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

Кафедра «Прикладная математика и информатика»
(наименование)

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

Разработка социальных и экономических информационных систем
(направленность (профиль) / специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему Разработка сервиса передачи и проверки введенных показаний приборов учета коммунальных услуг

Обучающийся А. Л. Дудин _____
(Инициалы Фамилия) (личная подпись)

Руководитель канд.пед.наук, доцент, О.М. Гущина
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

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

Тольятти 2024

Аннотация

Выпускная квалификационная работа состоит из разработки сервиса передачи и проверки введенных показаний приборов учета коммунальных услуг. Сервис представляет собой программное решение, разработанное для упрощения и автоматизации процесса передачи и проверки данных о потреблении коммунальных ресурсов.

Структура работы включает в себя введение, три главы, заключение, список литературы и приложение.

Цель данного сервиса – обеспечение удобного и безопасного механизма передачи показаний приборов учета конечными пользователями, а также автоматизации процесса проверки этих данных для обеспечения точности и достоверности информации о потреблении коммунальных услуг.

Во введении данной работы обозначены цели и задачи, определена актуальность исследования, а также выделены объект и предмет исследования.

В первой главе проводится общая характеристика области ЖКХ, которая включает разработку диаграмм, отражающих аспекты данной сферы. Во второй главе формулируются требования к разрабатываемому приложению, определяется технология разработки, создаются логические модели и разрабатывается прототип будущего интерфейса. В третьей главе описана архитектура веб-приложения, разработка алгоритма передачи показаний с использованием выбранных инструментов для разработки, представление разработанного приложения и проведение тестирования.

В заключении описаны выводи и результаты выполненной работы. Итогом ВКР является рабочее веб-приложение для передачи и проверки показаний приборов учета, способствующее повышению удобства в процессах контроля за расходом коммунальных ресурсов собственников.

В работе показано 17 рисунков, 2 таблицы, список использованной литературы содержит 26 источников. Целый объем ВКР содержит 53 страницы.

Abstract

The final qualifying work consists of developing a transmission service and checking the entered readings from utility metering devices. The service is a software solution designed to simplify and automate the process of transferring and verifying data on the consumption of utility resources.

The structure of the work includes an introduction, three chapters, a conclusion, a list of references and an appendix.

The purpose of this service is to provide a convenient and secure mechanism for transmitting meter readings by end users, as well as automating the process of verifying this data to ensure the accuracy and reliability of information on utility consumption.

In the introduction of this work, the goals and objectives are outlined, the relevance of the research is determined, and the object and subject of the research are highlighted.

The first chapter provides a general description of the housing and public utilities sector, which includes the development of diagrams reflecting aspects of this area. In the second chapter, the requirements for the application being developed are formulated, the development technology is determined, logical models are created and a prototype of the future interface is developed. The third chapter describes the architecture of the web application, the development of an algorithm for transmitting readings using selected development tools, the presentation of the developed application and testing.

In conclusion, the conclusions and results of the work performed are described. The result of the VKR is a working web application for transmitting and checking meter readings, which helps to increase convenience in the processes of monitoring the consumption of utility resources of owners.

The work shows 17 figures, 2 tables, the list of used literature contains 26 sources. The entire volume of the WRC contains 53 pages.

Оглавление

Введение

В современном обществе одним из наиболее важных направлений социально-экономических изменений является реформирование и развитие жилищно-коммунальной сферы, создающей условия для полноценной жизнедеятельности человека. Важные компоненты этой сферы – жилищное хозяйство и коммунальное обслуживание, обеспечивающие содержание жилищного фонда и предоставление жилищно-коммунальных услуг конечным потребителям.

Развитие информационных технологий в жилищно-коммунальном сервисе становится приоритетным направлением государственной политики в России. Реформы в жилищно-коммунальном хозяйстве (ЖКХ) привели к изменению структуры собственности муниципального фонда. Новая политика формирует класс собственников имущества, вызывая проблемы в обслуживании и управлении [2].

Трансформация российской экономики обусловила необходимость формирования нового хозяйственного механизма в жилищно-коммунальном хозяйстве. Преобразования направлены на вхождение этой сферы в рыночную среду, предоставляя населению высококачественные жилищные и коммунальные услуги. Данные услуги играют важную роль в обеспечении комфорта проживания и поддержании благоприятного состояния городов.

В условиях сложной коммерческой деятельности предприятий и организаций ЖКХ, учет и контроль бизнес-процессов, а также объединение данных в единой информационной системе становятся востребованными. Автоматизация управления многоквартирными домами достигается разработкой и внедрением программы, предназначенной для использования в организациях ЖКХ, осуществляющих учет квартплаты и прочих коммунальных услуг.

Цель выпускной квалификационной работы заключается в создании программного обеспечения для учета и контроля коммунальных платежей в

ЖКХ, учитывая недостаточное изучение практических вопросов учета и контроля расчетов за эти услуги.

Объектом исследования данной работы является жилищно-коммунальная сфера, а именно процессы передачи и проверки показаний приборов учета в рамках этой сферы.

Предметом исследования является разработка программного сервиса, предназначенного для учета и проверки показаний приборов учета в жилищно-коммунальном хозяйстве.

Для достижения поставленной цели необходимо будет выполнить следующие задачи:

- изучить существующие бизнес-процессы жилищно-коммунальной сферы;
- проанализировать существующие решения в области передачи и проверки показаний приборов учета;
- разработать логическую модель веб-приложения, определяющую основные компоненты и их взаимосвязи;
- выбрать оптимальные технологии для разработки веб-приложения;
- разработать и реализовать программное обеспечение для передачи показаний приборов учета в центральную систему;
- протестировать разработанное решение и оценить эффективность его работы.

Таким образом, тема выпускной квалификационной работы представляет актуальную и перспективную область разработки, требующую внимания к практическим аспектам учета и контроля платежей за жилищно-коммунальные услуги.

Глава 1 Анализ области жилищно-коммунального хозяйства

1.1 Общая характеристика ЖКХ

Жилищно-коммунальное хозяйство (ЖКХ) – область экономики, занимающаяся обеспечением функционирования зданий, для безопасного и комфортного проживания и нахождения в них людей [6].

Основная задача – эффективное использование средств собственников, а также обеспечение жилищно-коммунальными услугами, необходимыми для комфортной жизни человека.

Коммунальное хозяйство представляет собой комплекс предприятий, служб и хозяйств, занимающихся обслуживанием населения городов, посёлков и сёл, а также промышленных предприятий [14]. Эти организации поставляют воду, электроэнергию и газ тем объектам, которые не имеют своих собственных сооружений коммунального назначения. Уровень развития и объем деятельности коммунального хозяйства напрямую воздействуют на благосостояние населения, условия его быта, санитарно-гигиеническую обстановку и состояние водных и воздушных бассейнов. Эти факторы также оказывают влияние на производительность труда.

Объектами коммунального назначения в сфере жилищно-коммунального хозяйства (ЖКХ) являются водопровод, канализация, капитальный ремонт помещений, текущий ремонт зданий, теплоснабжение, уборка и утилизация мусора, электроснабжение. Характеристика этих объектов подчеркивает сложность и многообразие выполняемых задач в сфере ЖКХ. В связи с высокой степенью неудовлетворенности населения качеством предоставляемых услуг, особое значение приобретает реформа жилищно-коммунального хозяйства, направленная на улучшение качества обслуживания населения.

Для успешного функционирования современного населенного пункта деятельность ЖКХ – одна из основных и сложных задач городского

управления.

Жилищно-коммунальное хозяйство включает в себя:

а) инженерно-коммунальные системы:

1) водоснабжение,
2) водоотведение,
3) теплоснабжение;

б) энергоснабжение:

1) газоснабжение,
2) электроснабжение;

в) поддержание благоустройства территории:

1) озеленение,
2) утилизация отходов;

г) обслуживание жилищного хозяйства:

1) капитальный ремонт зданий и сооружений,
2) технический ремонт зданий и сооружений.

Водоснабжение обеспечивает население чистой и пресной водой, создавая и обслуживая водопроводные системы и насосные станции, что гарантирует бесперебойное поступление воды в дома граждан. Отвод и очистка сточных вод является неотъемлемой частью системы водоотведения, включающей в себя канализационные системы, очистные сооружения, а также технологии и оборудование для предварительной очистки сточных вод, прежде чем они возвращаются в природную среду. Теплоснабжение гарантирует поддержку необходимого уровня тепла в холодный период, что охватывает создание и поддержание систем отопления, установку и обслуживание тепловых насосов и котелен, а также прочих элементов инфраструктуры по теплоснабжению. Газоснабжение включает в себя разработку и поддержку систем подачи природного газа, что обеспечивает его доступность для бытовых и промышленных целей.

Электроснабжение занимается созданием и поддержанием систем, обеспечивающих подачу электричества, включая обслуживание

электростанций и распределительных сетей для надёжной трансмиссии электроэнергии к потребителям как в жилых домах, так и на предприятиях. Озеленение играет важную роль в повышении качества экологической среды, включая создание и поддержание парковых зон и аллей, посадку и уход за растительностью, а также улучшение ландшафтного дизайна города. Утилизация отходов отвечает за эффективное собирание, транспортировку и обработку бытовых и промышленных отходов, что влечет за собой создание и эксплуатацию систем сортировки и мусороперерабатывающих заводов, а также контроля за процессом утилизации.

Капитальный ремонт зданий и сооружений, необходимый для восстановления и обновления крупных структурных элементов инфраструктуры, укрепляет фундамент и способствует улучшению долгосрочной устойчивости и безопасности здания. Технический ремонт зданий и сооружений направлен на регулярную замену изношенных деталей, исправление мелких повреждений и поддержание систем видеонаблюдения и связи, что обеспечивает безопасность, контроль доступа и функционирование всех систем многоквартирных домов.

Таким образом, после описания деятельности ЖКХ, представим её общий вид на рисунке 1.

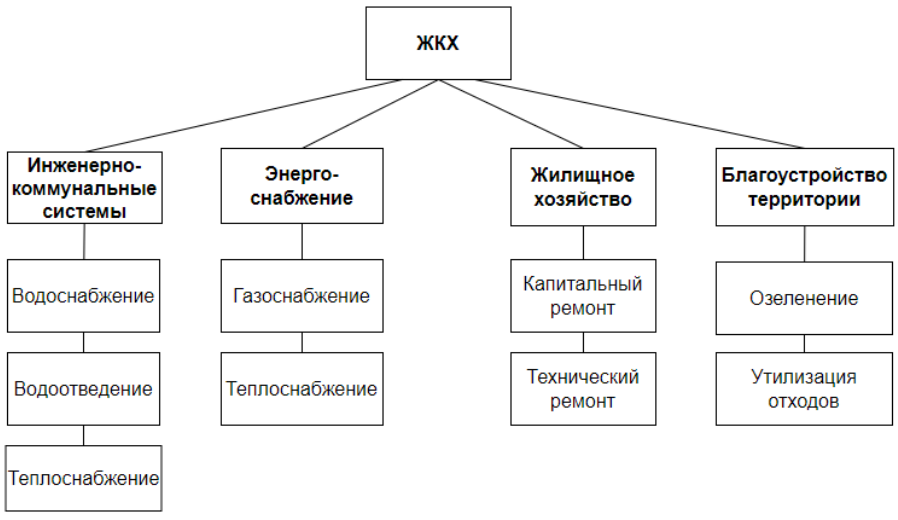


Рисунок 1 - Схема деятельности ЖКХ

Решением этих задач занимается ЖКХ, и каждая из них имеет абсолютную необходимость для комфортной жизни населения.

1.2 Разработка и анализ модели бизнес-процесса «Как есть»

Модель «Как есть» олицетворяет состояние бизнес-процесса, не имеющего информационной системы для упрощения передачи показаний приборов учета [4]. Для моделирования бизнес-процесса было решено использовать нотацию BPMN. Её главной задачей является возможность смоделировать бизнес-процесс так, чтобы он был прост в понимании для аналитиков или разработчиков [1]. Смоделированная диаграмма бизнес-процесса изображена на рисунке 2.

