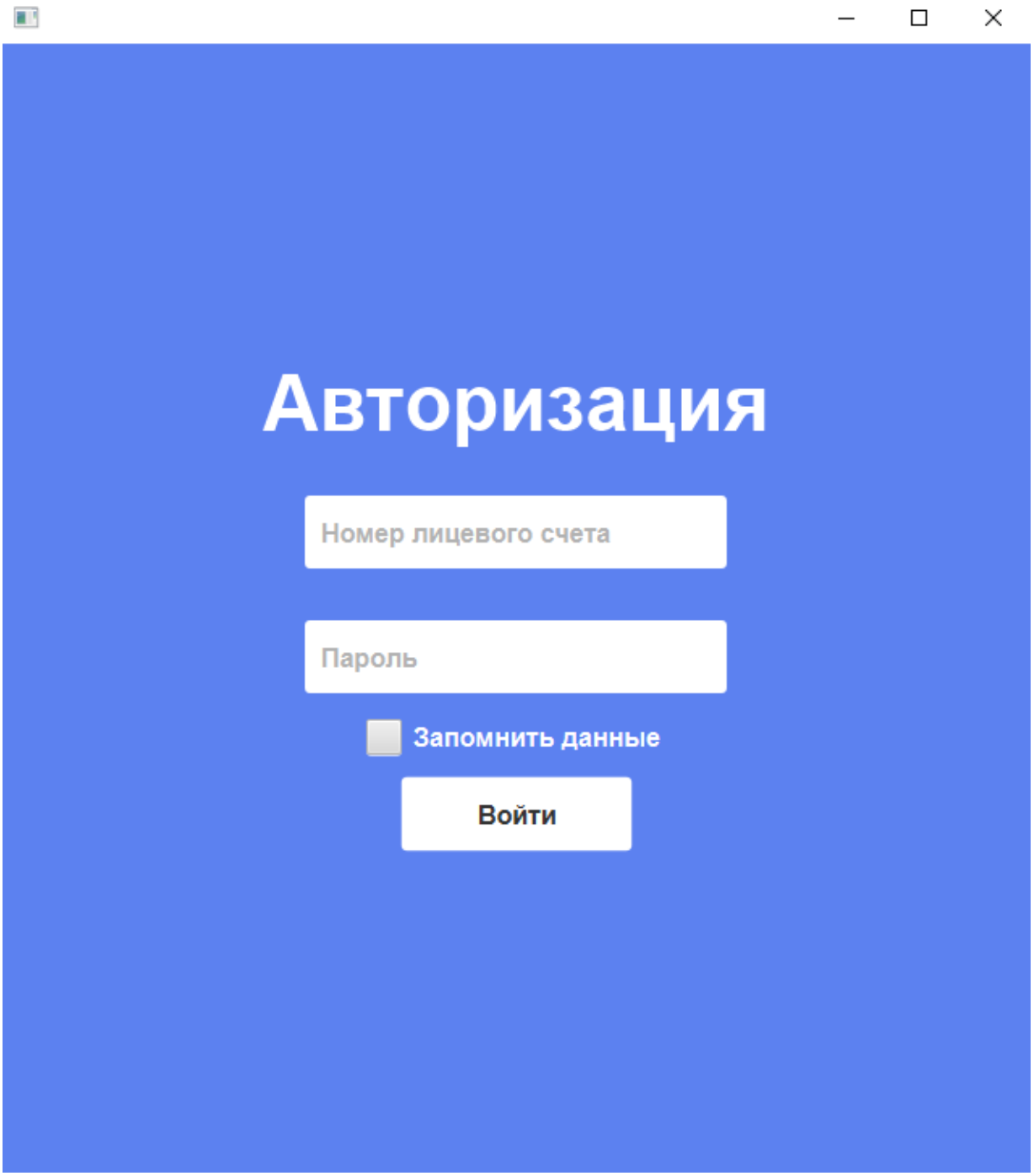


Рисунок 10 – Меню авторизации пользователя в приложении

После того, как пользователь успешно авторизовался, откроется главное меню приложения, в котором он сможет выбрать дальнейшие действия – просмотреть историю показаний приборов учета, передать новые показания, либо выйти из личного кабинета. Главное меню представлено на рисунке 11.

