****

**Частное учреждение профессионального образования**

**«Высшая школа предпринимательства»**

**(ЧУПО «ВШП»)**

ДИПЛОМНЫЙ ПРОЕКТ

по основной образовательной программе  
среднего профессионального образования по специальности  
09.02.07 «Информационные системы и программирование» на тему:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |
| --- | --- | --- |
| Выполнил студент | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *(подпись)* | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *(фамилия, имя, отчество)*  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Руководитель | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *(подпись)* | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *(ученая степень, фамилия, имя, отчество)*  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Рецензент | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *(подпись)* | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *(ученая степень, фамилия, имя, отчество)*  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *(должность)*  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *(место работы)* |
| Председатель ГЭК | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *(подпись)* | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *(ученая степень, фамилия, имя, отчество)*  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Оценка | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *(прописью)* | Дата защиты: \_\_.\_\_.20\_\_г. |

Оглавление

[Введение. 3](#_Toc199711629)

[Глава 1. Теория о разработке базы данных и платформы 5](#_Toc199711630)

[**1.1 Анализ предметной области** 5](#_Toc199711631)

[**1.2 Анализ существующих решений** 6](#_Toc199711632)

[**1.3 Требования к платформе** 7](#_Toc199711633)

[**1.4 Теоретические основы баз данных** 9](#_Toc199711634)

[**1.5 Требования для базы данных** 11](#_Toc199711635)

[**1.6 Проектировка схемы базы данных** 12](#_Toc199711636)

[**1.7 Проектирование сайта.** 14](#_Toc199711637)

[**1.8 Основные риски при разработке платформы** 17](#_Toc199711638)

[**1.9 Технологии безопасности при разработке сайтов** 19](#_Toc199711639)

[Глава 2. Проектирование и разработка платформы. 21](#_Toc199711640)

[**2.1 Определение функциональных требований** 21](#_Toc199711641)

[**2.2. Разработка архитектуры и структуры системы** 23](#_Toc199711642)

[**2.3. Реализация интерфейса пользователя** 26](#_Toc199711643)

[**2.4 Реализация интерфейса администратора** 33](#_Toc199711644)

[**2.5 Реализация ключевых функций** 37](#_Toc199711645)

[Заключение 41](#_Toc199711646)

[Список источников 43](#_Toc199711647)

[Приложение 1 45](#_Toc199711648)

[Приложение 2 46](#_Toc199711649)

[Приложение 1 47](#_Toc199711650)

# Введение.

**Актуальность**

В настоящее время информационные технологии играют ключевую роль в управлении и оптимизации процессов, в том числе управление недвижимостью чем в свою очередь занимается ТСЖ.

Что такое ТСЖ? Товарищество Собственников Жилья (ТСЖ) - это некоммерческая организация, созданная собственниками помещений в многоквартирном доме для обслуживания своего дома. Основная цель ТСЖ - эффективное и качественное содержание общего имущества в доме, включая проведение ремонта, уборки территорий, поддержка работоспособности лифтов и многое другое. [16, с.61, ст. 135]  
 Разработка платформы для ТСЖ актуальна по трем причинам. Во-первых, это обеспечение минимизации человеческого фактора в ошибках, учете и расчетах. Во-вторых, это прозрачность и доступность информации для участников управления ТСЖ. В-третьих, это улучшение качества и эффективности управления сразу несколькими объектами.

**Цель**

Целью данной дипломной работы является разработка платформы для ТСЖ, которая предназначена для автоматизации работы в товариществе собственников жилья.

**Задачи**

Для достижения цели необходимо решить несколько задач:

1. Изучить существующие решения и проанализировать потребность ТСЖ в информационных системах

2. Спроектировать схему базы данных, отражающую основную деятельность ТСЖ

3. Разработать базу данных

4. Заполнить базу данных

5. Проверить базу на недочеты и ошибки

6. Разработка платформы для управления тсж

7. Поиск уязвимостей и доработка функционала

8. Тестирование готовой платформы

6. Разработка рекомендаций по внедрению и использованию платформы

**Объект исследования**

Объектом является платформа управления ТСЖ

**Предмет исследования**

Предметом исследования является платформа для управления товариществом собственников жилья

# Глава 1. Теория о разработке базы данных и платформы

## **1.1 Анализ предметной области**

Основная задача ТСЖ — организация и координация технического обслуживания и содержания жилого фонда. Это включает в себя:

* учёт платежей жильцов;
* контроль состояния объектов недвижимости;
* взаимодействие с подрядными организациями;
* обеспечение прозрачной отчетности перед жильцами;
* планирование и организация ремонтных работ;
* хранение и обновление юридической, технической и финансовой информации.

Современные реалии показывают, что ручное ведение учета или применение устаревших решений приводит к множеству проблем: от ошибок в расчетах до потери или утечки данных. Также возникает необходимость в централизованной системе, обеспечивающей доступность информации для всех участников процесса, с возможностью разграничения прав доступа.

Особое внимание необходимо уделить следующим аспектам предметной области:

* Характеристики организаций (ТСЖ) — наличие уникальных реквизитов, таких как ИНН, ОГРН, юридический адрес, расчетный счет и т.д.
* Учет жильцов и объектов недвижимости — каждая квартира и дом должны быть связаны с конкретным ТСЖ, жильцами, их статусами и платежами.
* Интерактивность — жильцы должны иметь возможность направлять заявки, получать уведомления, видеть историю платежей и другую важную информацию.
* Защита персональных данных — в системе обрабатывается чувствительная информация, что требует соответствия требованиям законодательства в области ИТ-безопасности.

## **1.2 Анализ существующих решений**

Во время разработки и внедрения системы для Товарищества Собственников Жилья (ТСЖ) возникают различные проблемы, которые требуют внимания и анализа. Решение этих проблем способствует эффективности решений и оптимизации управления имуществом. На первый взгляд можно выделить несколько основных проблем:

1. **Безопасность данных.** Разработка базы данных подразумевает работу с большим объемом личной информации о собственниках. Обеспечение надежной защиты данных от несанкционированного доступа является одной из проблем.
2. **Масштабируемость системы.** Необходимо заранее предусмотреть возможность масштабирования системы для обеспечения ее эффективной работы с увеличением объема данных и количества пользователей.
3. **Учет законодательных требований.** Система должна соответствовать всем действующим законодательным и нормативным требованиям, включая правила обработки и защиты данных

Уже существуют готовые решения проблем. Вот несколько из них:

1. 1С:Учет в управляющих компаниях ЖКХ, ТСЖ и ЖСК — популярное решение, которое предлагает широкий спектр функциональных возможностей для автоматизации процессов учета и управления в ТСЖ. Система позволяет вести учет финансов, контролировать платежи жильцов, планировать бюджет и отслеживать выполнение ремонтных работ. [1]
2. Домовой — это еще одна система, которая ориентирована на упрощение взаимодействия между управляющими компаниями и жителями многоквартирных домов. Предлагает инструменты для онлайн-голосований, обсуждений и опросов среди жильцов, а также функции для учета и контроля платежей. [12]
3. ГИС.ЖКХ — это официальный сайт, где можно найти информацию о вашей управляющей компании, тарифах на коммунальные услуги, задолженностях по оплате и другую важную информацию. Сервис, акцентирующий внимание на прозрачности и контроле расходов на жилищно-коммунальные услуги. Предоставляет жильцам детальную отчетность по использованным ресурсам и выполненным работам [10].

При анализе существующих решений можно выявить у этих программ ряд недостатков:

1. Многие системы имеют заранее готовые набор функций, которые не всегда полностью соответствуют специфике работы ТСЖ
2. Некоторые программные продукты требуют значительных усилий для внедрения и обучения персонала, что может стать препятствием для маленьких ТСЖ
3. Цена на лицензии и обслуживание программного обеспечения может быть достаточно высока, что делает его недоступным для ТСЖ с ограниченным бюджетом.

## **1.3 Требования к платформе**

Платформа предназначена для автоматизации процессов управления ТСЖ. Для обеспечения эффективности система должна содержать в себе ряд функциональных, технических и информационных требований.

1. Функциональные требования:

* 1. Платформа должна обеспечивать:
  + учёт жилых домов, квартир, жильцов и связанных с ними данных;
  + ведение базы платежей, включая статусы оплат и суммы;
  + учёт и обработку заявок на обслуживание от жильцов;
  + разграничение доступа по ролям: администратор, владелец ТСЖ, управляющий, пользователь;
  + возможность отправки приглашений новым пользователям для подключения к ТСЖ;
  + просмотр и редактирование информации в соответствии с правами доступа;
  + экспорт данных (заявки, платежи) в формате PDF.
  1. Интерфейс системы должен быть интуитивно понятным и доступным для всех категорий пользователей, с адаптивной версткой для корректного отображения на различных устройствах.

2. Информационные требования:

* В базе данных должны храниться следующие сведения:
  + данные о ТСЖ (ИНН, ОГРН, наименование, юридический адрес, расчетный счет, БИК и др.);
  + информация о жилых домах и квартирах (этажность, подъезды, количество квартир и их параметры);
  + учет жильцов (ФИО, контактные данные, паспортная информация и регистрация);
  + история платежей с детализацией по жильцам, статусам и датам;
  + информация о заявках на обслуживание (описание, дата, статус выполнения);
  + действия пользователей (журнал изменений и активности).

3. Технические требования:

* Для хранения данных должна быть использована **реляционная база данных**, обеспечивающая надежную структуру таблиц и связей между сущностями.
* Система должна быть масштабируемой, с возможностью поддержки нескольких ТСЖ и большого объема записей.
* Архитектура должна предусматривать клиент-серверное взаимодействие, с безопасным обменом данными по HTTPS.
* Поддержка сессий и авторизации на стороне сервера с использованием защищённого хранения паролей (например, bcrypt).
* Реализация защиты от типовых уязвимостей (SQL-инъекции, XSS, CSRF).

4. Требования к безопасности и управлению доступом:

* Должна быть реализована система ролей, позволяющая ограничивать и контролировать доступ к определённым разделам и функциям платформы.
* Пользователи с разными правами (например, администратор и обычный пользователь) должны видеть только доступные им функции.
* Необходима реализация механизма ведения журнала действий пользователей, позволяющего отслеживать изменения, произведённые в системе.

**1.4 Теоретические основы баз данных**

Базы данных можно разделить на несколько основных типов. Вот основные виды баз данных:

1. **Реляционные базы данных.**   
   Реляционная модель данных была предложена сотрудником фирмы IBM Эдгаром Коддом и основывается на понятии отношение (relation). Отношение представляет собой множество элементов, называемых кортежами. Подробно теоретическая основа реляционной модели рассматривается в следующем разделе. Наглядной формой представления отношения является привычная для человеческого восприятия двумерная таблица. Таблица имеет строки (записи) и столбцы (колонки). Каждая строка таблицы имеет одинаковую структуру и состоит из полей. Строкам таблицы соответствуют кортежи, а столбцам атрибуты отношения. С помощью одной таблицы удобно описывать сведения о группах однородных (имеющих одинаковые свойства) объектов, явлений или процессов реального мира. Каждая строка таблицы содержит сведения о конкретном объекте, явлении или процессе. Строка (запись) имеет одинаковую структуру и описывает с помощью полей свойства объектов. [15, с.37] что дает возможность легко устанавливать взаимосвязь между элементами данных.
2. **Нереляционные базы данных.**

NOSQL ‒ хранилища данных. с самого начала была спроектирована для управления огромными распределенными наборами данных, запрос к которым должен был выполняться за миллисекунды. Вместо нормализации данных по реляционным таблицам, которые затем должны объединяться для ответов на запросы, сверхбольшие массивы данных распределяются по сотням или тысячам процессоров, организованных так, чтобы связанные данные располагались рядом. Запросы выполняются параллельно на всех процессорах; каждый возвращает ответы, основываясь на своих локальных данных. Этот простой и масштабируемый подход оказался очень эффективным и гибким, и он позволяет совместно хранить данные самых разных типов, а также выполнять сложные запросы. [19, с.27] Данные в них хранятся в формате, отличном от реляционных таблиц. Однако к нереляционным базам данным можно обращаться с помощью идиоматических языковых API, языков декларативно структурированных запросов и языков запросов по примеру, поэтому их также называют базами данных «не только SQL».