Рассмотрим порядок бизнес-процессов внутри диаграммы:

- процесс передачи показаний начинается с получения собственником уведомления о необходимости передачи показаний приборов учета;
- далее, собственник заполняет бумажную форму, вводя показания, которые отправляет, или приносит лично в управляющую компанию;
- данные показания проверяются сотрудником управляющей компании;
- в случае, если показания некорректны, сотрудник уведомляет об этом собственника. В этом случае, собственнику придется заново заполнять бумажную форму, и передать ее в управляющую компанию;
- в случае, если показания корректны, сотрудник вносит эти показания в базу данных;
- далее происходит составление отчета о потреблении, необходимого для анализа потребления ресурсов;
- заключением всех этих процессов, является составление квитанции о расчете, и отправка этой квитанции собственнику.

Получение собственником данной квитанции является подтверждением

того, что его показания были заполнены верно.

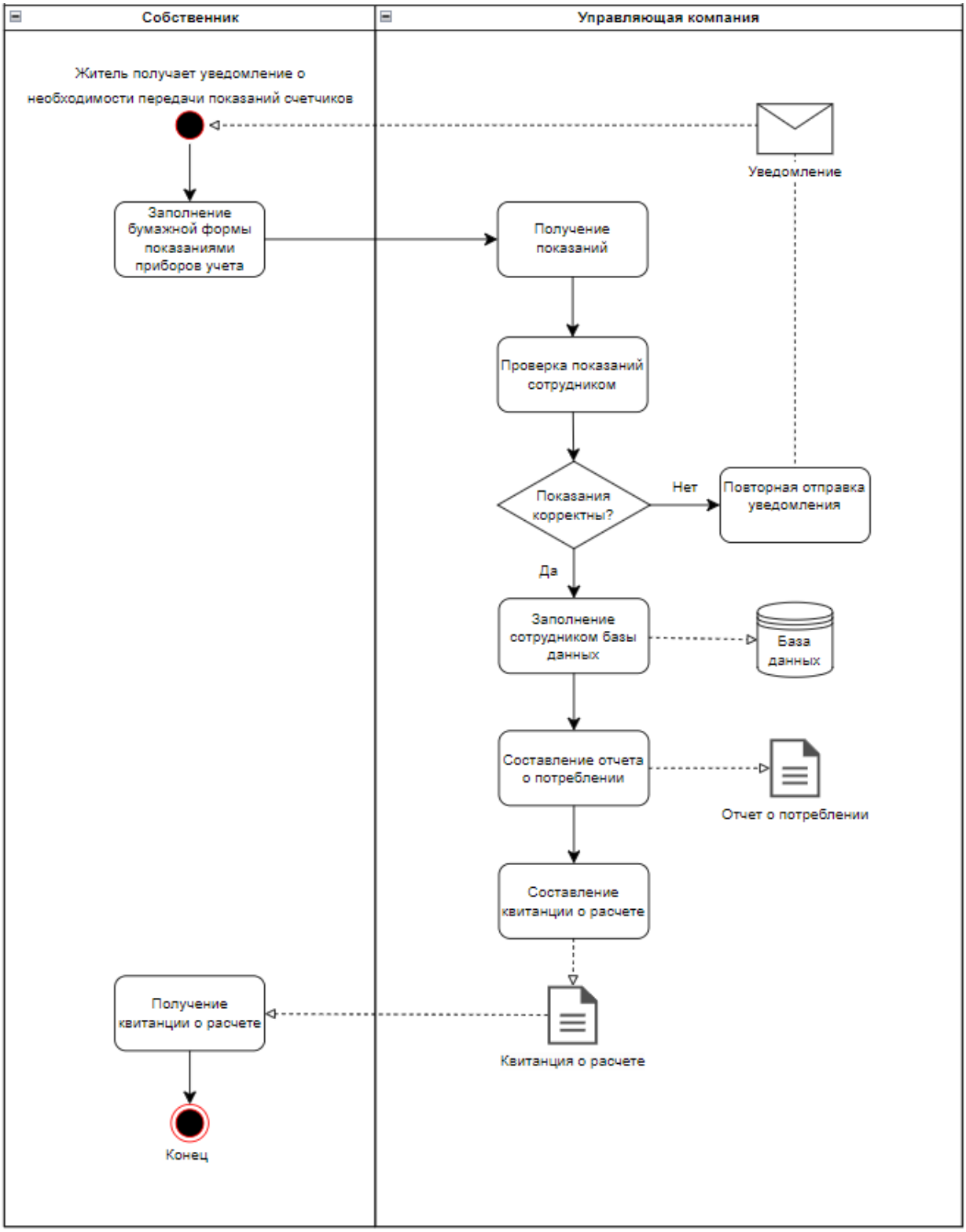


Рисунок 2 – Диаграмма бизнес-процесса «КАК ЕСТЬ»

В этой модели процесс сбора и передачи данных осуществляется вручную или с использованием устаревших методов и инструментов. Отсутствие автоматизации процесса может привести к ряду проблем и ограничений, таких как:

- ошибки в данных, происходящие по причине ручного ввода показаний, подверженного человеческим ошибкам и невозможностью своевременной проверки. Это может привести к неточным данным и недостоверным отчетам о потреблении ресурсов [8];
- медлительность процесса, объясняющаяся тем, что, ручной процесс сбора и передачи данных требует дополнительного времени и труда на обработку данной информации;
- ограниченная аналитика, а именно отсутствие централизованной системы сбора и анализа данных, затрудняющее проведение анализа потребления ресурсов и выявление тенденций, что может затруднить принятие информированных решений по управлению ресурсами;
- недостаточная прозрачность, заключающаяся в отсутствии автоматизированной системы, приводящее к недостаточной прозрачности процесса передачи данных. Это может вызвать недоверие и конфликты между сторонами.

Таким образом, данная схема бизнес-процесса несовершенна, так как процессы передачи и обработки показаний приборов учета не автоматизированы, и обработка данных занимает больше времени. Такой недостаток мы устраняем путем внедрения новой информационной системы.

1.3 Разработка модели бизнес-процесса «Как должно быть»

Модель бизнес-процесса «Как должно быть» создается на основе модели «КАК ЕСТЬ» с устранением недостатков и совершенствованием организации бизнес-процесса. На основе данной модели производится автоматизация

бизнес-процессов [11].

Смоделированная диаграмма бизнес-процесса «Как должно быть» изображена на рисунке 3.

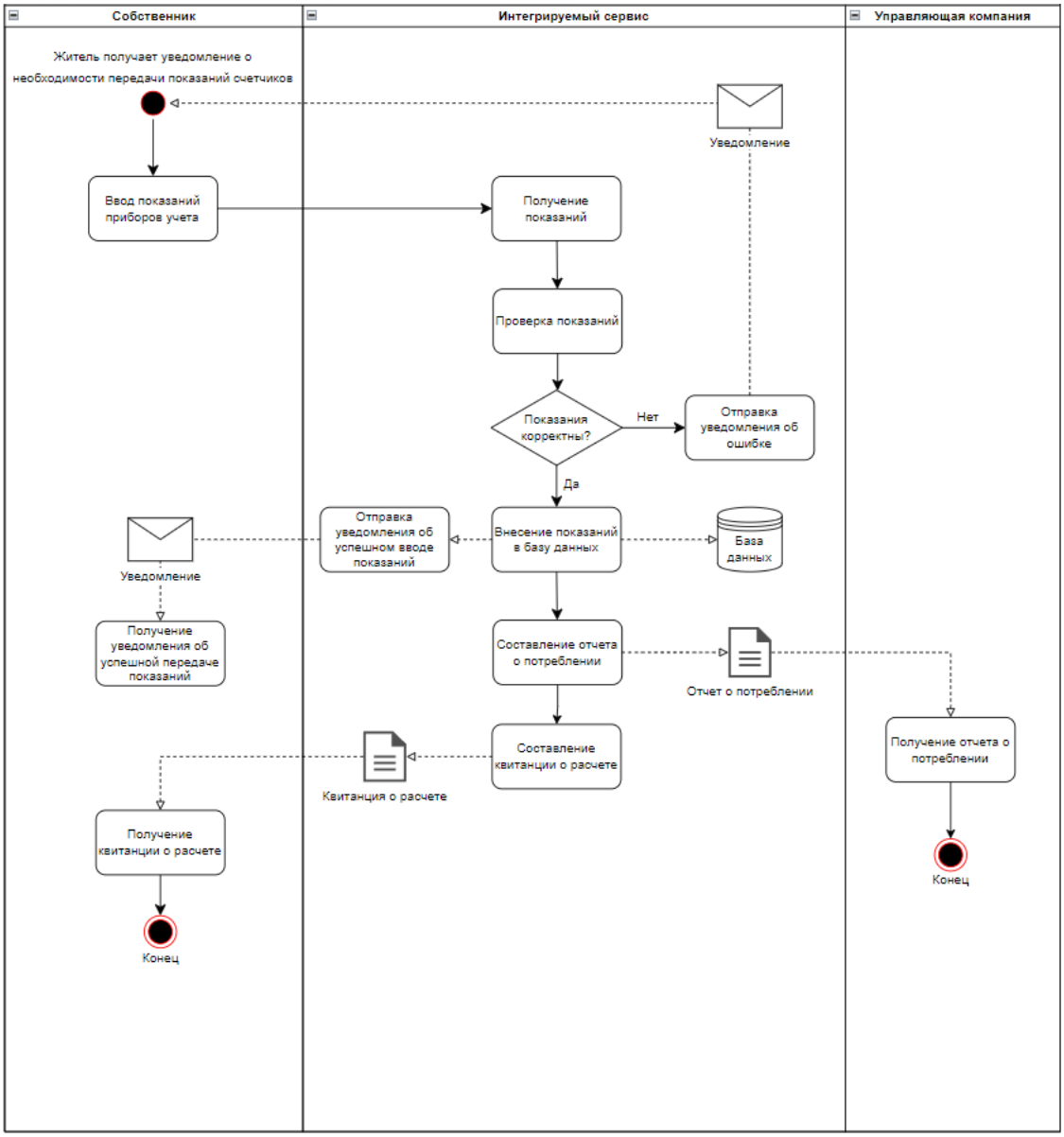


Рисунок 3 – Диаграмма бизнес-процесса «КАК ДОЛЖНО БЫТЬ»

Рассмотрим порядок бизнес-процессов внутри диаграммы:

– в данной модели процесс также начинается с получения собственником уведомления о необходимости передачи показаний счетчиков, он может отличаться тем, что получение уведомления

будет совершаться через сервис;

- собственник вводит показания прибора учета и отправляет их. Особым отличием от модели «КАК ЕСТЬ» является то, что первым делом данные отправляются не в управляющую компанию, а в наш интегрируемый сервис;
- сервис проверяет показания на корректность, анализируя правильность ввода и сравнивая введенные показания с предыдущими;
- в случае, если показания некорректны, сервис отправляет уведомление об ошибке;
- если показания корректны, то система вносит показания в БД;
- далее происходит отправка пользователю уведомления об успешном вводе показаний;
- заключением этих процессов будет происходить составление отчета о потреблении ресурсов, который управляющая компания сможет получить и использовать в своих целях, а также составление квитанции о расчете, которую сможет получить пользователь.

Таким образом, модель «Как должно быть» представляет собой улучшенную и оптимизированную версию процесса управления показаниями счетчиков, которая обеспечивает более эффективное и удобное взаимодействие для собственников и управляющих компаний.

1.4 Анализ существующих решений

Для лучшего понимания ситуации, происходящей в развитии данной отрасли, мы рассмотрим несколько автоматизированных решений.

Существует несколько автоматизированных решений, таких как «saures», которые основаны на использовании счетчиков с дистанционной передачей показаний [16]. Однако, такие решения обычно требуют замены старых счетчиков на дорогостоящие специализированные модели. Также для

подобной автоматизации необходимы контроллеры, обеспечивающие сбор и передачу данных, цена которых варьируется от 5 до 15 тысяч рублей, что может быть неоправданно дорого для многих компаний и товариществ. Цены на котроллеры представлены на рисунке 4. Новые счетчики с функцией дистанционной передачи имеют более высокую стоимость, что делает их недоступными для ограниченных в средствах компаний.

Рисунок 4 – Устройства сбора и передачи данных (УСПД)

Существует несколько программ для учета ЖКХ, которые позволяют управляющим компаниям в одном приложении принимать показания счетчиков, начислять платежи для оплаты и получать оплату от жильцов. Такой подход значительно увеличивает эффективность работы управляющих компаний и упрощает взаимодействие с собственниками, предоставляя им возможность обращаться по любым вопросам или заказывать необходимые справки в онлайн-режиме с гарантированным получением проверенной информации [7].