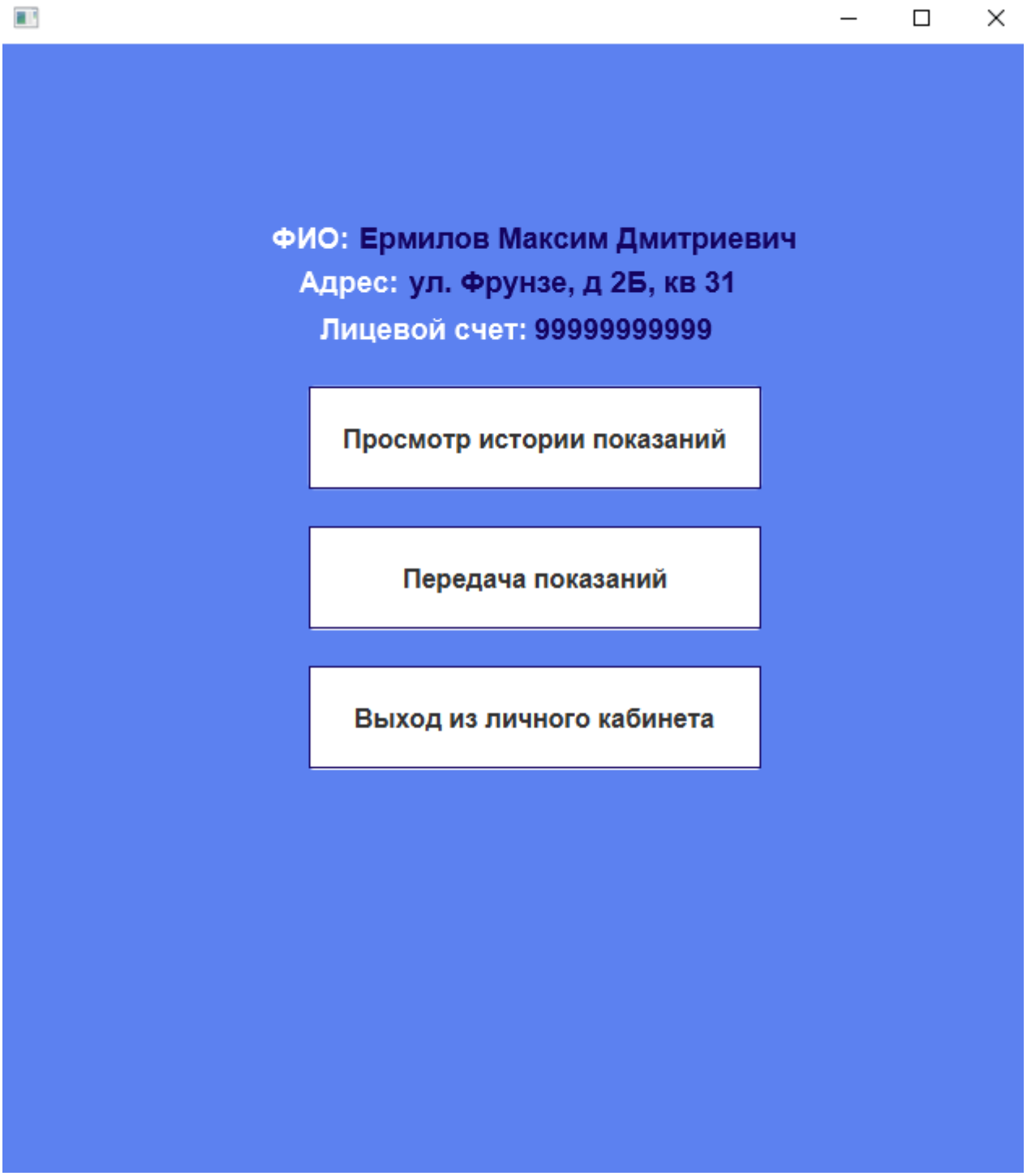


Рисунок 11 – Главное меню приложения

Для проверки возможности передачи показаний требуется перейти в меню передачи показаний приборов учета, интерфейс которого представлен на рисунке 12. Здесь представлены данные адреса пользователя, его лицевого счета, предыдущих показаний, последних показаний, а также имеется

возможность ввести новое показание и передать его управляющей компании.