1. **Иерархические базы данных**  
   Здесь между хранимыми объектами устанавливаются связи. Объекты делятся на родителей (основные классы или категории объектов) и потомков (экземпляры этих классов или категорий). При этом у каждого потомка может быть не более одного родителя. [9]
2. **Объектно-ориентированные базы данных**  
   Базы данных, в которых информация представлена в виде объектов, как в объектно-ориентированных языках программирования. [8]

Существует также не мало СУБД, но можно выделить лишь несколько интересных:

1. **MySQL -** свободно распространяемая серверная реляционная СУБД. В настоящее время является наиболее популярной в веб-программировании. [13, с.194 - 13.2]
2. **PostgreSQL(postgres) -** объектно-реляционная система управления базами данных с открытым исходным кодом. На первом месте в ней стоит расширяемость, техническое совершенство и совместимость. [11. с.45]
3. **MongoDB -** это ориентированная на документы база данных NoSQL с открытым исходным кодом, которая использует для хранения структуру JSON. Модель данных MongoDB позволяет представлять иерархические отношения, проще хранить массивы и другие более сложные структуры. [4]
4. **SQLite -** это встраиваемая система управления базами данных для клиент-серверных приложений. Эта СУБД отличается компактностью — она занимает один файл, благодаря чему встраивается в приложения, чтобы выполнять функции СУБД автономно, без использования сервера. Также SQLite применяется в веб-разработке небольших проектов. [20]

**1.5 Требования для базы данных**

Исходя из ранее исследованных типов баз данных, я выберу реляционные базы данных. Они достаточно популярные, подойдут для создания таблиц и связей между сущностями.

В базе данных будет храниться информация о домах, квартирах, счетах и жильцах. Одной из важных особенностей является масштабируемость базы. Так как внутри ТСЖ может быть не один дом, а несколько поэтому база данных должна быть подготовлена для этого

База данных должна отображать основные характеристики дома, квартиры, жильца. У каждого дома есть общие данные – это количество этажей, количество подъездов, количество квартир и год постройки дома. Данные, связанные с наличием подвала, площади целого дома, не будут учитываться.

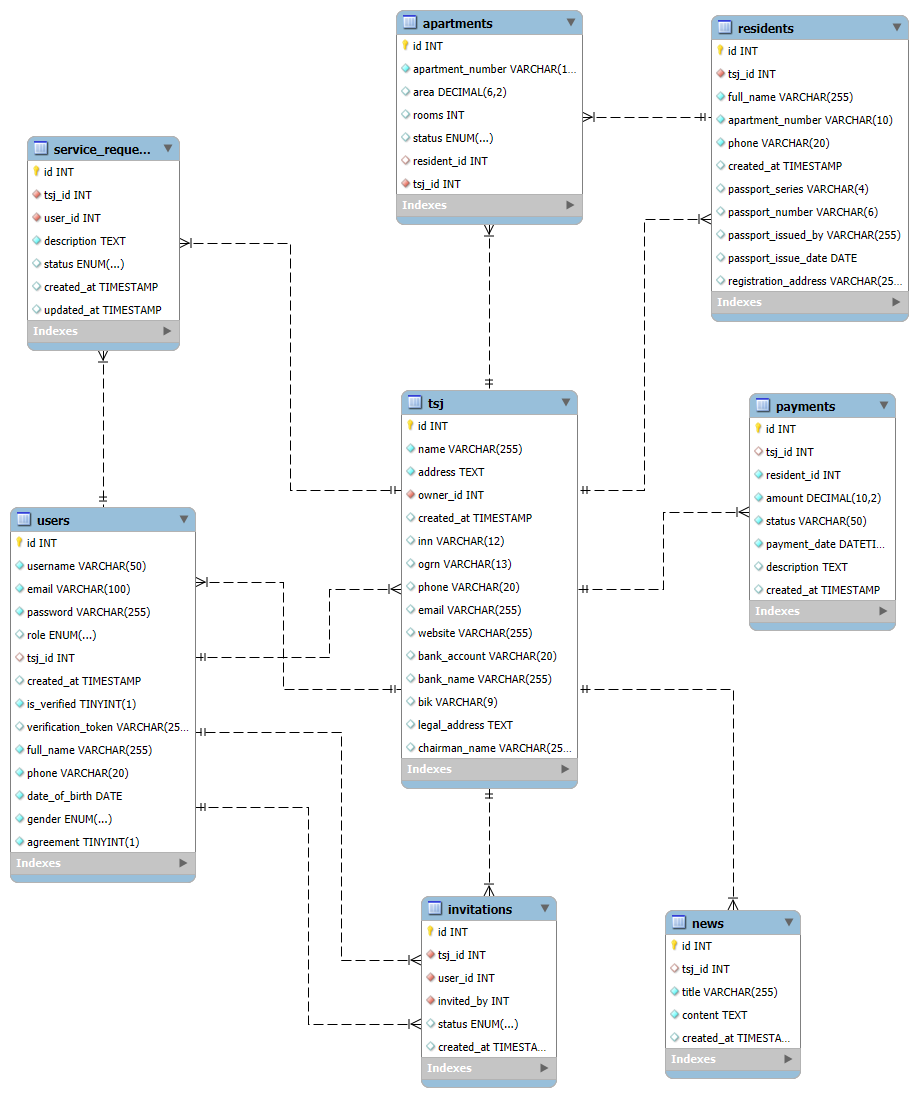
Удобное отслеживание и управление домом – это один из главных критериев для разработки базы данных. Такая база данных рассчитана на использование ее персоналом жилого фонда. Применение базы данных позволит повысить эффективность работы персонала по эксплуатации жилого фонда, снизить затраты времени на получение и обработку данных по жилому фонду и повысить эффективность работы. Правильная организация работы персонала обеспечивает точные и достоверные данные, которые способствуют снижению затрат на содержание жилого фонда, точные расчеты материалов, время и стоимость работ по содержанию жилищного фонда

Так же необходимо создать роли для управления этой базой данных. Какие-то пользователи смогут управлять, а какие-то только просматривать. Без создания ролей любой пользователь сможет не санкционированно вносить изменения в базу данных, что не сходится с одной из основных проблем.

## **1.6 Проектировка схемы базы данных**

Перед началом создания самой базы данных я построил схему, отражающую «сущность-связь». Такую схему называют ER-диаграмма или ERD. ER-диаграмма - это разновидность блок-схемы, где показано, как разные «сущности» (люди, объекты, концепции и так далее) связаны между собой внутри системы [17]

Для составления схемы базы данных я воспользуюсь функцией создания моделей в MySQL Workbench.



 В данной схеме (рис. 1) готовая спроектированная база данных на основе требований, составленных ранее.

В схеме находится 8 таблиц каждая из которых реализует или помогает реализовать поставленную задачу:

1. tsj [ТСЖ] – таблица которая содержит в себе основные данные про ТСЖ
2. residents [жильцы] - таблица, которая содержит жильцов, проживающих в определённой квартире, принадлежащей определённому ТСЖ.
3. apartments [квартиры] – таблица, которая содержит информацию о квартирах и о том, какому ТСЖ они принадлежат.
4. users [пользователи] – поскольку в моей платформе есть возможности: создавать ТСЖ, приглашать работников в созданное ТСЖ, данная таблица необходима для хранения информации о пользователях и о том, в каких ТСЖ они состоят.
5. invitations [приглашения] – таблица, которая содержит информацию о том, какой пользователь пригласил и от какого ТСЖ было отправлено приглашение, а также статус: принял ли пользователь приглашение или нет.
6. payments [платежи] – таблица, которая содержит в себе данные про выставленные платежи.
7. service\_request [Запросы на ремонт] - таблица, которая содержит в себе данные про предстоящие работы и их статус.
8. news [новости] - таблица, которая содержит в себе данные про новости на главной странице

## **1.7 Проектирование сайта.**

Перед выбором стека технологий были изучены существующие решения, используемые для разработки сайтов аналогичной направленности — например, системы управления контентом (CMS), такие как WordPress, Joomla или Bitrix. Эти платформы предоставляют удобные визуальные редакторы и модули для авторизации, обработки форм, управления пользователями и базами данных. Однако, несмотря на их функциональность, они часто избыточны для задач ТСЖ, требуют дополнительных ресурсов и настройки, что не всегда оправдано для небольших или специализированных решений.

В связи с этим было принято решение создать сайт "с нуля", используя лёгкие и гибкие технологии, которые обеспечивают как простоту разработки, так и гибкость в дальнейшем сопровождении и масштабировании проекта.

**Для реализации сайта выбраны следующие технологии:**

* **HTML и CSS** - базовые языки для создания структуры веб-страниц и их визуального оформления. Они позволяют создать читаемый, логичный и стандартизированный интерфейс.
* **Bootstrap** – это бесплатный фреймворк с открытым исходным кодом для создания адаптивных интерфейсов сайтов и веб-приложений. Он позволяет верстать сайты во много раз быстрее и проще, чем на «чистом» JavaScript и CSS, при этом он обеспечивает единство стилей и понимание кода, что упрощает работу в команде.

[2]

**PHP —** серверный язык программирования, который используется для веб-разработки. Он отвечает за бэкенд — та часть продукта, которую не видит пользователь [21]

**MySQL** (в связке с **phpMyAdmin**) — реляционная система управления базами данных для хранения информации о жильцах, квартирах, начислениях, платежах и обращениях.

**phpMyAdmin** — **веб-приложение с открытым кодом, написанное на языке PHP и представляющее собой веб-интерфейс для администрирования СУБД MySQL.**

[5] У него удобный веб-интерфейс для администрирования MySQL, который используется для создания таблиц, настройки прав доступа и выполнения SQL-запросов.  
Выбор этих технологий обусловлен их распространенностью, надежностью, большим количеством обучающих материалов и поддержкой сообществом. Кроме того, они являются бесплатными и хорошо сочетаются между собой, что упрощает как разработку, так и дальнейшую эксплуатацию сайта.

**Структура сайта**

Сайт будет состоять из нескольких ключевых разделов, каждый из которых реализуется с использованием выбранных ранее технологий - HTML, CSS, Bootstrap для внешнего вида и структуры, PHP для серверной логики, а также MySQL для хранения данных. Страницы будут динамически подгружать и отображать информацию в зависимости от роли пользователя и состояния системы.