В качестве примера рассмотрим одну из выдающихся программ учета «Домовладелец».

Программа «Домовладелец» предназначена для управления жилыми объектами и охватывает широкий спектр функциональных возможностей, необходимых для эффективного управления жилыми помещениями, жилищным фондом и коммунальными услугами [5]. Вот основные цели и задачи, для которых используется программа «Домовладелец»:

- программа помогает управляющим компаниям, ТСЖ, ЖСК, ресурсоснабжающим организациям и расчетным центрам эффективно управлять жилыми объектами, включая многоквартирные дома, коттеджные поселки, апартаменты и другие типы жилья;
- автоматизирует процесс расчета коммунальных платежей согласно требованиям законодательства РФ, учитывая потребленные ресурсы (электричество, газ, вода) и другие начисления;
- позволяет вести учет показаний счетчиков (газа, воды, электроэнергии) жилых помещений, обеспечивая точный и надежный мониторинг потребления ресурсов;
- сохраняет историю лицевых счетов жильцов, начисления коммунальных платежей и других операций, облегчая учет и анализ данных;
- формирует единую базу данных о бухгалтерии, паспортном столе,

жилом фонде, аренде и других смежных сферах в единой базе данных обеспечивает централизованный и удобный доступ к информации;

– позволяет раздельно учитывать начисления на капитальный ремонт, а также другие дополнительные услуги, такие как уборка, благоустройство и ремонт общественных зон. На рисунке 5 представлен функционал программы «Домовладелец» по выводу показаний приборов учета.

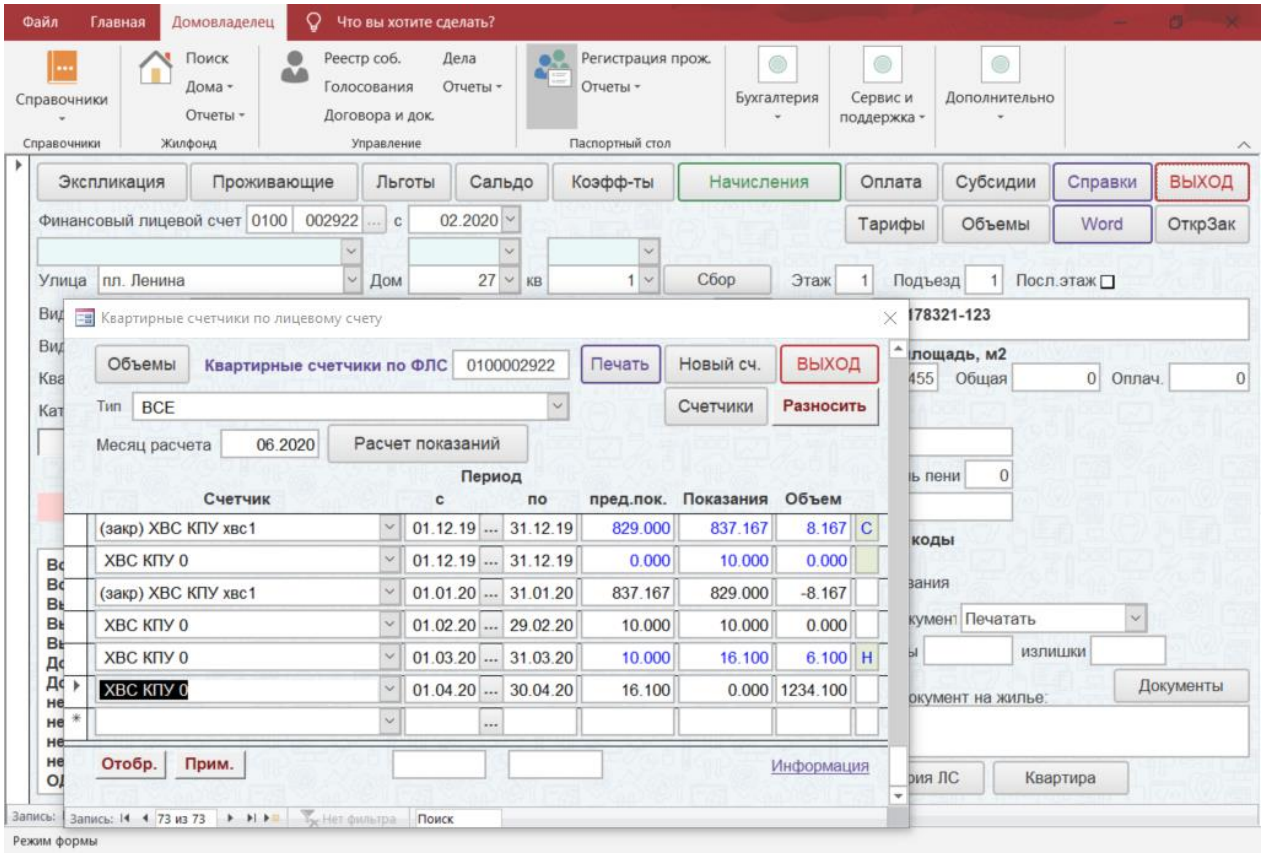


Рисунок 5 – Приборы учета по лицевому счету

Также система «Домовладелец» имеет собственное мобильное приложение, что позволяет собственникам удобно взаимодействовать с управляющей компанией, передавать показания счетчиков, оплачивать услуги ЖКУ и получать актуальную информацию.

Несмотря на множество положительных аспектов, у программы «Домовладелец» также могут быть некоторые минусы или ограничения, а именно:

- Сложность внедрения и настройки, так как программа может потребовать много времени и усилий на внедрение и настройку под конкретные потребности организации. Программа также может иметь ограничения в совместимости с другими системами или программным обеспечением, что может затруднить интеграцию и обмен данными;
- Стоимость лицензий может быть высокой для некоторых компаний или товариществ, особенно для небольших бюджетов;
- Некоторые пользователи могут столкнуться с трудностями в освоении интерфейса и функционала программы из-за его сложности или излишней детализации;
- Также, из-за большого функционала данная программа зависит от технической поддержки и обновлений программы, что требует дополнительных затрат.

Так, многие функции, предоставляемые в этом приложении, могут быть бесполезны для определенных типов товариществ, и платное обслуживание данной программы, может быть неоправданным по отношению к полезности. Опишем сравнительный анализ возможных решений в таблице 1.

Эта таблица отражает основные критерии, которые следует учитывать при выборе метода автоматизации для сбора показаний счетчиков и учета в сфере ЖКХ.

В то время как «Saures» и программы учета предлагают готовые решения с различными уровнями инвестиций и поддержки, самостоятельная разработка веб-приложения предоставляет возможности настройки и адаптации, которые могут быть необходимы для удовлетворения специфических потребностей.

Таблица 1 – Сравнение решений по критериям

Критерии сравнения	**Saures**	**Программы учета**	**Самостоятельная разработка**
Стоимость внедрения	Высокая из-за необходимости покупки специализированных устройств	Может быть высокой из-за лицензий и настройки	Бесплатно
Комфорт использования	Минимизация ручного учета за счет автоматизации	Удобство онлайн взаимодействий	Адаптируется под конкретные требования
Функционал	Только передача показаний	Обширный спектр функций для ЖКХ	Сбор и анализ показаний с возможностью расширения
Интеграция	Ограниченная с другими системами	Возможны ограничения в совместимости	Гибкая и настраиваемая под различные системы
Масштабируемость	Ограничена оборудованием	Масштабируема правообладателем	Высокая масштабируемость
Персонализация	Ограничена оборудованием	Стандартный набор функций	Адаптируема
Платное обслуживание	Замена сломанных устройств	Оплата подписки	Не требуется

Исходя из анализа существующих решений, наиболее подходящим способом решения существующей проблемы сбора показаний счетчиков будет самостоятельная разработка сервиса.

Для большей удобности и комфорта для пользователей будет целесообразным разработать данный сервис в формате веб-приложения. Это обеспечит постоянный доступ к функциям сервиса, имеющим доступ в браузер.

Разработка собственного сервиса даст нам возможность легко интегрировать его с другими информационными системами, что также позволит адаптировать его под конкретные нужды заказчика.

Выводы по главе 1

Был проведен анализ предметной области, а также разработана модель бизнес-процесса передачи показаний приборов учета «Как есть» и модель бизнес-процесса «Как должно быть». Эти модели позволили оценить текущий и желаемый состояния процесса передачи показаний приборов учета. Были отражены текущие шаги и этапы процесса, включая методы сбора, обработки и передачи данных о показаниях приборов учета. Этот анализ позволил выявить проблемные моменты и неэффективности в данном процессе.

Проанализированные существующие программные решения показали недостатки и ограничения, которые могут быть устранены через разработку собственного веб-сервиса, специально адаптированного под потребности предметной области.

Была поставлена задача – разработать собственный сервис в формате веб-приложения, для обеспечения возможности передачи и проверки показаний приборов учета.

Глава 2 Проектирование веб-приложения

2.1 Формирование требований для разрабатываемого приложения

На основе полученных результатов первой главы образовалась необходимость в разработке веб-приложения для проверки и передачи показаний приборов учета. Требования для приложения будут формироваться также, исходя из результатов первой главы.

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

Функциональные требования — это требования к поведению системы, они определяют функциональность разрабатываемого программного обеспечения, т.е. описывают возможности, которые предоставляет система [17].

Функциональные требования определяют «как» необходимо реализовать проект, включают в себя бизнес-требования и пользовательские требования [17].

На основе результатов анализа можно выявить следующие функциональные требования:

- система должна предоставлять возможность регистрации новых пользователей;
- после регистрации пользователи должны иметь возможность аутентифицироваться для доступа к системе;
- разработка удобного пользовательского интерфейса для ввода показаний по четырем типам счетчиков: газ, холодная вода, горячая вода и электричество;
- предоставление предыдущих показаний счетчиков для каждого типа, чтобы пользователи могли видеть их и избежать ввода некорректных данных;
- реализация алгоритмов проверки введенных показаний счетчиков на

предмет корректности и разумности;

- предупреждение пользователей о возможных ошибках при вводе данных;
- создание базы данных для хранения информации о показаниях счетчиков и пользователях;
- обеспечение структуры базы данных, позволяющей эффективное сохранение и доступ к данным;
- администратор должен видеть данные, введенные пользователями.
- администратор должен иметь возможность видеть общие показания всех жильцов по счетчикам.
- администратор должен иметь возможность добавлять новых пользователей в систему.

Также, мы должны поставить и нефункциональные требования.

Нефункциональные требования – это требования к характеру поведения системы, описывают как должна работать система, свойства и ограничения, накладываемые на разрабатываемое программное обеспечение [14]. Были выявлены следующие нефункциональные требования:

- доступ к нашему веб-приложению должен быть многопользовательским;
- обеспечение защиты данных от несанкционированного доступа с помощью механизмов аутентификации, авторизации и шифрования;
- использование безопасных протоколов для передачи данных между клиентом и сервером;
- обеспечение доступности приложения из любого устройства с доступом к интернету;
- создание интуитивно понятного и удобного для использования пользовательского интерфейса;
- гарантирование точности и надежности сохраняемых данных о показаниях счетчиков;

– реализация механизмов контроля целостности данных и предотвращения ошибок в системе;
– проведение тестирования приложения на соответствие функциональным и нефункциональным требованиям;
– исправление обнаруженных ошибок и неполадок в работе приложения;
– приложение должно быть реализовано на платформе, которая позволит достаточно просто расширять функционал для новых задач, по типу новостных лент и форм для голосования;

Ожидаемым результатом проекта должно стать веб-приложение, которое позволит пользователям удобно вводить и отслеживать показания счетчиков. Кроме того, целесообразно будет включить возможность отправки жалоб и предложений в управляющие компании в виде отдельного сервиса по передаче показаний приборов учета коммунальных услуг.

Какие преимущества это дает:

– улучшение Обратной Связи. Предоставление жильцам возможности отправлять жалобы и предложения напрямую управляющим компаниям создает канал обратной связи. Это позволяет жильцам выражать свои замечания и предложения по улучшению качества предоставляемых услуг;
– реагирование на Проблемы в Реальном Времени. Система обратной связи дает управляющим компаниям возможность оперативно реагировать на поступающие жалобы. Это способствует решению проблем в более короткие сроки и создает положительный опыт для жильцов;
– повышение Уровня Удовлетворенности. Возможность активно взаимодействовать с управляющей компанией через отправку жалоб и предложений может повысить уровень удовлетворенности жильцов. Они могут видеть, что их мнение ценится и принимается во внимание;

– итеративное Улучшение Сервиса. Анализ жалоб и предложений может служить основой для постоянного совершенствования работы управляющей компании. Управляющие компании могут выявлять паттерны и общие темы, которые могут быть использованы для улучшения процессов и услуг.