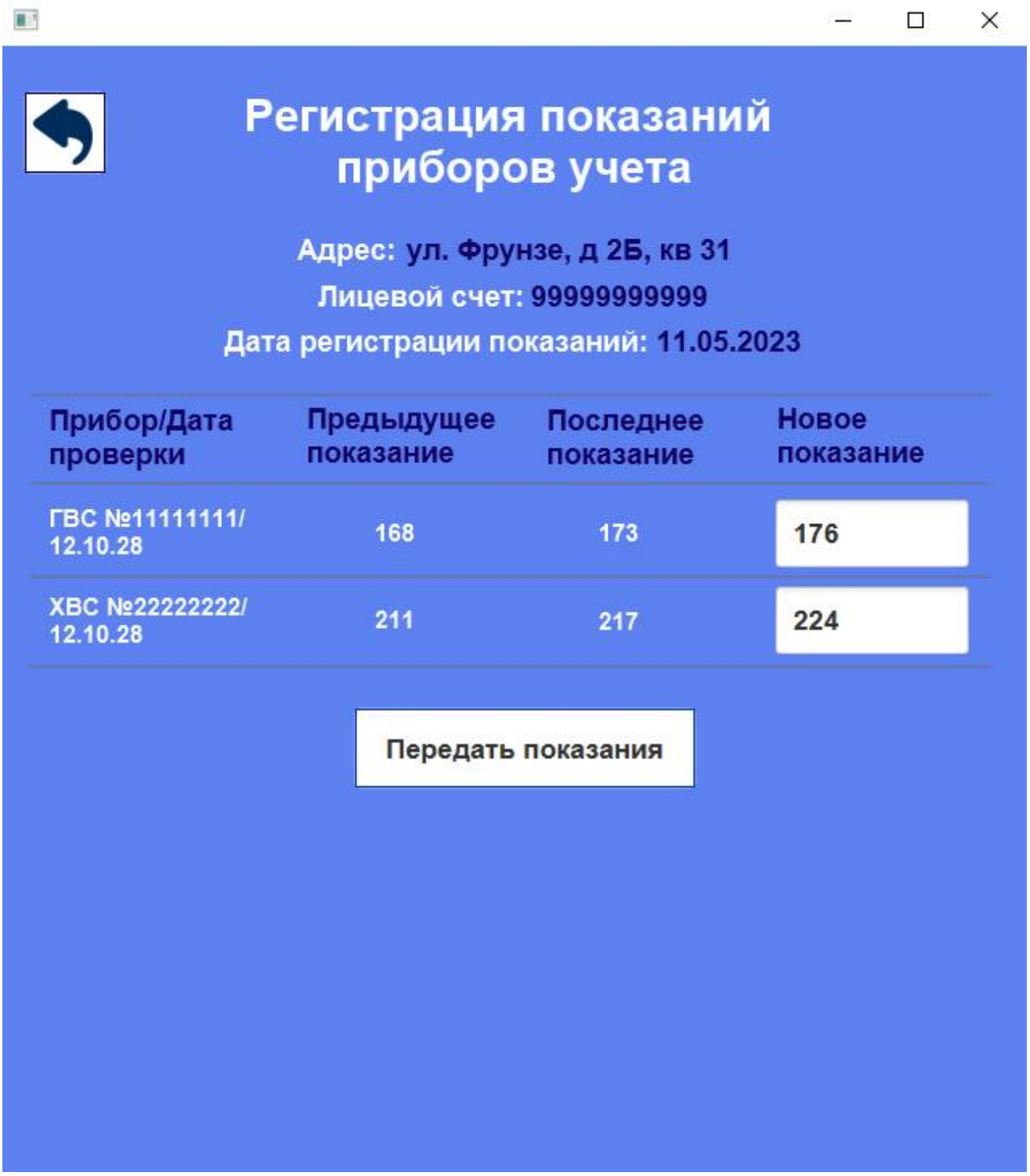


Рисунок 12 – Меню передачи показаний приборов учета

После нажатия кнопки передачи показаний данные будут отправлены в базу данных. Чтобы проверить, действительно ли они были отправлены,

вернемся в главную меню, нажав на кнопку в правом-верхнем углу экрана и перейдя в меню просмотра показаний приборов учета. Его интерфейс изображен на рисунке 13.