**Основные разделы:**

* **Главная страница** Содержит общую информацию о платформе ТСЖ, краткое описание возможностей, контактные данные, а также формы для входа и регистрации. Разметка страницы выполнена с помощью HTML и Bootstrap, что обеспечивает адаптивность дизайна. Форма входа обрабатывается PHP-скриптом, который проверяет данные через базу MySQL.
* **Панель администратора** Предоставляет доступ к управлению базой данных жильцов, платежами, домами, подъездами, а также заявками и обращениями. Интерфейс построен с использованием Bootstrap-компонентов (таблиц, карточек, кнопок), а вся логика работы - на PHP с прямым взаимодействием с базой данных.
* **Панель работника** Позволяет просматривать данные о начислениях, истории платежей, а также создавать или отслеживать заявки на техническое обслуживание. Данные выводятся из базы с помощью SQL-запросов, а интерфейс отображается динамически. Применение PHP позволяет разграничить доступ к функциям в зависимости от роли пользователя.
* **Регистрация и авторизация** Реализованы с валидацией форм, подтверждением e-mail и возможностью восстановления пароля. Все введенные данные проверяются на стороне сервера, затем записываются в базу данных через PHP. Для безопасности используются стандартные практики обработки паролей (шифрование и защита от SQL-инъекций).

**Архитектура сайта**

Сайт будет реализован по принципу клиент-серверной архитектуры:

* Клиентская часть будет отвечать за отображение информации и взаимодействие с пользователем.
* Серверная часть — за обработку данных, их хранение и передачу.
* Для безопасного доступа пользователей будут предусмотрены сессии, авторизация по логину и паролю, а также разграничение прав.

**Требования к интерфейсу**

Интерфейс должен соответствовать следующим требованиям:

* Простой и понятный дизайн с минималистичным стилем.
* Поддержка адаптивности — корректное отображение на мобильных устройствах.
* Безопасность — защита от SQL-инъекций, XSS-атак и несанкционированного доступа.

 Таким образом, проектирование сайта основывается на использовании проверенных технологий, которые обеспечивают надежность, безопасность и удобство работы для всех категорий пользователей платформы.

## **1.8 Основные риски при разработке платформы**

Разработка информационной платформы для управления ТСЖ связана с рядом рисков, которые необходимо учитывать на этапе проектирования и реализации системы. Игнорирование возможных угроз может привести к снижению надёжности, безопасности или функциональности проекта.

Вот основные риски и пути их решения:

1. Угрозы безопасности данных

Платформа будет обрабатывать персональные данные жильцов, включая контактную информацию, данные о платежах и собственности. Нарушение конфиденциальности информации может привести к серьёзным юридическим последствиям.

Способ защиты: использование надёжного хэширования паролей, защита от SQL-инъекций, настройка HTTPS-соединений.

1. Потеря данных

В случае сбоев оборудования или программных ошибок существует риск потери критически важной информации.

Способ защиты: регулярное резервное копирование баз данных, внедрение системы аварийного восстановления данных.

1. Масштабируемость платформы

С увеличением числа пользователей система может столкнуться с проблемами производительности.

Способ защиты: правильное проектирование базы данных, использование кеширования, планирование возможного перехода на более производительные серверы.

1. Ошибки в логике работы системы

Неправильная реализация процессов может привести к некорректной работе платформы, проблемам при учёте платежей или заявок.

Способ защиты: тщательное тестирование всех модулей перед вводом в эксплуатацию, написание юнит-тестов для ключевых функций.

1. Недостаточная защита авторизации и аутентификации

При отсутствии надлежащих мер злоумышленники могут получить доступ к личным кабинетам пользователей.

Способ защиты: многоуровневая система аутентификации, контроль времени сессии, ограничение количества неудачных попыток входа.

1. Низкая удобство использования (UX)

Сложный или неудобный интерфейс может отпугнуть пользователей, сделав систему неэффективной.

Меры защиты: проведение тестирования интерфейса на целевой аудитории, создание понятной навигации, минимизация количества действий для выполнения основных задач.

1. Несоответствие требованиям законодательства

Нарушение норм обработки персональных данных (например, в рамках закона №152-ФЗ «О персональных данных») может привести к штрафам и запрету использования системы.

Меры защиты: соблюдение требований законодательства при проектировании хранения и обработки данных, консультирование с юристами.

## **1.9 Технологии безопасности при разработке сайтов**

При разработке веб-платформы для ТСЖ одним из важнейших аспектов является защита данных и обеспечение безопасности этих данных. Так же потребуется предотвращать все несанкционированные доступы к данным.

Платформа работает с персональными данными жильцов, поэтому потребуется соблюдение высоких стандартов защиты.

Вот основные методы защиты:

1. Защита от SQL-инъекций

SQL-инъекция – это уязвимость веб-безопасности, которая позволяет злоумышленнику вмешиваться в запросы, которые приложение делает к своей базе данных. [6]

Пути защиты:

* Использование подготовленных выражений (prepared statements) в PHP.
* Экранирование пользовательских данных перед выполнением SQL-запросов.
* Ограничение прав учетных записей базы данных.

1. Защита от XSS-атак

XSS-атака – это тип уязвимости, встречающийся в web приложениях. XSS атаки позволяют внедрить вредоносный скрипт (или как его еще часто называют эксплойт) на страницу приложения, в результате чего у пользователей, посещающих эту страницу, могут украсть данные разной степени чувствительности: куки, сессионные токены, логины с паролями и просто личную информацию о пользователе. [7]

Пути защиты:

* Очистка и экранирование всех данных, выводимых в HTML.
* Запрет на вставку непроверенного HTML или JavaScript кода.
* Использование Content Security Policy (CSP) для ограничения выполнения скриптов.

1. Защита от CSRF-атак

CSRF-атака – это атака на авторизованного пользователя сайта или веб-приложения, при которой злоумышленник обманом заставляет его выполнить опасное или некорректное действие. [3]

Пути защиты:

* Использование токенов защиты (CSRF-токенов) в формах.
* Проверка подлинности запросов на стороне сервера.

1. Использование защищённого соединения (HTTPS)

HTTPS (расшифровывается как HyperText Transfer Protocol Secure) – это протокол безопасной передачи гипертекста. Это расширение протокола HTTP, но с уровнем защиты. Если сформулировать проще, протокол шифрует все, что вы вводите на сайте, чтобы злоумышленники не могли перехватить данные. [18]

Таким образом, реализация описанных технологий безопасности позволит обеспечить защиту пользовательских данных, минимизировать риски атак на веб-платформу и соответствовать требованиям законодательства в области информационной безопасности.

# Глава 2. Проектирование и разработка платформы.

## **2.1 Определение функциональных требований**

Разрабатываемая информационная система предназначена для автоматизации управления товариществом собственников жилья (ТСЖ). Система должна обеспечивать централизованное управление жильцами, квартирами, заявками и платежами с возможностью разграничения прав пользователей.

Цель системы:

Целью системы является предоставление современного и удобного веб-инструмента для:

* ведения электронного реестра жильцов;
* учета квартир и их состояния (занята / свободна);
* обработки заявок жильцов на обслуживание;
* учета платежей с детализацией по жильцам и статусам;
* обеспечения простого интерфейса для всех категорий пользователей: администратора, владельца, управляющего и жильца.

Ключевые функции:

1. Авторизация и аутентификация:

* Регистрация и вход в систему
* Поддержка сессионного входа через $\_SESSION
* Перенаправление неавторизованных пользователей на страницу входа
* Разграничение доступа по ролям
* Приглашения для подключения жильцов к ТСЖ (механизм invitations)

1. Управление пользователями:

* Просмотр профиля
* Отображение роли пользователя
* Отображение активного приглашения
* Переход в админ-панель или панель управления ТСЖ осуществляется в зависимости от роли.

1. Работа с жильцами:

* Добавление жильца
* Редактирование информации о жильце
* Валидация полей
* Связь жильца с квартирой
* Ограничение доступа к функциям только для владельцев и менеджеров.

4. Работа с квартирами:

* Добавление новой квартиры
* Привязка квартиры к конкретному жильцу
* Указание номера, площади, количества комнат
* Редактирование статуса (свободна / занята)
* Управление всеми квартирами текущего ТСЖ

5. Учет и управление заявками:

* Отправка заявки на обслуживание пользователем
* Редактирование статуса заявки
* Фильтрация заявок по владельцу и статусу
* Экспорт всех заявок в формат PDF (с использованием TCPDF).

6. Учет платежей:

* Добавление нового платежа через форму
* Валидация данных
* Статусы: оплачено, частично, в ожидании, возврат, отклонено;
* Просмотр всех платежей с фильтрацией
* Экспорт данных в PDF по текущему ТСЖ.

7. Панель управления ТСЖ:

* Общая статистика: количество жильцов, суммарные платежи, заявки;
* Быстрые ссылки на модули добавления/просмотра данных;
* Панель доступна только владельцу, администратору и менеджеру, связанному с ТСЖ.

Функциональные роли пользователей

|  |  |
| --- | --- |
| **Роль** | **Возможность** |
| Admin | Имеет полный доступ ко всем ТСЖ и действиям. Может просматривать и управлять всеми заявками, платежами, квартирами. |
| Owner | Владельцы могут управлять только своим ТСЖ: жильцами, квартирами, платежами, заявками. |
| Manager | Управляющие также имеют доступ к управлению конкретным ТСЖ, но не могут создавать другие ТСЖ. |
| User | Простые пользователи могут принимать приглашения, просматривать профиль, отправлять заявки, но не имеют доступа к управлению данными других людей. |

Дополнительные требования

* Все формы должны быть стилизованы с использованием Bootstrap для обеспечения адаптивного и читаемого интерфейса;
* Поддержка экспорта данных в PDF;
* Использование PDO и подготовленных выражений для защиты от SQL-инъекций;
* Предусмотрена валидация как на клиентской, так и на серверной стороне;
* Доступ к страницам осуществляется только при наличии активной сессии.

## **2.2. Разработка архитектуры и структуры системы**

Разрабатываемая система управления ТСЖ построена по **классической архитектуре клиент–сервер**, где фронтенд реализован на HTML/CSS с применением Bootstrap, а серверная часть — на PHP с использованием базы данных MySQL. Такая архитектура обеспечивает гибкость, масштабируемость и достаточную надежность при развертывании в локальной сети или в интернете.