Помимо всего прочего, это имеет огромную пользу и для самих управляющих компаний, так как, за неимением возможности обратной связи с управляющими компаниями, жители нередко направляют свои жалобы напрямую в государственную жилищную инспекцию(ГЖИ).

По информации гос. жилищной инспекции, более половины жалоб от собственников подтверждается. Неубранная территория, вышедший из строя мусоропровод, сосульки. Эти жалобы рассматриваются жилищным инспектором как нарушение правил содержания дома.

«Это опасное нарушение, особенно для управляющих организаций(УО). Дело в том, что в отношении УО эту категорию нарушений инспекторы квалифицируют как нарушение лицензионных требований, и сумма штрафа может достигать 300 тыс. руб. (ч. 2 ст. 14.1.3 КоАП)»[9].

2.2 Выбор технологии разработки веб-приложения

Для разработки нашего веб-приложения нам необходимо такое решение, чтобы с интерфейса пользователя, отражаемого на странице, пользователю давался доступ к базе данных, для получения или внесения необходимых данных. Решение должно быть безопасным и удобным в использовании [18]. Для реализации такого решения существуют передовые языки программирования, имеющие готовые фреймворки, способные упростить разработку приложения [13].

Для разработки веб-приложения, которое требует доступа к базе данных и обеспечивает безопасное взаимодействие с пользователем через веб-интерфейс, можно рассмотреть несколько фреймворков.

Django (Python) - Django представляет собой полноценный фреймворк, который включает множество встроенных функций и модулей для создания веб-приложений любой сложности [19]. Имеет встроенный ORM, обеспечивающий простоту взаимодействия с базой данных. Django подходит для создания крупных и сложных проектов, так как обладает множеством встроенных инструментов для работы с базой данных, аутентификации, администрирования и т.д.

Flask(Python) - более упрощенный по сравнению с django фреймворк, но также подходит для веб-разработки. Позволяет быстро создавать веб-приложения и работать с базой данных. Минусом является то, что он имеет меньший объем встроенных инструментов чем django и требует большей настройки для больших и сложных проектов [21].

ASP.NET (C#) - это мощный фреймворк для разработки веб-приложений. Он предоставляет разработчикам широкий набор инструментов и технологий для создания современных и масштабируемых приложений [26]. Однако ASP.NET и язык программирования C# могут показаться более сложными для изучения новичкам по сравнению с некоторыми другими языками и фреймворками. Это может потребовать времени и усилий для освоения основных методов разработки.

Spring (Java) - Эффективный инструментарий для разработки веб-приложений на Java. Мощная поддержка работы с базой данных и безопасности. Достаточно сложный синтаксис и большое количество аннотаций и настроек может быть причиной затруднения разработки для начинающих [23]. Spring Framework основан на языке Java, который, хотя и мощный, но может быть более громоздким по сравнению с другими более легковесными языками, такими как Python.

Проведем оценку фреймворков для веб-разработки в таблице 2. Для оценки будем использовать ряд критериев, а именно: простота изучения, популярность, простота использования, удобство работы с базой данных, эффективность работы при обработке данных и скорость разработки. Оценим

каждую технологию по шкале от 1 до 5, где 5 - высокая оценка, а 1 - низкая.

Таблица 2 – Сравнение выбранных фреймворков для разработки

Критерий оценивания	Анализируемый фреймворк			
	Django (Python)	Flask (Python)	ASP.NET (C#)	Spring (JAVA)
Простота изучения	5	5	3	3
Популярность	5	4	4	4
Простота использования	4	5	3	3
Удобство работы с БД	5	5	4	4
Эффективность работы при обработке данных	4	3	5	5
Скорость разработки	5	5	3	3

Критерий простоты изучения оценивает, насколько легко изучить эту технологию. Оценка популярности показывает, насколько широко этот фреймворк используется в индустрии при разработке веб-приложений. Чем популярнее фреймворк, тем легче найти решение возникающих проблем при разработке, и тем больше имеется готовых решений, для нашего проекта [24].

Простота использования оценивает удобство разработки при помощи этого фреймворка. Чем проще его использовать, тем быстрее и эффективнее можно создавать программное решение.

Удобство работы с БД оценивает, насколько эффективно можно работать с базой данных, и насколько удобны инструменты для работы с ней, например, django имеет встроенный ORM, облегчающий работу с БД. По умолчанию Django в качестве БД использует SQLite. Все файлы базы данных могут легко переноситься с одного компьютера на другой. при необходимости мы можем использовать в Django и другие СУБД. [25].

Эффективность при обработке данных оценивает скорость работы при обработке больших объемов поступающей информации. Так, например, java является компилируемым языком, что позволяет достичь высокой скорости обработки данных. Скорость разработки оценивает, насколько быстро и

просто можно разработать приложение при помощи данного фреймворка.

Исходя из итоговой суммы оценки, можно сделать вывод, что целесообразнее использовать фреймворк Django языка python. Python является высокоуровневым языком программирования, что существенно упрощает его использование и ускоряет процесс разработки. Также django имеет встроенную ORM, что упрощает взаимодействие с базой данных. Это позволяет легко определять модели данных, выполнять запросы и работать с данными, сводя к минимуму ручное написание SQL-запросов.

2.3 План разработки веб-приложения

Веб-приложение, как и любой проект, перед началом разработки должен иметь план. Для правильного построения плана мы первым делом должны спланировать структуру и общий вид приложения перед пользователем. Это нужно для того, чтобы понять порядок действий, выполняемый пользователями для поставленной цели.

Для демонстрации основных функций, выполняемых пользователями данного приложения, мы будем использовать диаграмму вариантов использования. Диаграмма вариантов использования (Use Case diagram), представленная на рисунке 6, показывает основные функции системы для каждого типа пользователей и отношения между актерами и вариантами использования [3].

Пользователями нашего приложения будут являться собственники квартир или дачных участков, а также администратор управляющей компании или товарищества. Первым этапом является регистрация пользователя. В целях безопасности приложения мы дадим администратору возможность создавать новых пользователей. Это обеспечит более строгий контроль доступа, что в свою очередь поможет избежать нежелательной активности, такой как создание фальшивых или ненужных учетных записей, с которых пользователи, не являющиеся собственниками квартир или дачных участков,

будут иметь возможность отправлять заявления в управляющую компанию.

Рисунок 6 – Диаграмма вариантов использования

Далее, после того как пользователь будет зарегистрирован, он сможет войти в свой профиль на странице авторизации.

В личном кабинете пользователя у него будет возможность передать показания приборов учета, посмотреть свои предыдущие показания или написать сообщение. Важным замечанием является то, что при вводе показаний, они будут проверяться на валидность, а именно, значения должны быть больше нуля, и должны быть больше или равны предыдущим.

Администратор, авторизовавшись в системе, будет иметь возможность как увидеть общую статистику показаний по всем собственникам, так и показания отдельно взятого собственника. Также администратор будет иметь возможность просматривать сообщения от собственников для оперативного решения возникающих проблем.

Таким образом, структура нашего приложения будет разделена на три основных экрана.

– Экран авторизации.
– Экран личного кабинета собственника.

– Экран кабинета администратора.

Все эти экраны играют важную роль в работе приложения, обеспечивая доступ к основной функциональности и информации. Они должны быть удобными и интуитивно понятными для пользователей.

Благодаря данной структуре весь используемый функционал нашего приложения сможет находиться на одном экране для пользователя.

2.4 Разработка пользовательского интерфейса веб-приложения

При разработке пользовательского интерфейса важно придерживаться определенных принципов, которые определяют способы взаимодействия пользователя с приложением и его общую структуру. Ниже будут представлены основные принципы и их требования.

Принцип интуитивно понятного и доступного интерфейса. Элементы интерфейса должны быть понятны пользователю, чтобы он мог легко выполнить нужные действия и получить или передать нужную информацию.

Требования:

– Интерфейс должен быть простым и лаконичным, используя минимальное количество кнопок и предоставляя необходимые варианты выбора.
– Кнопки должны иметь понятные графические или текстовые обозначения, которые не вызывают сомнений у пользователя.

Принцип структурности интерфейса. Данный принцип определяет последовательность представления информации пользователю в удобной форме.

Требования:

– Структура интерфейса должна быть логичной и последовательной, чтобы пользователь мог легко выполнять операции.
– Элементы интерфейса должны быть устроены с учетом стандартной структуры размещения для лучшей ориентации пользователя.

Принцип привлекательности интерфейса. Данный принцип отвечает за гармоничный и приятный для пользователя интерфейс, который учитывает психологические и визуальные аспекты.

Требования:

– Цветовая схема интерфейса должна быть единообразной, не затрудняя читаемость и соответствуя содержанию приложения.
– Элементы интерфейса должны быть связаны и выделены с использованием различных стилей в зависимости от их назначения.

Все эти принципы направлены на создание удобного, легко понятного и эстетичного интерфейса, который поможет пользователям легко взаимодействовать с вашим приложением.

Учтя вышеперечисленные принципы разработки пользовательского интерфейса, были спроектированы шаблоны рабочих экранов пользователей.

На рисунке 7 представлен экран авторизации. Он имеет поле ввода логина и ввода пароля. Как уже было описано ранее, экрана регистрации пользователя не будет в открытом доступе, поэтому ссылки на страницу регистрации указано не будет.

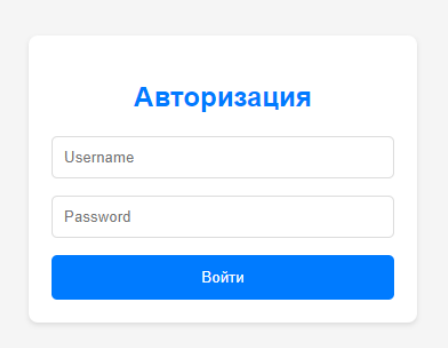


Рисунок 7 – Экран Авторизации

Данный экран имеет конкретную цветовую схему, состоящую из белого и светло-голубого цветов. Данная цветовая схема будет единообразной на всех экранах приложения.

Пользователь вводит данные для входа – логин и пароль. После нажатия на кнопку «Войти» данные отправляются на сервер и, если они корректны, происходит переход на экран личного кабинета собственника.

На рисунке 8 представлен экран личного кабинета собственника.

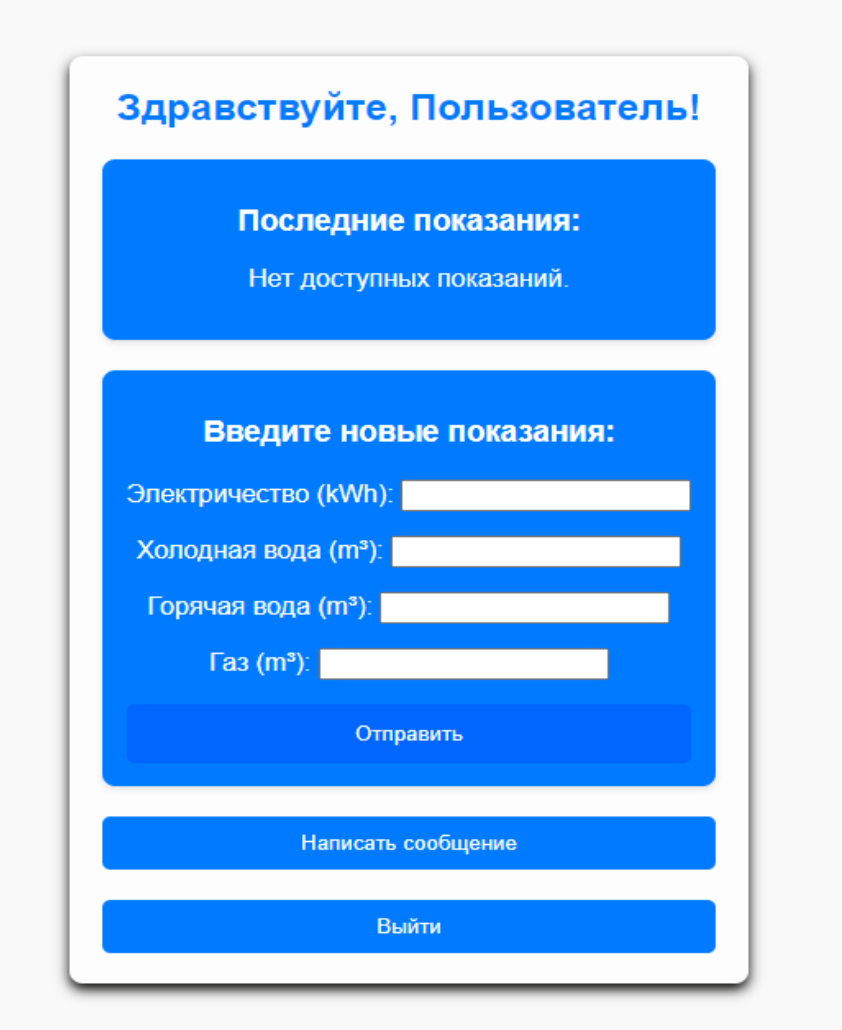


Рисунок 8 – Экран личного кабинета собственника

На данном экране будут представлены:

– контейнеры для показа последних показаний приборов учета, представленных пользователем.
– контейнер для ввода новых показаний.
– кнопка «написать сообщение», по нажатию которой будет выводиться форма для написания сообщения.