Рисунок 13 – Интерфейс меню просмотра истории показаний приборов учета

Как мы видим, данные действительно были переданы. Также в данном меню представлены показания приборов учета за предыдущие месяцы. Таким образом весь функционал приложения был успешно протестирован и его работоспособность подтверждена по всем приведенным пунктам.

## Выводы по 3 главе

В данной главе был проведен анализ существующих интегрированных средств разработки, и на его основании для реализации приложения была выбрана наиболее подходящая среда - IntelliJ IDEA.

Также была выбрана наиболее современная и имеющая обширный инструментарий библиотека для реализации графического интерфейса приложения – JavaFX.

С помощью выбранных технологий было разработано приложение с указанным функционалом.

Тестирование приложения показало, что все реализованные функции работают корректно, следовательно - работоспособность приложения успешно подтверждена.

# Заключение

Выпускная квалификационная работа посвящена разработке приложения для автоматизации сбора показаний приборов учета.

В ходе выполнения ВКР был проведен анализ существующих в организации ООО «Квартплата 24» способов автоматизации сбора показаний приборов учета.

Было произведено сравнение существующих аналогов разрабатываемого приложения, в результате которого был сделан вывод, что выбранные сервисы являются достойными аналогами по функционалу, но у них отсутствует приложение для ПК, что свидетельствует о целесообразности разработки ПО в рамках данной ВКР.

Для разрабатываемого приложения были выбраны требования на основании методологии FURPS+ и задания заказчика, а также указан приоритет реализации каждого из требований.

Для реализации приложения была выбрана операционная система Windows, так как она является самой используемой ОС в мире. Была разработана архитектура приложения - «Клиент-сервер», составлена диаграмма компонентов, описаны варианты использования приложения при помощи диаграммы вариантов использования и классов приложения, и описан алгоритм действий при использовании функции передачи показаний приборов учета.

Далее был проведен анализ существующих интегрированных сред разработки приложений на языке Java, и на его основании принято решение разрабатывать приложение в IntelliJ IDEA. Для реализации пользовательского интерфейса была выбрана современная библиотека JavaFX, предоставляющая мощный API для взаимодействия пользователя с интерфейсом и обработки событий.

После выбора описанного инструментария с помощью языка программирования Java и СУБД Postgres были реализованы все требуемые функции приложения: Добавление/удаление адреса, добавление прибора

учета к адресу, изменение типа прибора учета, добавление нового показания, просмотр истории показаний.

С использованием библиотеки JavaFX и языка разметки FXML был реализован интерфейс приложения со всеми необходимыми элементами для работы функционала приложения элементами.

Также разработанное приложение было протестировано методом функционального тестирования, который подразумевает проверку работоспособности всех реализованных функций. В итоге приложение успешно прошло тестирование по всем пунктам.

Задачи для достижения цели выпускной квалификационной работы пыли в полном объеме выполнены процессе её создания.

Цель работы достигнута – разработано приложение для автоматизации сбора показаний приборов учета.

В ходе работы над ВКР были получены знания и навыки работы с базами данных Postgres, и улучшены навыки программирования на языке Java.

Разработанное приложение может быть доработано и использовано в качестве метода автоматизации сбора показаний приборов учета для компаний ЖКХ.