1. **Аутентификация и авторизация**

* login.php – авторизация в системе
* register.php – регистрация в системе
* система приглашений – привязка пользователей к ТСЖ
* таблица users –данные пользователей (см. пункт 1.6)
* profile.php — отображение профиля пользователя и доступа к роли
* разделение на роли: admin, owner, manager, user

1. **Управление ТСЖ – модуль dashboard**
   1. Модуль dashboard/apartment – работа с квартирами:

add\_apartment.php – добавление квартиры в систему

edit\_apartment.php – изменение существующей квартиры

manage\_apartment.php – управление квартирами, изменение статуса

* 1. Модуль dashboard/payment – работа с платежами:

add\_payment.php – добавлеие платежа

view\_payments.php – просмотр всех платежей

* 1. Модуль dashboard/request – работа с заявками:

add\_request.php – добавление заявки

view\_request.php – просмотр всех заявок

1. **Жильцы**

* Таблица residents – данные жильцов (см. пункт 1.6)
* add\_resident.php – создание нового жильца и добавление его к квартире
* edit\_resident.php – редактирование данных жильца
* delete\_resident.php – удаление жильца
* Проверка связки с ТСЖ через tsj\_id
* Связь жильцов с квартирами через resident\_id

1. **Заявки на обслуживание -** Модуль dashboard/request

* Таблица service\_requests – данные заявок (см. пункт 1.6)
* add\_request.php – создание заявки
* view\_requests.php – просмотр заявок
* Статусы: «Ожидание», «В процессе», «Готово», «Отклонена»
* Экспорт заявок в PDF через TCPDF

1. **Платежи-** Модуль dashboard/payment

* Таблица payments – данные платежей (см. пункт 1.6)
* add\_payment.php – создание нового платежа
* view\_payments.php – просмотр всех платежей
* Поддержка фильтрации по жильцу и статусу
* Экспорт платежей в PDF
* Статусы: оплачено, в ожидании, частично оплачено, отклонено, возврат

Взаимосвязь компонентов

Компоненты системы связаны через следующие ключевые элементы базы данных:

tsj.id → residents.tsj\_id, apartments.tsj\_id, payments.tsj\_id, service\_requests.tsj\_id

users.id → service\_requests.user\_id, invitations.user\_id

residents.id → apartments.resident\_id, payments.resident\_id

Таким образом, все действия в системе связаны с конкретным ТСЖ, обеспечивая изолированность данных.

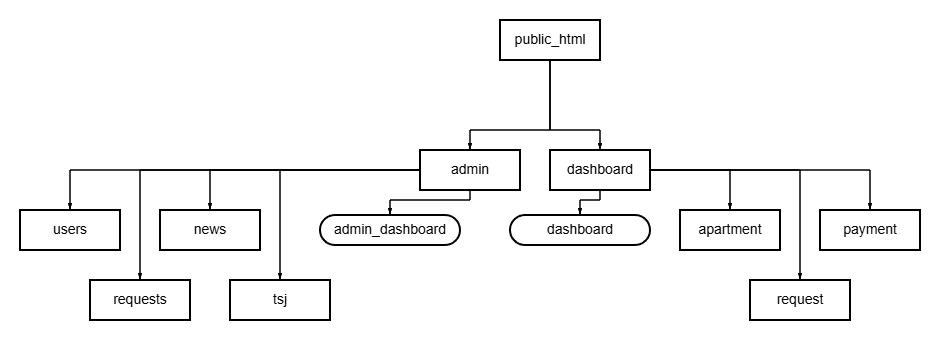
Физическая структура проекта

Рис 2. Структура проекта

## **2.3. Реализация интерфейса пользователя**

Интерфейс системы разработан с учетом принципов адаптивности, доступности и простоты взаимодействия для различных ролей пользователей: администратора, владельца ТСЖ, менеджера и жильца. Для оформления интерфейсов была использована библиотека **Bootstrap 5**, что позволило добиться современного и отзывчивого дизайна без сложных CSS-реализаций.

**Главная панель пользователя (profile.php)**

После входа в систему пользователь попадает на **панель профиля**, где отображается:

* Приветствие в зависимости от времени суток;
* Информация о роли (админ, владелец, пользователь);
* Кнопки для входа в:
  + Админ-панель для администратора;
  + Панель управления ТСЖ — для владельца или менеджера;
  + Приглашение в ТСЖ — для жильца без привязки;
* Кнопка выхода.

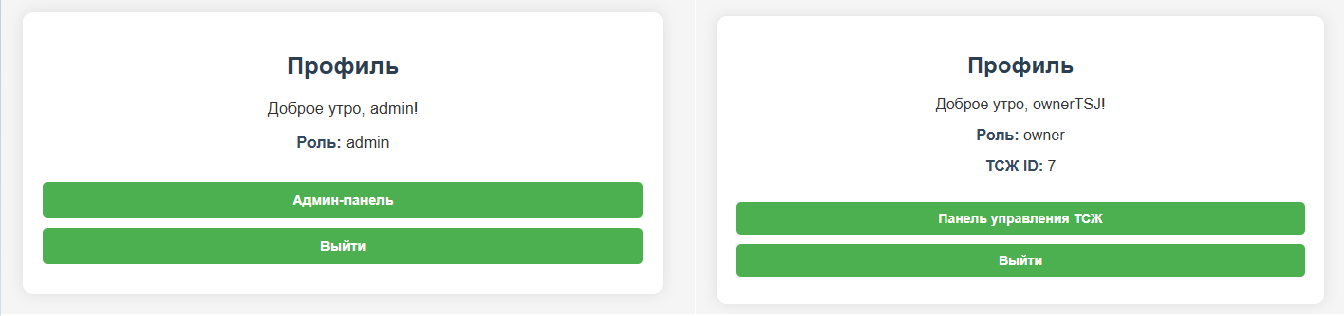


Рис 3 – внешний вид profile.php – для разных пользователей

**Панель управления ТСЖ (dashboard.php)**

Панель позволяет получить общую статистику по текущему ТСЖ:

* Количество жильцов;
* Сумма всех платежей;
* Количество заявок в статусе "ожидание".
* Кнопки управлением ТСЖ

Кнопки быстрого перехода ведут к:

* Добавлению жильца;
* Просмотру платежей;
* Просмотру заявок;
* Управлению квартирами;
* Приглашению новых пользователей.

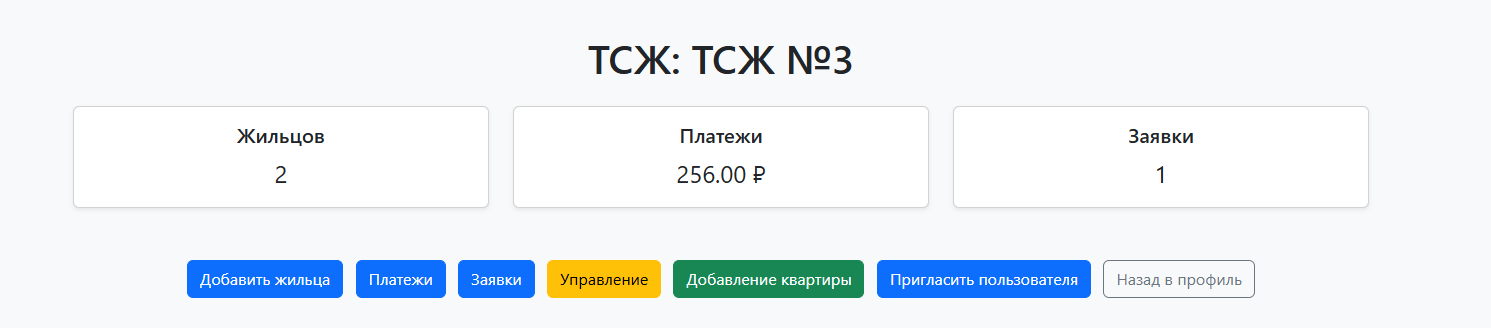


Рис. 4 - внешний вид dashboard.php

**Добавление жильца (dashboard/add\_resident.php)**

Панель позволяет добавить нового жильца в текущее ТСЖ

Что требуется для создания жильца:

* ФИО жильца
* Номер квартиры
* Телефон
* Серия паспорта (опционально)
* Номер паспорта (опционально)
* Кем выдан паспорт (опционально)
* Дата выдачи паспорта
* Адрес регистрации (опционально)

Для быстрого создания некоторые поля опциональные, их можно не вводить и без этого создавать жителя.

**Просмотр платежей (dashboard/payment/view\_payments.php)**

Возможности на данной странице

* Поиск платежа по имени или статусу
* Добавление платежа
* Выгрузка всех платежей в PDF-файл
* Просмотр таблицы с платежами где есть:
  + Дату платежа
  + Плательщика
  + Сумму
  + Статус

Внешний вид см. приложение №1

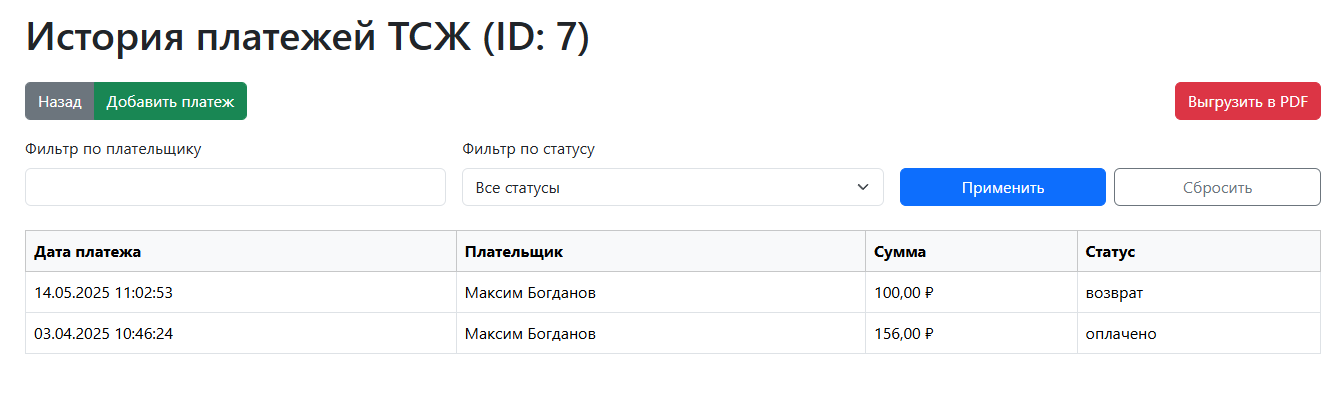


Рис. 5 – внешний вид dashboard/view\_payments.php

**Просмотр созданных заявок на обслуживание (dashboard/request/view\_requests.php)**

Возможности на данной странице

* Поиск заявки по владельцу, статусу
* Добавление заявки (см. Приложение)
* Выгрузка заявок в PDF
* Просмотр таблицы с платежами где есть:
  + Создатель заявки
  + Дата создания
  + Описание проблемы
  + Статус
  + Действие – изменение статуса заявки