– кнопка «Выйти», по нажатию которой пользователь выйдет из своего профиля

Контейнер для ввода новых показаний является основным сценарием работы пользователя. Именно здесь осуществляется ввод, проверка и отправка показаний приборов учета.

Далее, на рисунке 9 будет показан экран личного кабинета администратора.

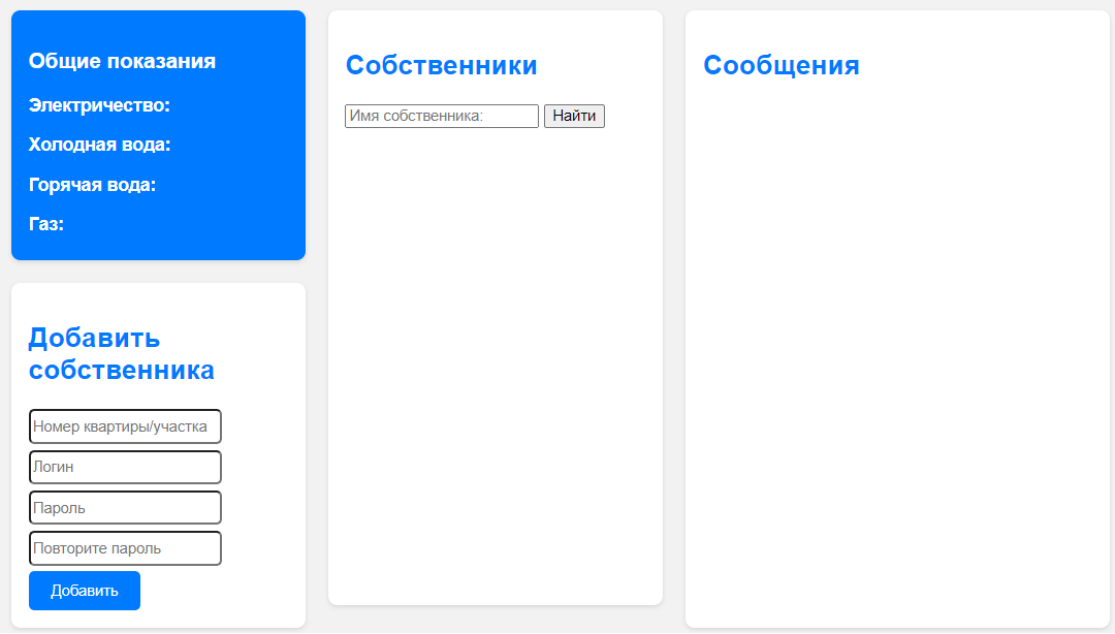


Рисунок 9 – Экран кабинета администратора

На данном экране будут представлены:

– Контейнер с общими показаниями всех собственников;
– Контейнер для регистрации нового собственника, содержащий поля логина, пароля, повтора пароля, и поле для номера квартиры или участка, в зависимости от целевой организации, которому будет предоставляться доступ к данному приложению;
– Контейнер для поиска конкретного собственника;
– Контейнер с сообщениями от собственников.

Спроектированные экраны, которые формируют пользовательский

интерфейс, соответствуют основным принципам интуитивной понятности, структурности и привлекательности интерфейса. Они служат ориентиром для разработки, упрощая процесс и придавая целостность будущему приложению. Это упрощает процесс разработки, так как разработчики имеют четкий план оформления и распределения элементов, а пользователи получают удовлетворение от использования интуитивно понятного и привлекательного приложения.

Выводы по главе 2

В данной главе были четко определены как функциональные, так и нефункциональные требования к проекту, что будет являться фундаментом для дальнейшей разработки и реализации приложения.

Выбор технологии разработки веб-приложения является ключевым моментом в проектировании системы. Основываясь на детальном сравнении различных фреймворков, для разработки был выбран Django, платформа, которая наилучшим образом сочетает в себе скорость разработки, гибкость и мощные функции работы с базами данных. Такой выбор ориентирован не только на удобство создания и управления приложением для разработчиков, но и на обеспечение легкого доступа и безопасного взаимодействия для конечных пользователей.

Также был разработан пользовательский интерфейс, который имеет решающее значение для успеха приложения, поскольку от него напрямую зависит удовлетворенность пользователя. При проектировании интерфейса были учтены основные принципы его создания, включая интуитивную понятность, структурность и привлекательность интерфейса. Это обеспечивает создание удобного, простого в использовании и в то же время функционального пользовательского интерфейса.

Глава 3 Реализация веб-приложения

3.1 Архитектура разрабатываемого веб-приложения при помощи фреймворка «Django»

При реализации веб-приложения с помощью фреймворка «Django» процесс разработки упрощается за счет организации кода на функциональные блоки [22]. Данные блоки представляют собой некий каркас проекта с готовой архитектурой, имеющий необходимые инструменты для реализации веб-приложения.

Рассмотрим работу приложения на Django. На рисунке 10 продемонстрирована схема работы, по которой будет построено приложение.

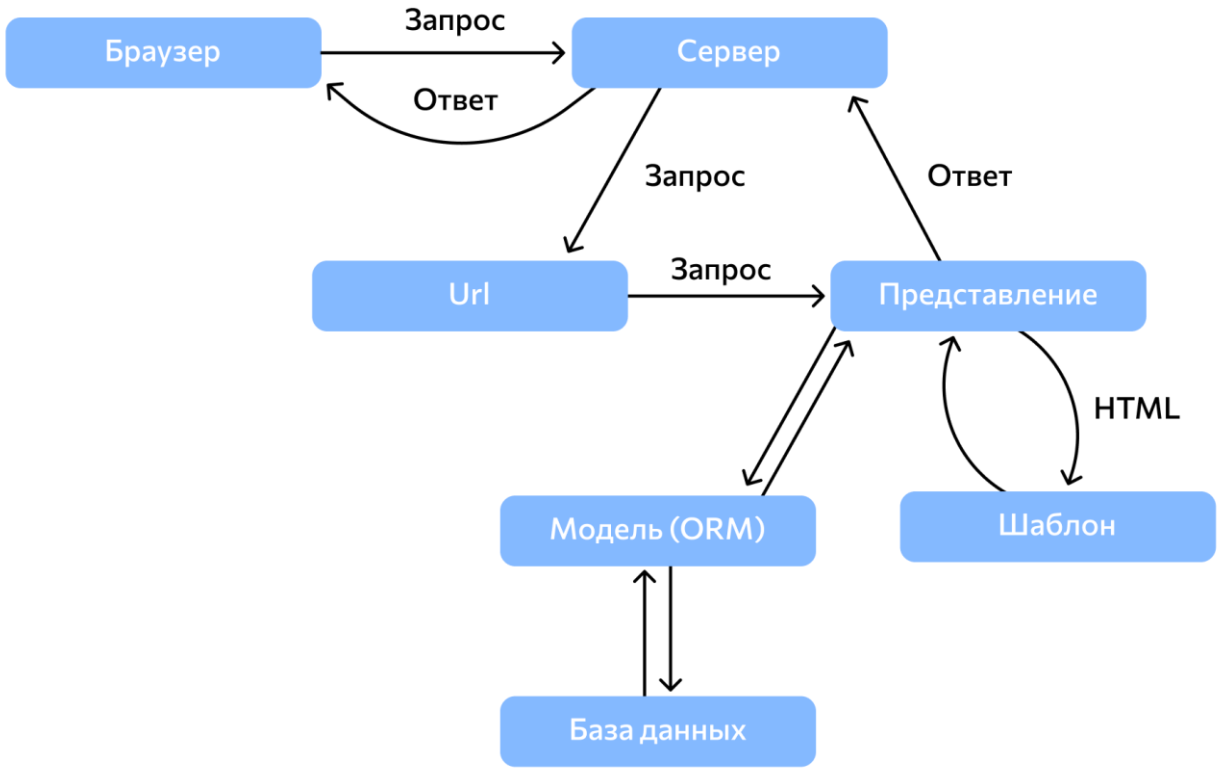


Рисунок 10 – Схема работы Django предложения

Веб-приложение на Django следует архитектурному стилю MVT (Model-

View-Template), который схож с широко известным MVC (Model-View-Controller) [20]. В Django представление (View) исполняет большую часть логики, связанной с обработкой запросов и отображением, а шаблоны (Templates) отвечают за представление информации в удобной для пользователя форме [15].

Работа подобных приложений начинается с запроса, который пользователь отправляет из своего браузера, нажимая на ссылку или кнопку, или вводя URL-адрес в адресную строку браузера. Запрос попадает на сервер, где работает Django-приложение.

Далее, Сервер использует URL-диспетчер Django для определения, какому представлению (view) следует передать запрос. URL-диспетчер хранит таблицу соответствия между путями URL и функциями Python, которые обслуживают эти URL-адреса. Это может быть функция или класс представления.

Функции представления принимают запрос, обрабатывают переданные данные, обращаются к моделям (Models), если задействован доступ к данным БД, и формируют ответ для клиента. Представления также определяют, какие данные необходимо отобразить на странице.

В случае, если при обработке запроса нужно получить или изменить данные, представление обращается к моделям. Модели - это функциональный блок, который позволяет Django представлять таблицы базы данных в виде классов Python. Модели определяют структуру данных, обеспечивают интерфейс для их запроса, добавления, изменения и удаления.

Если в представлении были операции с данными, модели взаимодействуют с базой данных, сохраняют изменения или запрашивают данные. Получив данные из моделей, представление обрабатывает их и посылает в шаблон для формирования ответа. Шаблон - это файл, содержащий статический HTML и теги шаблонизатора Django, который позволяет вставлять в HTML-код динамическую информацию, которую в последствии принимает пользователь. Затем шаблон обрабатывается с новыми данными и

преобразуется в HTML-файл, который отображается в браузере пользователя.

Таким образом, каждый запрос от браузера пользователя проходит через данную систему блоков Django приложения, внутри которых он обрабатывается, и возвращается пользователю в виде HTML-страницы.

Структура нашего проекта разработки веб-приложения представлена на рисунке 11.