## Список используемой литературы


1. Бейли Л. Изучаем SQL. СПБ.: Питер, 2012. – 573с.
2. Вигерс К. Разработка требований к программному обеспечению. 3-е изд., дополнительное / К. Вигерс, Д. Битти М.: Издательство «Русская редакция»; СПб.: БХВ-Петербург, 2014. – 736 стр.
3. Гагарина Л. Г., Киселев Д. В., Федотова Е. Л. Разработка и эксплуатация автоматизированных информационных систем М.: ИД «ФОРУМ»; ИНФРА-М, 2012. – 384с.
4. Голицина О.Л., Максимов Н.В., Попов И.И. Базы данных: Учебное пособие. – М.: Формум: ИНФРА-М, 2013. – 352 с.
5. Грофф Д. SQL: Полное руководство / Д. Грофф, П. Вайнберг. / - К.: BHV. 2001. – 816с.
6. Грекул, В. И. Проектирование информационных систем : учебное пособие / В. И. Грекул, Г. Н. Денищенко, Н. Л. Коровкина. — 3-е изд. — Москва : Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2020. — 299 с.
7. Дейт Д. Введение в системы баз данных. М.: Издательский дом "Вильяме". 2005. – 1328 с.
8. Емельянова Н.З. Проектирование информационных систем: учебное пособие / Н.З. Емельянова, Т.Л. Партыка, И.И. Попов. М.: Форум. 2014. – 432 с.
9. Заботина Н. Н. Проектирование информационных систем - М.: ДРОФА, 2013. – 336 с.
10. Колесов Ю. Б. Моделирование систем. Объектно-ориентированный подход: учебное пособие / Ю. Б. Колесов, Ю. Б. Сениченков. СПб.: БХВПетербург. 2012. – 192 с.
11. Конноли Т. Базы данных. Проектирование, реализация и сопровождение. Теория и практика / Т. Конноли, К. Бегг, А. Стратчан. М.: Вильямс. 2000. – 1093 с.

12. Корнеев И. К. Информационные технологии: учебное пособие. М: Проспект. 2007. – 224 с.

13. Куроуз Д. Компьютерные сети. Многоуровневая архитектура Интернета: 2-е издание / Д. Куроуз, К. Росс. СПб.: Питер. 2004. – 765с.

14. Хомоненко А.Д., Цыганков В.М., Мальцев М.Г. Базы данных: Издание второе, дополненное и переработанное. М. 2012. – 672 с.

15. Зыков С. В. Проектирование и разработка корпоративных информационных систем : учебное пособие / С. В. Зыков. — Москва : Ай Пи Ар Медиа, 2023. — 394 с.

16. Джуба С., Волков А. Изучаем PostgreSQL 10: ДМК Пресс, 2018. – 400с.

17. Герберт Шилдт Java: Полное руководство, Десятое издание: Диалектика-Вильямс, 2018. – 1488 с.

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

19. Deitel, H. Java How to Program / H. Deitel, P. Deitel. – 9th edition, Prentice Hall. – 2015

20. Hudson O. Getting started with IntelliJ IDEA // O. Hudson, Birmingham: Packt Publishing, 2013. – 114p.

21. Kaner, Falk, Nguyen. Testing Computer Software. – USA: Wiley

22. Computer Publishing, 1999. – 42 p.

23. Reto Meier. Professional Android 4 Application Development. / Reto Meier - Wrox, 2012. – 864 p.

24. Krochmalski J. IntelliJ IDEA Essentials // J. Krochmalski. – Birmingham: Packt Publishing, 2014. – 263p.

25. Прохоренок Н. JavaFX. В подлиннике: БХВ-Петербург, 2020. 768с.

26. Смит Р. Аутентификация: от паролей до открытых ключей: Издательский дом «Вильямс», 2002. – 424с.

27. Колесов Ю. Б. Моделирование систем. Объектно-ориентированный подход: учебное пособие / Ю. Б. Колесов, Ю. Б. Сениченков. СПб.: БХВ-Петербург, 2012. – 192 с.

28. Filipiuk M. UI Design Principles, 2021. – 323c

29. Malewicz M, Malewicz D. Designing User Interface, 2021. – 419c.

30. Купер А. Интерфейс. Основы проектирования взаимодействия. 4-е издание: Питер, 2021. – 722с.