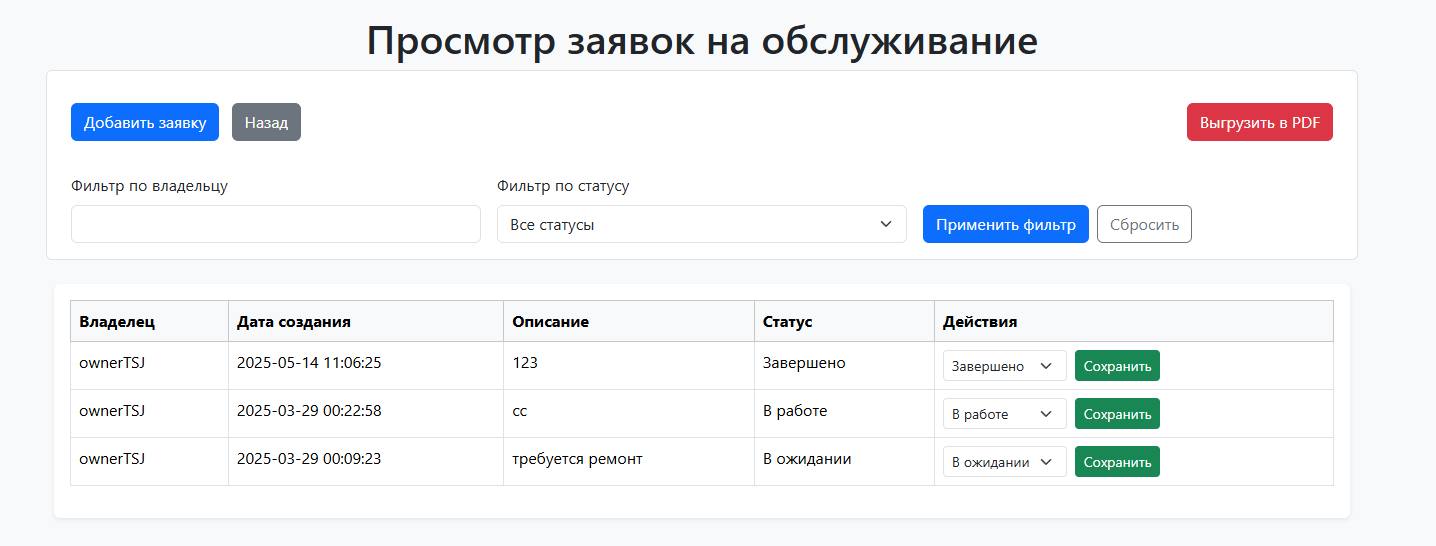
****

Рис. 6 – внешний вид request/view\_requests

**Панель быстрой работы с ТСЖ и жильцами (dashboard/manage\_tsj.php)**

Возможности на данной странице:

* Редактирование и удаление жильцов
  + Редактирование содержит в себе изменение всех данных:
    - ФИО жильца
    - Номер квартиры
    - Телефон
    - Серия паспорта
    - Номер паспорта
    - Кем выдан паспорт
    - Дата выдачи паспорта
    - Адрес регистрации

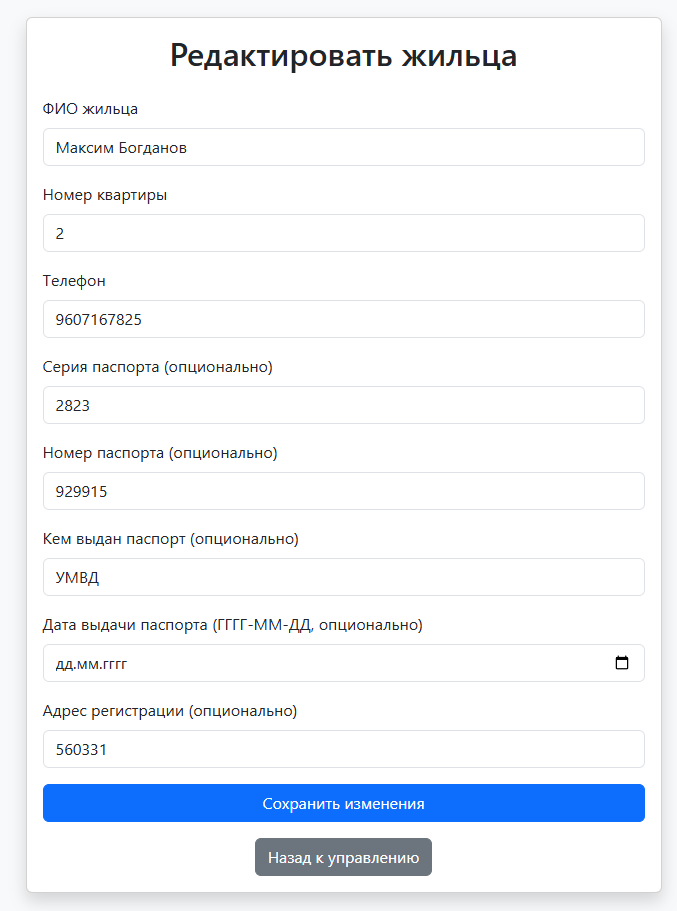


Рис.7 – редактирование пользователя

* Быстрый просмотр заявок (см рис)
* Быстрый просмотр платежей (см рис)
* Удобный просмотр количества жильцов, платежей и заявок

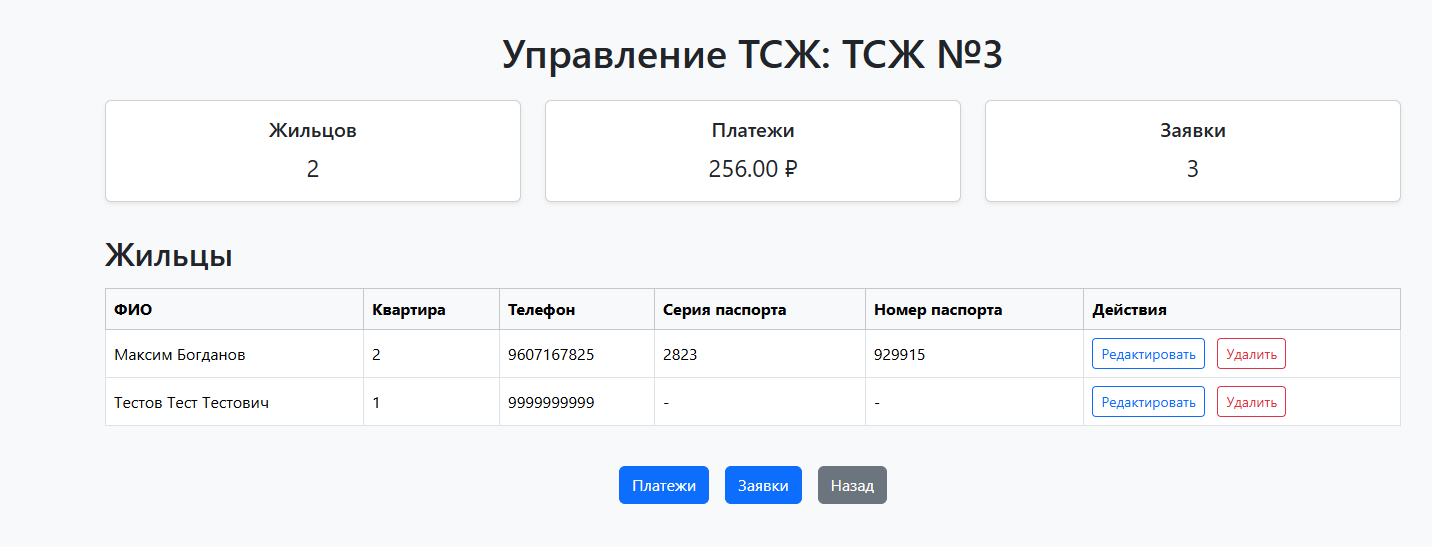
****

Рис.8 – внешний вид dashboard/manage\_tsj.php

**Управление квартирами (dashboard/apartment/manage\_apartments)**

Возможности на данной странице

* Просмотр номера квартиры, площадь, кол-во комнат, статус и кто в ней проживает.
* Изменение статуса квартиры
* Изменение параметров и жильца квартиры

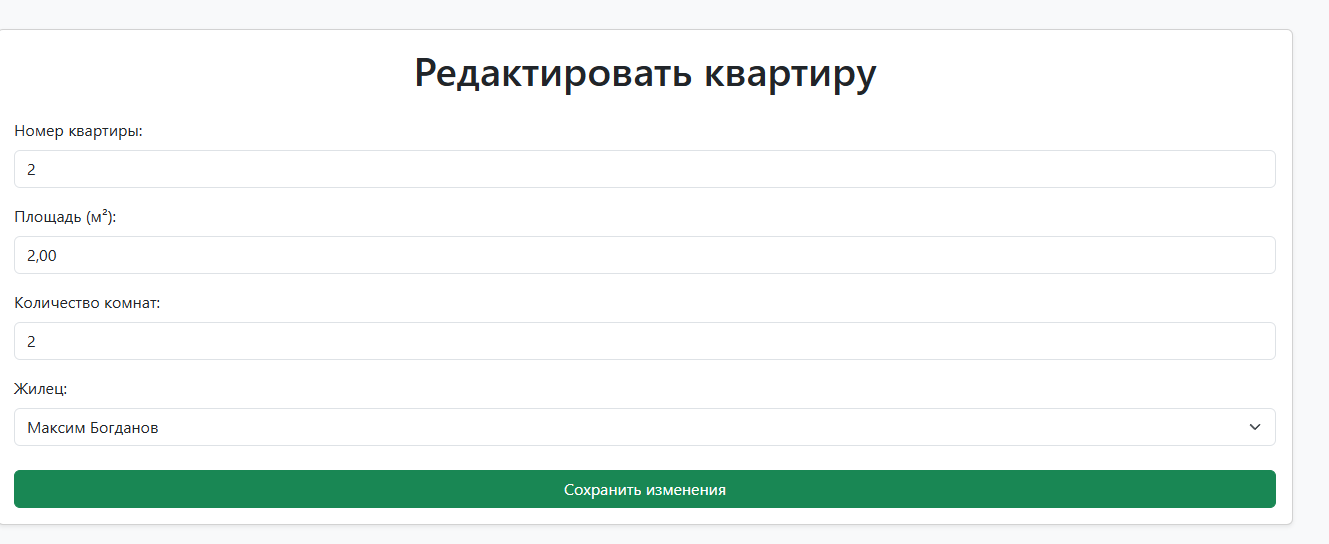


Рис. 9 - Редактирование данных квартиры

**Приглашения пользователя**

Возможности на данной странице:

Указать имя пользователя для приглашения в ТСЖ.

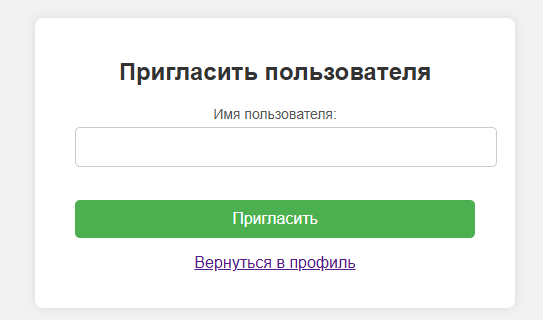


Рис. 10 – внешний вид приглашения в систему ТСЖ

Имя нужно вводить то которое пользователь использует для авторизации в системе. Реализовано таким образом, чтобы тот, кто приглашает не мог видеть всех пользователей, зарегистрированных в системе.

## **2.4 Реализация интерфейса администратора**

Интерфейс администратора разработан с целью предоставить расширенный контроль над системой, включая управление всеми ТСЖ, пользователями и ключевыми модулями проекта. Он обеспечивает доступ к административным функциям, скрытым от обычных пользователей, владельцев и жильцов.

Доступ к интерфейсу

Доступ к админ-панели (admin\_dashboard.php) осуществляется из профиля пользователя, если его роль в сессии ($\_SESSION['role']) равна admin. На странице profile.php в зависимости от роли отображается кнопка перехода в админ-панель:

<?php if ($role === 'admin'): ?>

<a href="admin/admin\_dashboard.php" class="btn btn-primary">Админ-панель</a>

<?php endif; ?>

Структура интерфейса администратора

Интерфейс разработан с использованием фреймворка Bootstrap 5, что обеспечивает:

* Адаптивную вёрстку;
* Быстрый отклик;
* Унифицированный стиль всех элементов управления.