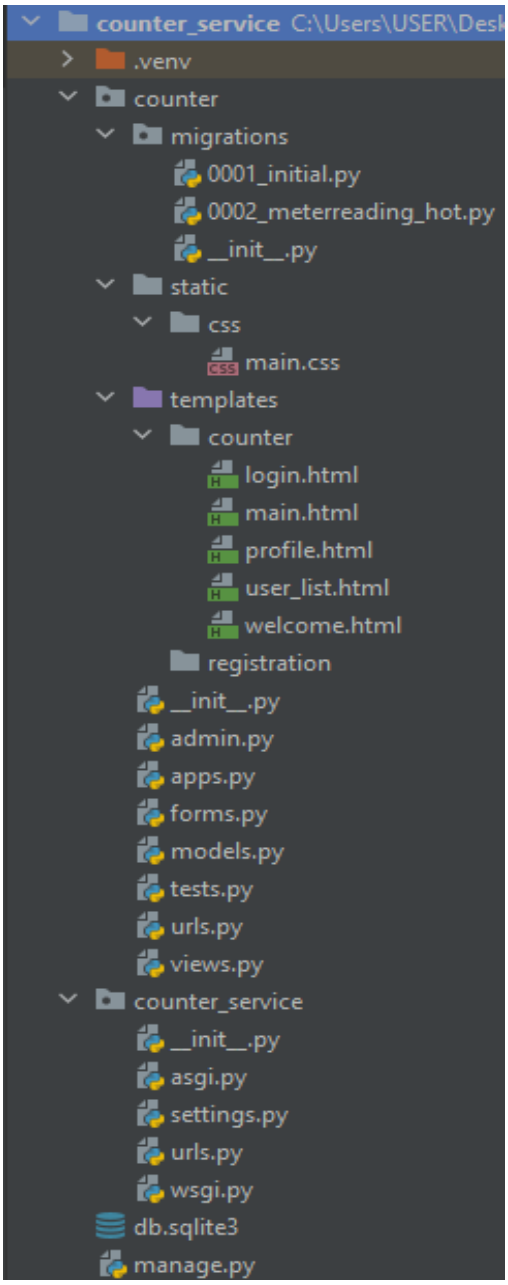


Рисунок 11 – Реализованная структура веб-приложения

В нашем приложении расположены все необходимые составные части для работы Django приложения. Ключевыми файлами являются: models.py, forms.py, views.py, urls.py приложения. Также затрагиваются папки: templates, css, migrations.

В файле models.py определяются модели данных приложения, которые используются для создания таблиц в нашей базе данных. Каждый класс модели представляет собой таблицу в базе данных, а атрибуты класса - это поля таблицы. Например, в данном файле мы описываем модели для показаний приборов учета или сообщений пользователей.

В файле forms.py определяются формы Django, которые используются для валидации и обработки данных, введенных пользователем на странице. Формы могут быть связаны с моделями для создания, обновления или удаления данных. В forms.py описаны формы написания сообщения, ввода показаний приборов учета и добавления новых пользователей.

Файл views.py представляет собой главную часть работы приложения, отчасти являясь связующим звеном в работе всей программы. В ней обрабатываются запросы от пользователей и происходит взаимодействие с моделями, формами и шаблонами для выполнения необходимой логики. После выполнения операций представления могут возвращать результат на страницу пользователя или обновлять содержимое базы данных.

Файл urls.py определяет маршруты URL для приложения, которые указывают Django на соответствующие представления. Здесь описываются пути к различным функциям представлений. Эти маршруты используются для навигации пользователей по вашему веб-приложению. Когда пользователь вводит определенный URL, Django определяет соответствующий маршрут и вызывает связанную функцию представления для обработки запроса.

Папка templates является необходимым звеном в работе Django приложения. В ней содержатся html-шаблоны, которые отображаются пользователю. По сути, они являются страницами в браузере, которые пользователь видит и контактирует с ними.

В директории css содержатся файлы формата .css, используемые для формирования стилей в html-шаблонах. Стили из этих файлов применяются к элементам веб-страниц для улучшения внешнего вида и упрощения использования пользователем.

В папке migrations хранятся файлы миграций Django, которые представляют собой набор инструкций для обновления базы данных с учетом изменений в моделях приложения. Когда мы определяем новые модели или вносим изменения в существующие модели, Django автоматически создает миграции, которые представляют собой файлы Python, описывающие изменения, необходимые для обновления схемы базы данных, Django автоматически создает таблицы баз данных, определяет какие поля были добавлены, изменены, удалены и многое другое. Таким образом, миграции позволяют вносить изменения в структуру базы данных без необходимости ручного вмешательства.

Используя данные инструменты и было реализовано веб-приложение с использованием фреймфорка Django. Далее опишем ключевой процесс нашей программы, а именно: передачу и проверку показаний приборов учета.

3.2 Разработка алгоритма проверки показаний приборов учета

Собственник, используя свой браузер, переходит на специализированную страницу и отправляет свои показания. Перед тем как занести новые показания в базу данных, из неё производится запрос последних показаний данного собственника. По получении последних показаний, они сравниваются с новыми. В случае, если новые показания больше предыдущих, они заносятся в базу данных. После сохранения собственник получает уведомление об успешном внесении показаний. Диаграмма последовательности процесса проверки показаний приборов учета представлена на рисунке 12.

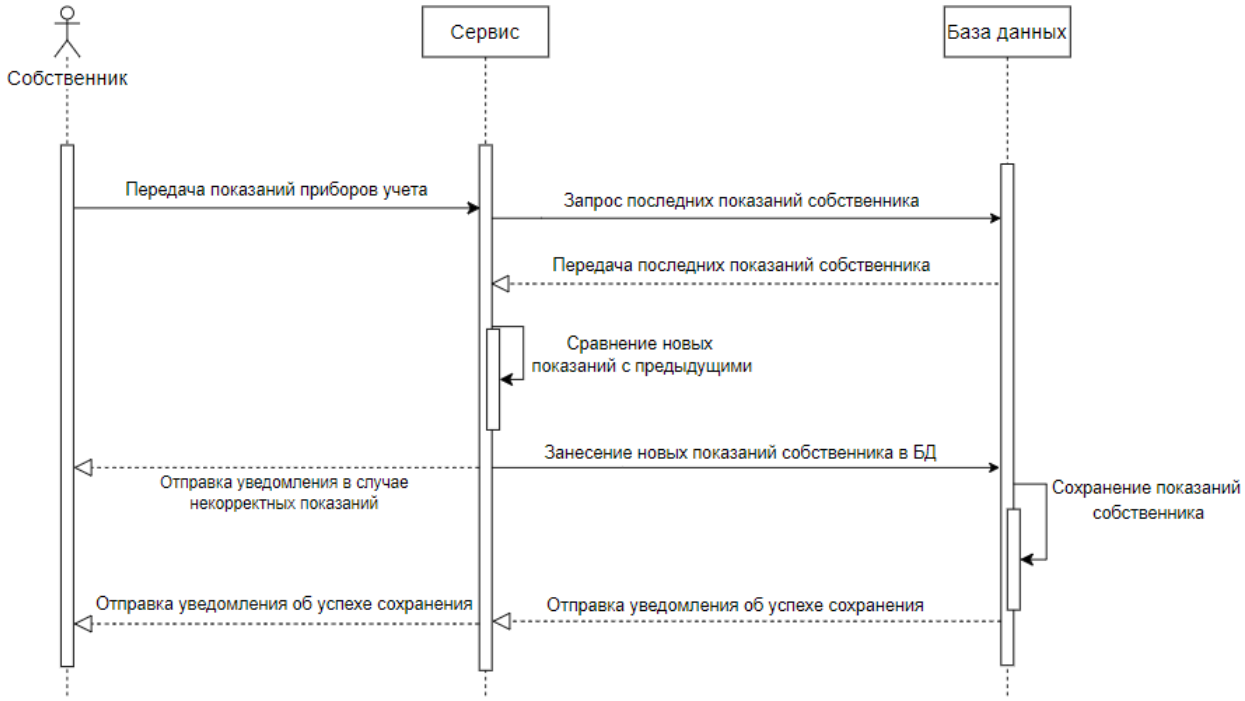
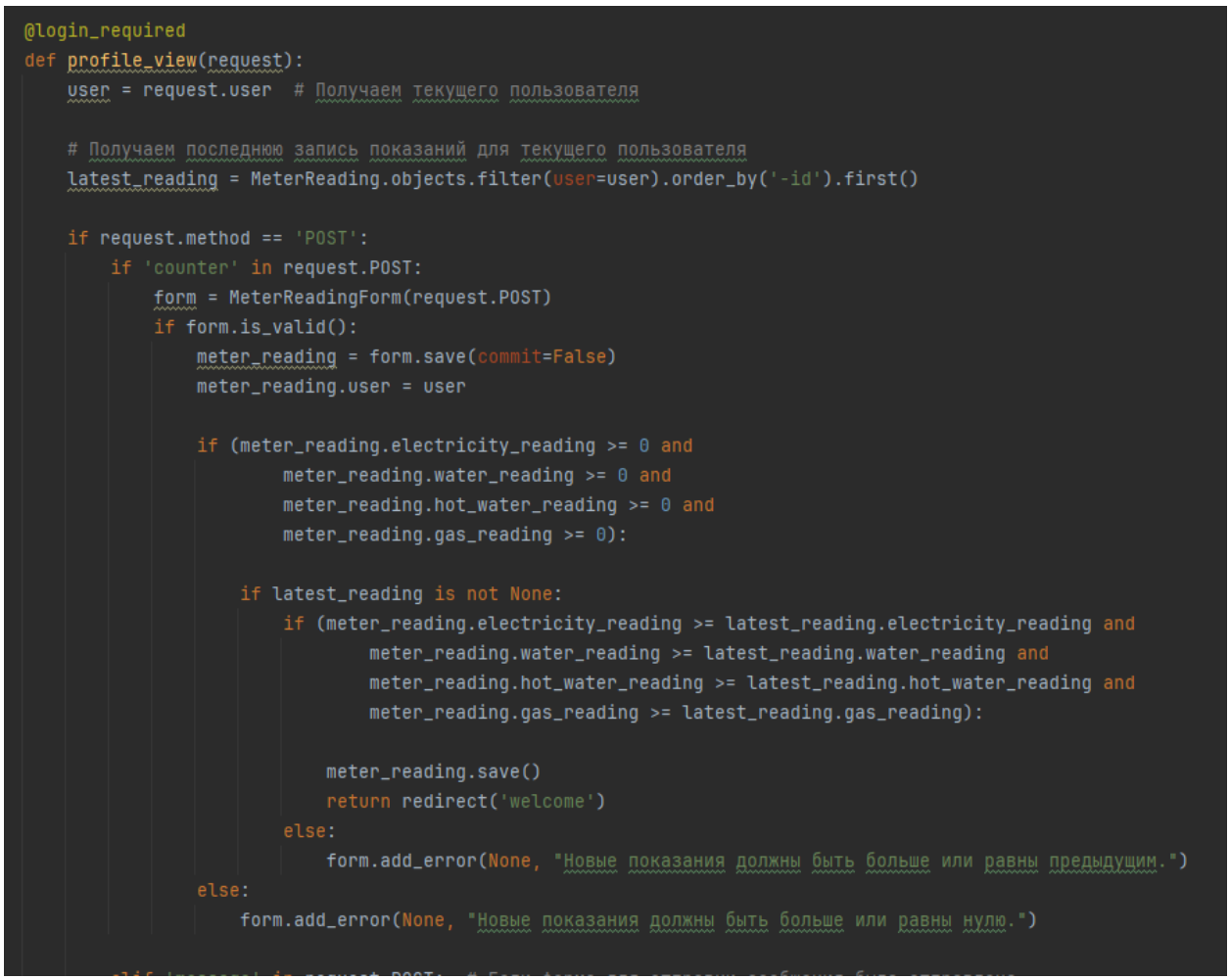


Рисунок 12 – Диаграмма последовательности процесса проверки показаний приборов учета

Реализация алгоритма проверки введенных показаний представлена на рисунке 13.

Первым делом, в данном коде запрашивается пользователь, с которым мы зашли, и его последние записи показаний. Дальше выполняется условие if, если была отравлена форма для ввода показаний. Если была отправлена она, то мы проверяем ее валидность, описанную в файле forms.py. Далее мы выполняем проверку на то, чтобы значения показаний были не меньше нуля. Если условие выполнено, мы сравниваем новые показания со старыми используя уже запрошенные данные последних показаний. Если условие выполняется, мы сохраняем новые показания и перенаправляем на страницу с результатом. В случае, если какое-либо условие не выполняется, выводится ошибка о неправильно введенных показаниях.



```python
@login_required
def profile_view(request):
    user = request.user  # Получаем текущего пользователя

    # Получаем последнюю запись показаний для текущего пользователя
    latest_reading = MeterReading.objects.filter(user=user).order_by('-id').first()

    if request.method == 'POST':
        if 'counter' in request.POST:
            form = MeterReadingForm(request.POST)
            if form.is_valid():
                meter_reading = form.save(commit=False)
                meter_reading.user = user

                if (meter_reading.electricity_reading >= 0 and
                        meter_reading.water_reading >= 0 and
                        meter_reading.hot_water_reading >= 0 and
                        meter_reading.gas_reading >= 0):

                    if latest_reading is not None:
                        if (meter_reading.electricity_reading >= latest_reading.electricity_reading and
                                meter_reading.water_reading >= latest_reading.water_reading and
                                meter_reading.hot_water_reading >= latest_reading.hot_water_reading and
                                meter_reading.gas_reading >= latest_reading.gas_reading):

                            meter_reading.save()
                            return redirect('welcome')
                        else:
                            form.add_error(None, "Новые показания должны быть больше или равны предыдущим.")
                    else:
                        form.add_error(None, "Новые показания должны быть больше или равны нулю.")
```

Рисунок 13 – Программный код проверки показаний приборов учета

Таким образом, этот алгоритм обеспечивает надежный и безопасный процесс ввода и сохранения показаний приборов учета, предотвращая ошибки и обеспечивая правильное взаимодействие с пользователем.