Основные разделы админ-панели:

* Просмотр и управление всеми пользователями;
* Управление ТСЖ;
* Обзор всех заявок и платежей;
* Управление новостями платформы.

**Управление платформой – модуль admin/**

1. **admin\_dashboard.php** – главное меню администратора из которого можно попасть в другие панели. Данная панель не доступна обычному пользователю, доступ есть только у пользователя с ролью admin.

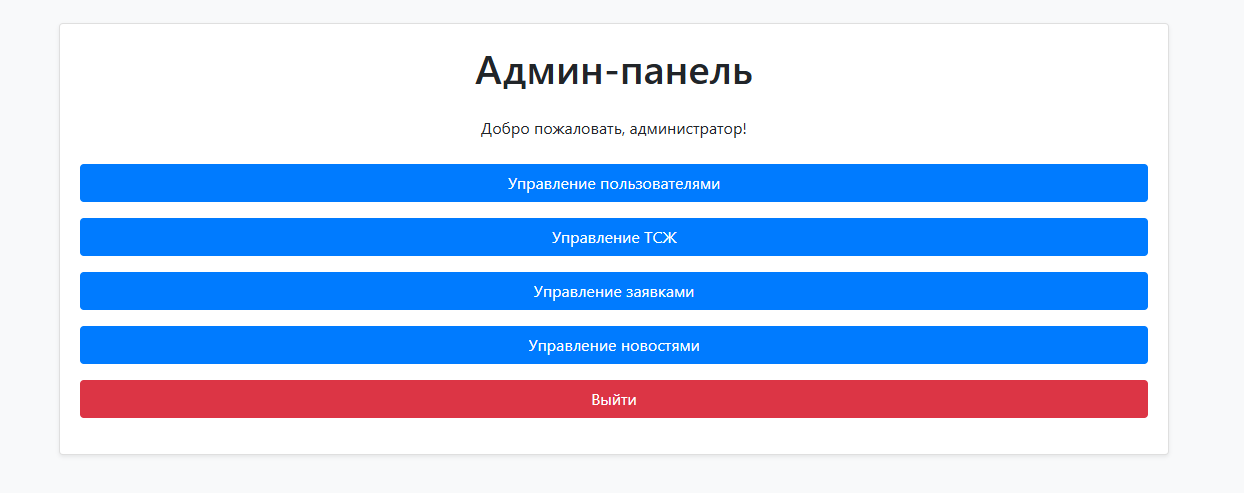


Рис. 11 – внешний вид админ панели

1. **news/ - подсистема для управления новостями платформы**

Попасть в нее можно по нажатию кнопки «Управление новостями»

Данная система отвечает за новости на главной странице. Добавляет возможность создавать, редактировать и удалять их.

* 1. list.php – просмотр всех новостей и кнопки быстрых действий

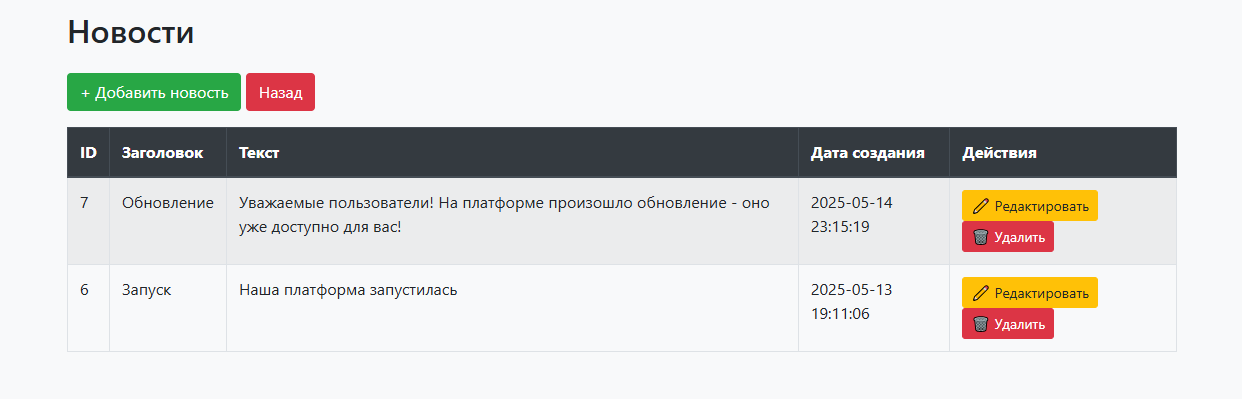


Рис. 12 – внешний вид list.php

* 1. create.php и edit.php – создаие и редактирование новостей. Используется TinyMCE. TinyMCE - платформонезависимый JavaScript HTML WYSIWYG редактор на основе Web. К основным характеристикам программы относятся поддержка тем/шаблонов, языковая поддержка и возможность подключения модулей (плагинов). Используется в различных системах управления содержимым (CMS). [14]

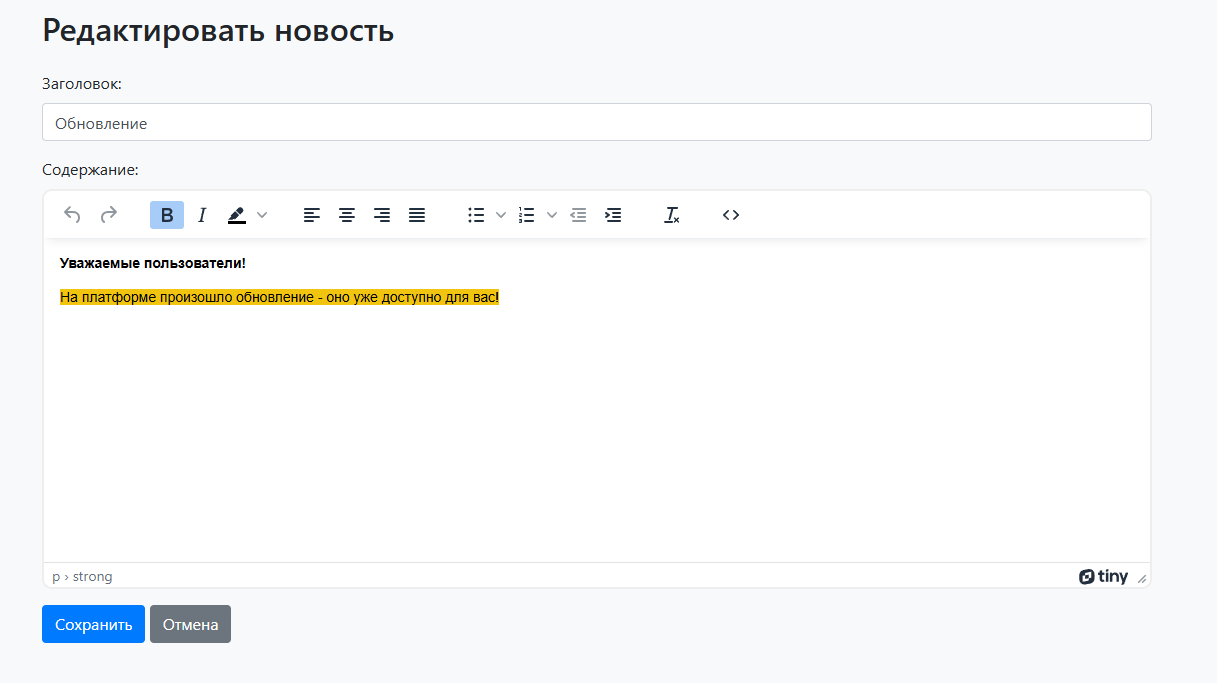


Рис. 13 – внешний вид редактора текста новостей

* 1. delete.php – используется для удаления новостей, работает через кнопку «Удалить» в панели новостей.

1. **Requests/ - подсистема управления всеми заявками внутри платформы**

Попасть в эту панель можно при нажатии «Управление заявками»

Данная система отвечает за все заявки, которые создавали внутри ТСЖ.

* 1. list.php – краткий просмотр всех заявок



Рис.14 – внешний вид списка заявок

* 1. view.php – полный просмотр всей заявки с возможностью изменения статуса

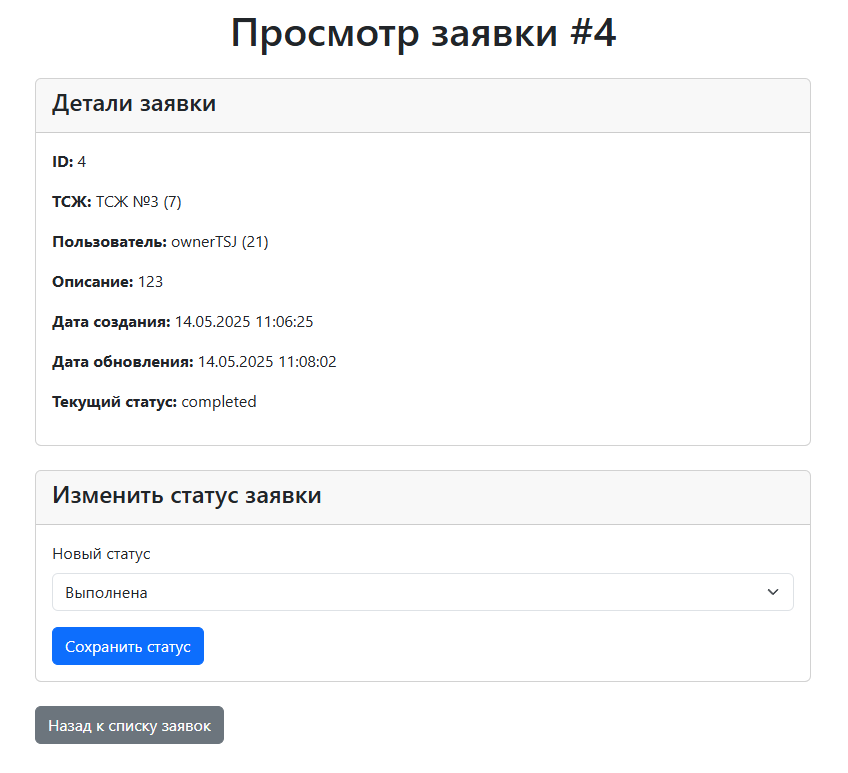


Рис.15 – внешний вид полной заявки в админ панели

1. **tsj/ - подмодуль управления всеми ТСЖ**

Попасть в эту панель можно при нажатии «Управление ТСЖ»

Данная система отвечает за ТСЖ которые созданы внутри платформы. Добавляет возможность изменять данные, а также изменять владельца

* 1. list.php – просмотр списка всех существующих ТСЖ в платформе

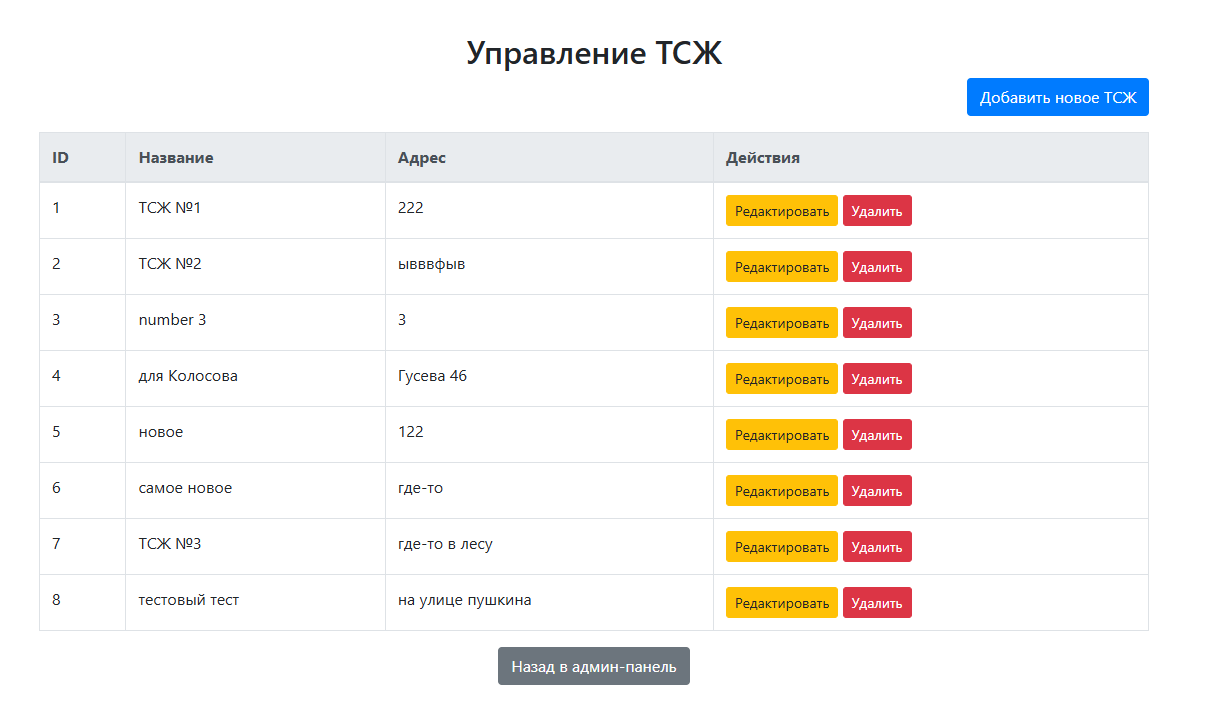


Рис. 16 – Панель управления ТСЖ через админ панель

* 1. edit.php – изменение всех данных о ТСЖ.
  2. add.php – создание ТСЖ через панель администратора с выбором владельца ТСЖ (см. Приложение 2)

1. **users/ - подмодуль управления пользователями**

Попасть сюда можно при нажатии «Управление пользователями»

Данная система отвечает за аккаунты пользователей, через него можно управлять ими, редактировать данные и сбрасывать пароль.

* 1. list.php – просмотр всех пользователей, поиск по фильтрам и кнопки действий

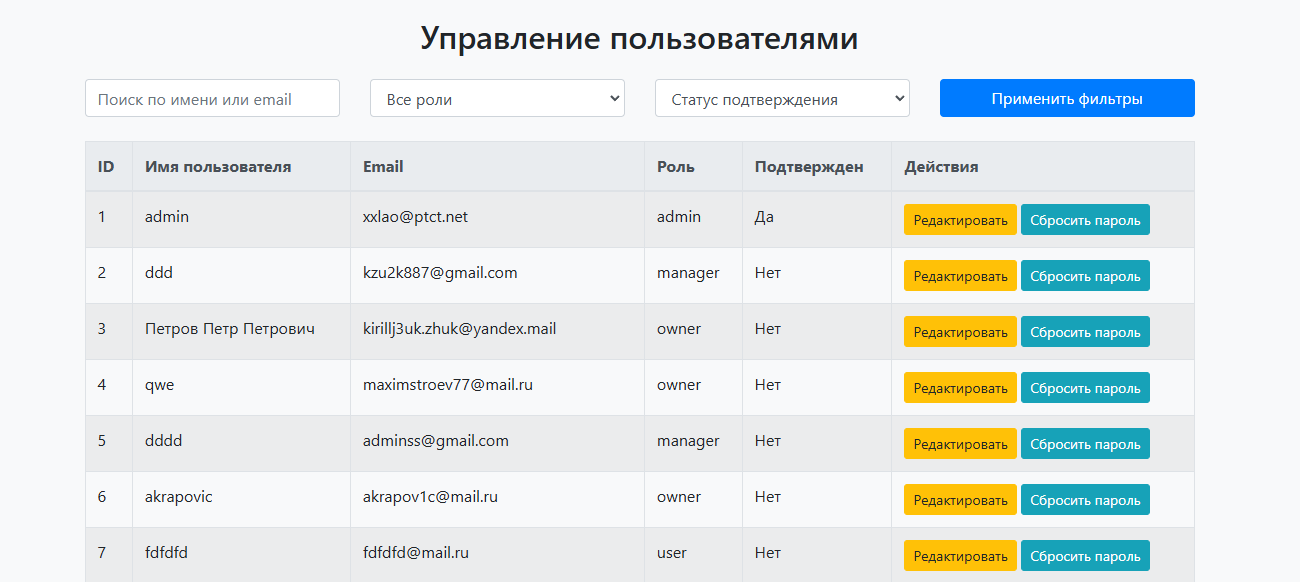


Рис.17 - Управление пользователями в админ панели

* 1. edit.php – редактирование пользователя
  2. reset\_password.php – сбор пароля пользователя. Новый пароль приходит на почту пользователя.

## **2.5 Реализация ключевых функций**

Для обеспечения полноценной работы веб-системы управления ТСЖ была реализована серия ключевых функций, обеспечивающих взаимодействие с пользователями, защиту данных, обработку запросов и управление содержимым. Ниже рассмотрим отдельные участки кода, выполняющие наиболее важные операции.

**Авторизация и контроль доступа**

Каждая страница системы начинается с проверки наличия авторизованного пользователя и соответствующей роли. Это обеспечивает ограничение доступа к функционалу в зависимости от полномочий:

session\_start();

if (!isset($\_SESSION['user\_id']) || $\_SESSION['role'] !== 'admin') {

header("Location: login.php");

exit;

}

Такая проверка выполняется перед доступом к административным или чувствительным разделам сайта (например, управлению пользователями или сбросу паролей).

**Сброс пароля и отправка email**

Администратор имеет возможность сбросить пароль любому пользователю. Новый временный пароль генерируется случайным образом и отправляется на email пользователя с помощью PHPMailer.

Пример:

$new\_password = bin2hex(random\_bytes(4)); // Пример: 'a7f3d9e1'

$hashed\_password = password\_hash($new\_password, PASSWORD\_DEFAULT);

Отправка письма осуществляется через SMTP с помощью PHPMailer:

$mail->isSMTP();

$mail->Host = 'smtp.yandex.ru';

$mail->SMTPAuth = true;

$mail->Username = 'kirilljuk.zhuk@yandex.ru';

$mail->Password = 'секретный\_пароль';

$mail->setFrom('kirilljuk.zhuk@yandex.ru', 'ТСЖ Администрация');

$mail->addAddress($user['email'], $user['username']);

$mail->Subject = 'Ваш пароль был сброшен';

$mail->Body = 'Ваш новый пароль: ' . $new\_password;

$mail->send();

**Добавление жильца с валидацией**

Добавление жильца сопровождается строгой серверной валидацией:

if (empty($full\_name)) {

$errors[] = "ФИО жильца обязательно.";

}

if (!preg\_match('/^\+?[0-9]{7,15}$/', $phone)) {

$errors[] = "Некорректный формат телефона.";}

**Регистрация и авторизация**

Регистрация требует ввода рабочей почты и ее подтверждения для дополнительной защиты аккаунта. При входе так же требуется код с почты для входа в аккаунт. Письма отправляются через PHPMailer, как в примере со сбросом пароля.

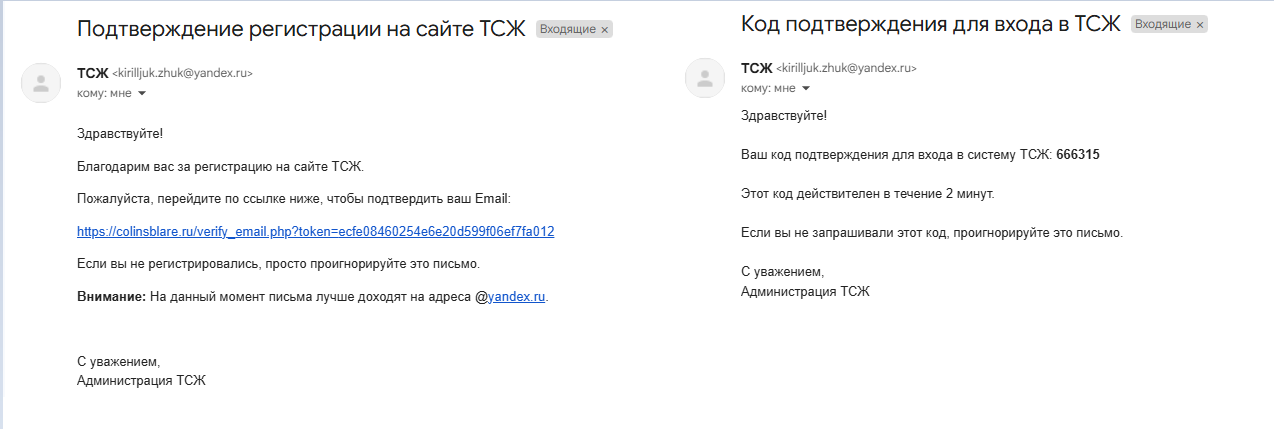


Рис. 18 – Вид письма приходящего на почту

**Фильтрация и вывод записей**

Для повышения удобства пользовательского интерфейса реализована фильтрация данных, например, заявок или платежей:

$filter\_status = filter\_input(INPUT\_GET, 'status', FILTER\_SANITIZE\_FULL\_SPECIAL\_CHARS);

$where\_clause = "WHERE tsj\_id = :tsj\_id";

if (!empty($filter\_status)) {

$where\_clause .= " AND status = :status";

$params['status'] = $filter\_status;

}

Такой подход позволяет быстро сортировать данные по статусу, плательщику и другим параметрам без перезагрузки интерфейса.

**Использование Bootstrap**

Для унификации стиля и обеспечения адаптивности интерфейса в проект был интегрирован фреймворк Bootstrap. Например, формы и таблицы управления оформлены при помощи стандартных компонентов

# Заключение

В процессе выполнения дипломной работы была реализована информационная платформа для автоматизации деятельности товарищества собственников жилья (ТСЖ), целью которой стало упрощение процессов управления жилым фондом, повышения прозрачности ведения учета, а также улучшение взаимодействия между жильцами и управляющей организацией.

На начальном этапе был проведён глубокий анализ предметной области, в ходе которого были выявлены основные задачи и проблемы, характерные для ТСЖ. Среди них — отсутствие единой системы для ведения учета жильцов, квартир, заявок и платежей, а также высокая зависимость от человеческого фактора, что нередко приводит к ошибкам и затрудняет оперативное принятие решений. Проведён сравнительный анализ существующих решений на рынке, таких как 1С:Учет в ЖКХ, Домовой, ЖКХ Контроль. На основании выявленных недостатков этих систем были определены требования к разрабатываемой платформе.