3.3 Представление и тестирование приложения

Исходя из требований, представленных во второй главе, было разработано веб-приложение для передачи показаний счетчиков. Рассмотрим его функциональность. Каждый пользователь, будь то собственник или администратор, переходит на корневую страницу сайта, и перенаправляется на страницу авторизации. Окно авторизации представлено на рисунке 14.

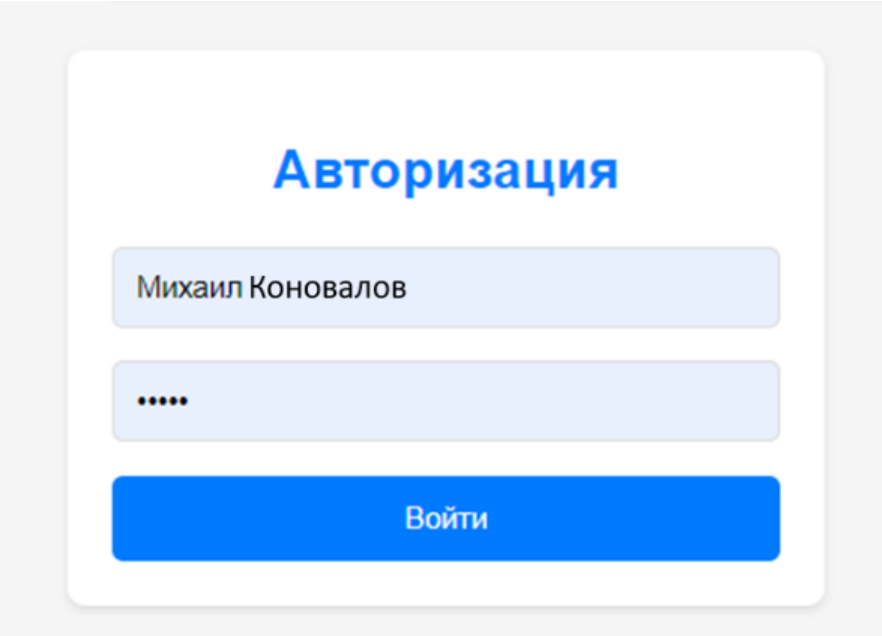


Рисунок 14 – Окно авторизации

Для авторизации пользователю нужно ввести свой логин и пароль. По желанию, данные авторизации можно сохранить в браузере.

После прохождения пользователем авторизации, он получает доступ к личному кабинету. Страница личного кабинета представлена на рисунке 15. В личном кабинете есть несколько функциональных разделов: последние показания, ввод новых показаний, отправка сообщения и выход. Раздел последних показаний нужен для упрощения ввода показаний для собственников, что уменьшает шанс неверного ввода. Помимо отправки новых показаний, также существует возможность отправить сообщение администратору в управляющую компанию. Это нужно для того, чтобы упростить процесс сбора жалоб от собственников. По нажатию на кнопку «Написать сообщение» открывается форма с полями «Заголовок» и «Введите сообщение». Форма отправки сообщения представлена на рисунке 16. Кнопка «Выйти» выводит пользователя из личного кабинета обратно на окно авторизации.

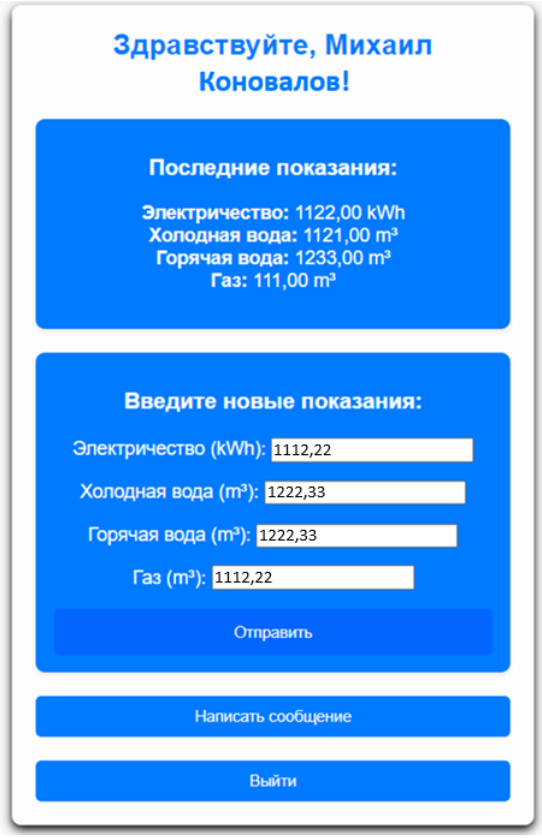


Рисунок 15 – Личный кабинет собственника

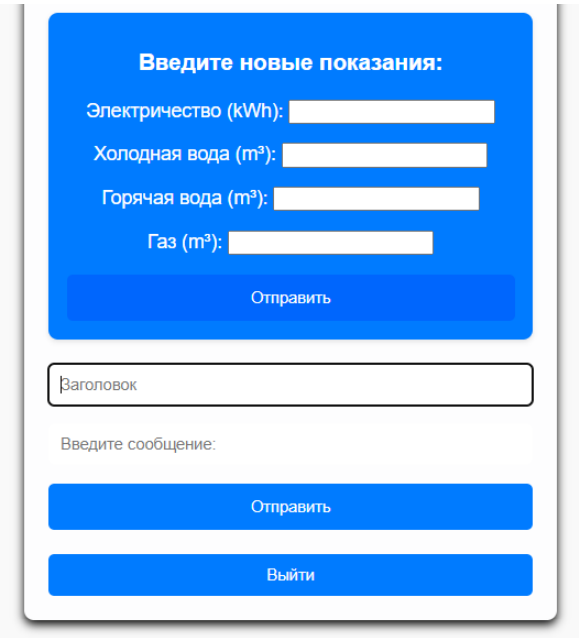


Рисунок 16 – Форма отправки сообщения

Если в процессе авторизации зайдет администратор, то он будет

перенаправлен в кабинет администратора. Кабинет администратора, изображенный на рисунке 17, имеет 4 раздела: общие показания, добавление собственника, собственники, сообщения.

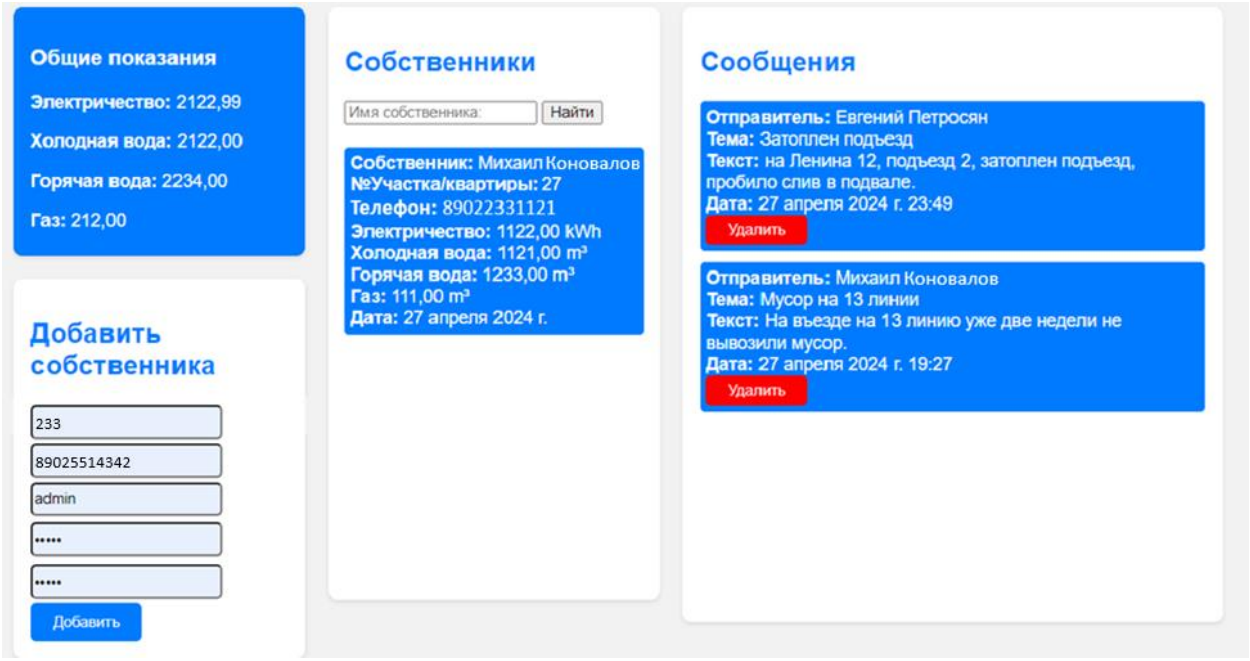


Рисунок 17 – Кабинет Администратора

Раздел общих показаний отображает сумму всех последних показаний от собственников. Добавление собственника представлено в кабинете администратора в целях безопасной работы сервиса. В разделе «Собственники» отображена форма поиска собственников по имени. Результатом поиска является информация о собственнике, включая показания.

В разделе сообщения отображены все сообщения от собственников. Данный раздел упрощает взаимодействие между администрацией и собственниками.

Работоспособность веб-приложения можно протестировать, используя ряд тест-кейсов функциональных требований сервиса. Позитивные и негативные тест-кейсы представлены в Приложении А (таблицы тест-кейсов).

Таблица тест-кейсов (Приложение А) иллюстрирует полное

представление функциональности приложения, включая различные сценарии использования для собственников и администраторов. Каждый тест-кейс содержит ясное описание действия, шаги для его выполнения, входные данные, ожидаемые результаты, фактические результаты и статус теста. В таблице представлены как позитивные, так и негативные тест-кейсы. Позитивные тест-кейсы описывают сценарии использования, при которых приложение должно работать правильно и ожидаемо. Например, позитивные тест-кейсы включают в себя успешную авторизацию, добавление корректных показаний счетчиков, отправку сообщения собственником и т. д. Эти сценарии предполагают, что пользователь взаимодействует с приложением согласно его предназначению [12]. Негативные тест-кейсы, напротив, моделируют ситуации, когда пользователь вводит некорректные данные или выполняет неправильные действия [10]. Например, попытка добавить показания счетчиков, имеющие значения меньше предыдущих, или попытка создать пользователя с некорректными данными.

Анализируя результаты тестирования, можно сделать вывод о том, что основные сценарии использования приложения работают корректно, и все проверки данных, такие как ограничения на ввод показаний и аутентификацию пользователей, реализованы правильно.

Выводы по главе 3

В результате данной главы было разработано веб-приложение, способное эффективно обрабатывать показания приборов учета и сообщения пользователей. В частности, была описана архитектура приложения, описан алгоритм реализации проверки показаний приборов учета, а также проведено представление и тестирование приложения. Дальнейшее развитие приложения может включать в себя расширение его функциональности и оптимизацию производительности.

Заключение

В рамках данной выпускной квалификационной работы был успешно разработан веб-сервис для автоматизации процесса передачи и проверки показаний приборов учёта коммунальных услуг. Созданный сервис представляет собой комплексную систему, ориентированную на удобство и безопасность передачи информации конечными собственниками и их взаимодействие с управляющими компаниями.

На стадии анализа объекта исследования были изучены текущие процессы передачи и проверки показаний приборов учета, выявлен ряд значимых проблем существующих подходов. Был проведен анализ возможных решений. На основе полученных данных была разработана новая оптимизированная модель бизнес-процессов, устраняющая описанные недостатки.

На стадии проектирования были сформированы функциональные и нефункциональные требования нашего сервиса. Также были проанализированы существующие языки программирования для разработки веб-приложения, описан план разработки и спланирована структура разрабатываемого сервиса. Также было проведено построение шаблонов интерфейса пользователя, для создания эргономичного и интуитивно понятного дизайна.

Реализация веб-приложения с помощью фреймворка Django позволила достичь поставленных целей. Значительное внимание было уделено архитектуре приложения, обеспечению безопасности, разработке удобного и интуитивно понятного пользовательского интерфейса, а также созданию эффективного механизма проверки введённых данных. Дополнительным преимуществом стала возможность обратной связи для пользователей, позволяющая напрямую сообщать о проблемах и предложениях, что способствует улучшению качества обслуживания.

Тестирование веб-приложения подтвердило его функциональность и

эффективность, показав соответствие ожиданиям, как в стандартных, так и в негативных сценариях использования. Тестирование выявило также и высокий уровень стабильности приложения, являющегося важным фактором для дальнейшей эксплуатации и развития системы.

Таким образом, созданный сервис вносит вклад в развитие информационных систем в области ЖКХ, предлагая эффективный инструмент для собственников жилья и управляющих компаний. Разработанное приложение не только способствует повышению прозрачности и удобства в процессах контроля за расходом коммунальных ресурсов, но и закладывает основу для дальнейшей цифровизации отрасли.

Дальнейшие пути развития проекта могут включать интеграцию с другими информационными системами, расширение функциональности приложения для управления другими аспектами в сфере ЖКХ, а также внедрение дополнительных инструментов аналитики и отчетности для повышения эффективности управления ресурсами.

Список используемой литературы

1. Вичугова, А.Н. Как начать моделировать бизнес-процессы в BPMN [Электронный ресурс]. – URL: https://systems.education/bpmn-start (дата обращения: 19.02.2024).
2. Давыденко, Е. Все о ЖКХ на 2019 год: услуги, тарифы, платежи и сборы, основания не платить или платить меньше [Электронный ресурс]. – Москва: Издательство АСТ, 2018. – 224 с. – URL: https://www.litres.ru/book/e-i-davydenko/vse-o-zhkh-na-2020-god-uslugi-tarify-platezhi-i-sbory-sposob-35480079 (дата обращения: 19.02.2024).
3. Диаграмма вариантов использования (UseCase diagram) // Flexberry PLATFORM Documentation [Электронный ресурс]. – URL: https://flexberry.github.io/ru/fd_use-case-diagram.html (дата обращения: 19.02.2024).
4. Долганова, О.И. Моделирование бизнес-процессов: учебник и практикум для академического бакалавриата [Электронный ресурс]. – Москва: Издательство Юрайт, 2019. – 289 с. – ISBN 978-5-534-00866-1. – URL: https://urait.ru/bcode/433143 (дата обращения: 19.02.2024).
5. Домовладелец. Современная удобная программа для предприятий ЖКХ [Электронный ресурс]. – URL: https://domovladelec.lps.ru/ (дата обращения: 19.02.2024).
6. Жилищно-коммунальное хозяйство [Электронный ресурс]. – URL: https://ru.wikipedia.org/wiki/Жилищно-коммунальное_хозяйство (дата обращения: 29.01.2024).
7. Как начисляется плата за ЖКУ [Электронный ресурс]. – URL: https://www.mos.ru/otvet-dom-i-dvor/kak-nachislyaetsya-plata-za-zhku (дата обращения: 19.01.2024).
8. Контроль корректности введенных показаний [Электронный ресурс] - VGKH.RU. – URL: https://vgkh.ru/faq-1s-

zhkh/rabota_s_priborami_ucheta_3_0/kontrol-korrektnosti-vvedennykh-pokazaniy (дата обращения: 24.02.2024).

9. На что жители чаще всего жалуются в ГЖИ [Электронный ресурс] – Государственная Жилищная Инспекция. – URL: https://xn----8sbqfnvs2b6d0a.xn--p1ai/na-chto-zhiteli-chashhe-vsego-zhaluyutsya-v-gzhi (дата обращения: 11.02.2024).

10. Негативное тестирование – суть метода и его главные приемы [Электронный ресурс]. – URL: https://www.careerist.com/ru-insights/negativnoe-testirovanie-sut-metoda-i-ego-glavnye-priemy (дата обращения: 21.04.2024).

11. Простое руководство по UML-диаграммам и моделированию баз данных [Электронный ресурс]. – URL: https://www.microsoft.com/ru-ru/microsoft-365/business-insights-ideas/resources/guide-to-uml-diagramming-and-database-modeling (дата обращения: 19.02.2024).

12. С чего начинается тестирование: что такое тест-кейс, зачем он нужен и как его писать [Электронный ресурс]. – URL: https://practicum.yandex.ru/blog/chto-takoe-test-keys-i-kak-ego-sostavit/ (дата обращения: 21.04.2024).

13. Серверные языки программирования [Электронный ресурс]. – URL: https://web-creator.ru/articles/server_side_languages (дата обращения: 29.03.2024).

14. Содержание и структура сферы жилищно-коммунального хозяйства [Электронный ресурс]. – URL: https://cyberleninka.ru/article/n/soderzhanie-i-struktura-sfery-zhilischno-kommunalnogo-hozyaystva (дата обращения: 29.01.2024).

15. Создаём первое веб-приложение с Джанго [Электронный ресурс]. – URL: https://tproger.ru/translations/create-your-first-django-app (дата обращения: 29.02.2024).

16. Устройства сбора и передачи данных (УСПД) [Электронный ресурс]. – URL: https://www.saures.ru/ (дата обращения: 19.02.2024).

17. Формирование требований и классификация требований [Электронный ресурс] // Бизнес-анализ в России. – URL: https://analytics.infozone.pro/formation-requirements-and-classification-requirements/ (дата обращения: 19.02.2024).

18. Шор, Александр Михайлович. Сравнительный анализ подходов в разработке API веб-приложений // StudNet. 2020. №9. [Электронный ресурс]. – URL: https://cyberleninka.ru/article/n/sravnitelnyy-analiz-podhodov-v-razrabotke-api-veb-prilozheniy (дата обращения: 22.02.2024).

19. Эспозито, Д. Разработка современных веб-приложений: анализ предметных областей и технологий. – М.: Вильямс И.Д., 2017.

20. Difference between MVC and MVT design patterns [Электронный ресурс]. – URL: https://www.geeksforgeeks.org/difference-between-mvc-and-mvt-design-patterns/ (дата обращения: 29.03.2024).

21. Django vs Flask: Which is the Best Python Web Framework [Электронный ресурс]. – URL: https://blog.jetbrains.com/pycharm/2023/11/django-vs-flask-which-is-the-best-python-web-framework (дата обращения: 19.02.2024).

22. Hillar, Gaston C. Django RESTful Web Services: The Easiest Way to Build Python RESTful APIs and Web Services with Django. 2018. [Электронный ресурс]. – URL: https://books.google.com/books?hl=en&lr=&id=xNRJDwAAQBAJ&oi=fnd&pg=PP1&dq=Django+for+APIs&ots=-afQobhjpO&sig=t0_pDX5mrnYtlIVj35qwQ3xAWAE (дата обращения: 23.02.2024).

23. Introduction to Spring Framework [Электронный ресурс]. – URL: https://docs.spring.io/spring-framework/docs/3.2.x/spring-framework-reference/html/overview.html (дата обращения: 19.02.2024).

24. The web framework for perfectionists with deadlines [Электронный ресурс]. – URL: https://www.djangoproject.com/ (дата обращения: 23.02.2024).

25. Using SQLite as a Database Backend in Django Projects [Электронный ресурс]. – URL: https://medium.com/@codewithbushra/using-sqlite-as-a-database-backend-in-django-projects-code-with-bushra-d23e3100686e (дата обращения: 19.02.2024).

26. What is .NET Framework [Электронный ресурс]. – URL: https://dotnet.microsoft.com/en-us/learn/dotnet/what-is-dotnet-framework (дата обращения: 19.02.2024).

Приложение А

Тест-кейсы для тестирования приложения

Таблица А.1 – Тест-кейс приложения

Описание	Шаги	Входные данные	Ожидаемые результаты	Фактические результаты	Статус
Авторизация с нормальными данными	1. Перейти на страницу авторизации 2. Ввести логин и пароль пользователя 3. Нажать кнопку "Войти"	Логин: user1 Пароль: password123	Успешный вход в личный кабинет пользователя	Согласно ожидаемому	Пройден успешно
Авторизация с ненормальными данными	3. Перейти на страницу авторизации 4. Ввести некорректный логин и пароль пользователя 3. Нажать кнопку "Войти"	Логин: 122e12 Пароль: rwedqwd	Отображение сообщения о неверных учетных данных	Согласно ожидаемому	Пройден успешно
Добавление нормальных показаний	1. Перейти на страницу личного кабинета пользователя 2. Ввести показания приборов учета в соответствующие поля формы 3. Нажать кнопку «Отправить»	Показания: Электричество: 100 Вода: 50 Горячая вода: 30 Газ: 20	Успешное добавление показаний в базу данных	Согласно ожидаемому	Пройден успешно
Добавление показаний ниже нуля	1. Перейти на страницу личного кабинета пользователя 2. Ввести отрицательные значения показаний приборов учета в соответствующие поля формы 3. Нажать кнопку «Отправить»	Показания: Электричество: -50 Вода: -20 Горячая вода: -10 Газ: -5	Отображение сообщения об ошибке ввода данных	Согласно ожидаемому	Пройден успешно

Продолжение таблицы А.1

Описание	Шаги	Входные данные	Ожидаемые результаты	Фактические результаты	Статус
Добавление показаний ниже предыдущих	1. Перейти на страницу личного кабинета пользователя 2. Ввести значения показаний приборов учета меньше предыдущих значений 3. Нажать кнопку «Отправить»	Показания (меньше предыдущего значения): Электричество: 80 Вода: 40 Горячая вода: 20 Газ: 15	Отображение сообщения о некорректных показаниях	Согласно ожидаемому	Пройден успешно
Добавление сообщения собственником	1. Перейти на страницу личного кабинета пользователя 2. Ввести заголовок и текст сообщения 3. Нажать кнопку «Отправить»	Заголовок: Новое сообщение Текст: Привет, это тестовое сообщение	Сообщение успешно добавлено в базу данных	Согласно ожидаемому	Пройден успешно
Выход из кабинета собственником	1. Нажать кнопку «Выйти» на странице личного кабинета	-	Переход на страницу авторизации	Согласно ожидаемому	Пройден успешно
Авторизация администратора	1. Перейти на страницу авторизации 2. Ввести логин и пароль администратора 3. Нажать кнопку «Войти»	Логин: admin Пароль: admin	Успешный вход в кабинет администратора	Согласно ожидаемому	Пройден успешно
Добавление нового собственника администратором	1. Перейти на страницу кабинета администратора 2. Заполнить поля формы для создания нового пользователя 3. Нажать кнопку «Добавить»	№участка\квартиры: 21 Номер: 89023341221 Имя: Андрей Дудин Пароль: password Повтор пароля: password	Новый собственник успешно добавлен в базу данных	Согласно ожидаемому	Пройден успешно

Продолжение таблицы А.1

Описание	Шаги	Входные данные	Ожидаемые результаты	Фактические результаты	Статус
Добавление нового пользователя администратором с некорректными данными	1. Перейти на страницу кабинета администратора 2. Заполнить поля формы для создания нового пользователя некорректными данными 3. Нажать кнопку «Сохранить»	№участка\квартиры: вв Номер: 89023341 Имя: 123 Пароль: 13 Повтор пароля: 123	Отображение сообщения об ошибке ввода данных	Согласно ожидаемому	Пройден успешно
Поиск собственника	1. Перейти на страницу кабинета администратора 2. Ввести логин собственника в поле поиска 3. Нажать кнопку «Поиск»	Имя: Михаил Задорнов	Отображение информации о найденном собственнике	Согласно ожидаемому	Пройден успешно
Удаление сообщений от собственников	1. Перейти на страницу кабинета администратора 2. Выбрать сообщение собственника для удаления 3. Нажать кнопку «Удалить»	-	Сообщение успешно удалено из базы данных	Согласно ожидаемому	Пройден успешно