На втором этапе была спроектирована структура базы данных с учётом логических связей между сущностями, таких как жильцы, квартиры, платежи, заявки и само ТСЖ. Выбран тип реляционной базы данных и соответствующие технологии — MySQL в связке с PHP и phpMyAdmin. Также был произведён выбор технологий для создания пользовательского интерфейса: HTML, CSS, Bootstrap и JavaScript. Такой стек технологий обеспечил не только визуальную привлекательность, но и адаптивность, безопасность и простоту сопровождения проекта.

Особое внимание в работе уделено вопросам безопасности. В ходе проектирования были реализованы механизмы защиты от основных видов уязвимостей, включая SQL-инъекции, XSS и CSRF-атаки. Организована система аутентификации и авторизации пользователей с различными ролями и уровнями доступа. Также предусмотрена возможность сброса пароля и подтверждения регистрации через email с использованием библиотеки PHPMailer.

Реализованный функционал охватывает все ключевые процессы деятельности ТСЖ: от регистрации пользователей и добавления жильцов до обработки заявок, учёта платежей и формирования отчетности. Для удобства пользователей предусмотрены фильтрация данных, экспорт в PDF и единый интерфейс управления. Система поддерживает разделение прав между администраторами, владельцами ТСЖ, менеджерами и жильцами, что делает её универсальным решением для организаций различного масштаба.

Платформа показала свою работоспособность и эффективность на тестовых данных. Все поставленные в начале работы задачи были успешно реализованы:

* проведён анализ потребностей ТСЖ и существующих решений;
* спроектирована и реализована структура базы данных;
* разработан полноценный пользовательский и административный интерфейс;
* внедрены механизмы безопасности и защиты данных;
* проведено тестирование системы и устранены выявленные ошибки.

Таким образом, цель дипломной работы достигнута. Созданная платформа может быть внедрена в реальной среде и использоваться как готовое решение для автоматизации учета и управления в ТСЖ. В перспективе возможно расширение функционала, интеграция с платёжными системами, поддержка мобильных устройств и модуль аналитики, что позволит еще больше повысить её ценность и применимость.

Разработанный программный продукт имеет практическую значимость, так как способен не только сократить трудозатраты персонала, но и повысить уровень доверия со стороны жильцов, обеспечив прозрачность всех процессов в рамках ТСЖ.

# Список источников

1. "1С:Предприятие 8. Учет в управляющих компаниях ЖКХ, ТСЖ и ЖСК". [Электронный ресурс] / – Режим доступа: <https://1c.platforma-it.ru/catalog/otrasl/gkh/1c-uchet-v-upravlyayushchikh-kompaniyakh-zhkkh-tszh-i-zhsk/>

2. Bootstrap. [Электронный ресурс] / – Режим доступа: <https://gitverse.ru/blog/articles/architecture/623-bootstrap-chto-eto-za-frejmvork-i-kak-im-polzovatsya>

3. CSRF-атака. [Электронный ресурс] / – Режим доступа: <https://startx.team/blog/statyi/csrf-ataka-metody-zashhity-i-primery-uyazvimostej/>]

4. MongoDB: обзор у правляемой базы данных в облаке. [Электронный ресурс] / – Режим доступа:

<https://yandex.cloud/ru/blog/posts/2020/12/managed-mongodb-overview>

5. phpMyAdmin — веб-приложение с открытым кодом. [Электронный ресурс] / – Режим доступа: <https://ru.wikipedia.org/wiki/PhpMyAdmin>

6. SQL-инъекция [Электронный ресурс] / – Режим доступа: <https://habr.com/ru/articles/725134/>

7. XSS-атака. [Электронный ресурс] / – Режим доступа: <https://habr.com/ru/companies/alfa/articles/717896/>

8. Введение в объектно-ориентированные базы данных. [Электронный ресурс] / – Режим доступа: <https://habr.com/ru/articles/56399/>

9. Виды баз данных. [Электронный ресурс] / – Режим доступа:

<https://selectel.ru/blog/databases-types/>

10. ГИС.ЖКХ [Электронный ресурс] / – Режим доступа: <https://dom.gosuslugi.ru/?ysclid=mbascj34oe129935236#!/main>

11. Джуба С., Волков А. Изучаем PostgreSQL 10 / пер. с анг. А. А. Слинкина. – М.: ДМК Пресс, 2019. – 400 с

12. Домовой. [Электронный ресурс] / – Режим доступа:

<https://infinnity.solutions/project/upravdom>

13. Дронов В.А. РНР и MySQL. 25 уроков для начинающих. - СПб.: БХВ-Петербурr, 2021. -432 с.: ил. - (Для начинающих)

14. Редактор HTML WYSIWYG. [Электронный ресурс] / – Режим доступа: <https://ru.wikipedia.org/wiki/TinyMCE>

15. Сергеева Т.И. Базы данных: модели данных, проектирование, язык SQL: учеб. пособие / Т.И. Сергеева, М.Ю. Сергеев. Воронеж: ФГБОУ ВПО «Воронежский государственный технический университет», 2012. 233 с.

16. ТСЖ. Жилищный кодекс Российской Федерации (по состоянию на 25 января 2014 года) – Москва: Проспект, 2014 – 128 с.

17. Что такое ER-модель и ER-диаграмма? [Электронный ресурс] / – Режим доступа:

<https://it.vshp.online/#/pages/op08/op08_10_lec?id=Что-такое-er-модель-и-er-диаграмма>

18. Что такое HTTPS? [Электронный ресурс] / – Режим доступа: <https://hi-tech.mail.ru/review/125752-chto-takoe-https/>

19. Что такое NoSQL? Тесленко И. Б. BIG DATA БОЛЬШИЕ ДАННЫЕ: Учебное пособие/ И. Б. Тесленко, А. М. Губернаторов, О. Б. Дигилина, В. Е. Крылов Владим. гос. ун-т им. А. Г. и Н. Г. Столетовых. – Владимир : Изд-во ВлГУ, 2021 – 123 с.

20. Что такое SQLite. [Электронный ресурс] / – Режим доступа:

<https://workspace.ru/tools/database/sqllite/>

21. Язык PHP. [Электронный ресурс] / – Режим доступа: <https://practicum.yandex.ru/blog/yazyk-programmirovaniya-php/>

# Приложение 1

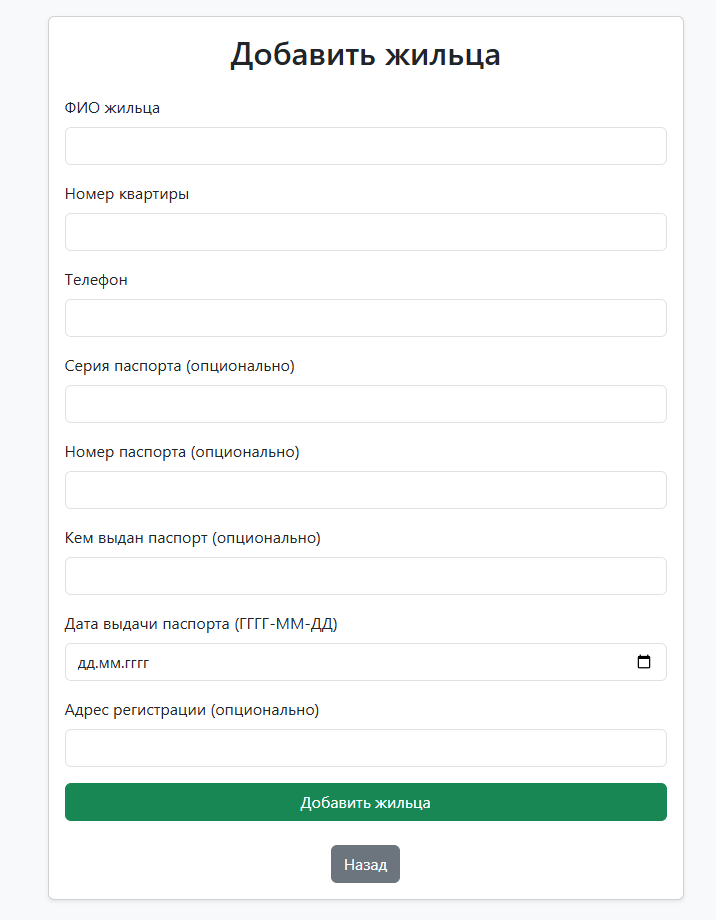


Рис. – внешний вид dashboard/add\_resident.php

# Приложение 2

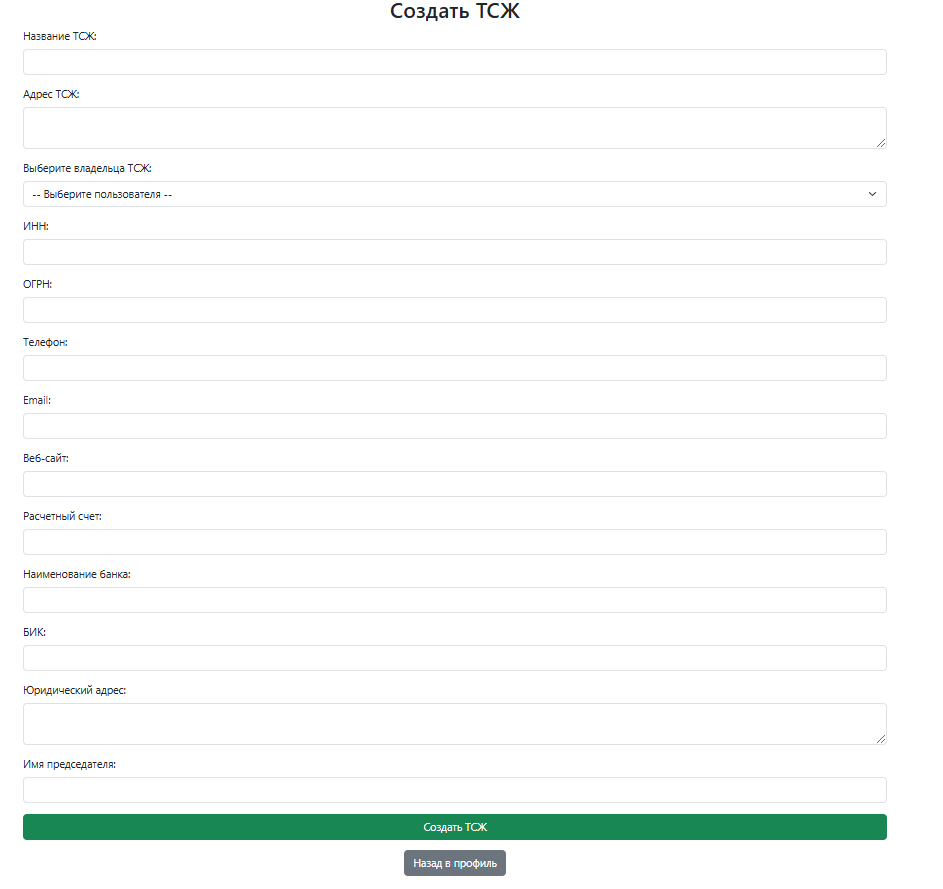


Рис. – Создание ТСЖ через панель администратора

# Приложение 1



Ссылка для ручного ввода: github.com/ColinsBlares/diplom