Министерство науки и высшего образования Российской Федерации

ФГБОУ ВО «БАЙКАЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

ЧИТИНСКИЙ ИНСТИТУТ

Кафедра информационных технологий и высшей математики

**Производственная практика (преддипломная)**

ОТЧЕТ ПО ПРАКТИЧЕСКОЙ ПОДГОТОВКЕ

обучающегося бакалавриата группы ИСТУ-21 Елиной И.А.

Руководитель(-и) по практической подготовке

от Института: канд. техн. наук, доцент, декан ФЭФ Михайлова Е.А.

Чита, 2025

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc199237513)

[1. ВЫБОР ИНСТРУМЕНТАЛЬНЫХ СРЕДСТВ 5](#_Toc199237514)

[1.1. Инструменты создания веб-интерфейса 5](#_Toc199237515)

[1.2. Язык программирования 7](#_Toc199237516)

[1.3. Работа с базами данных 9](#_Toc199237517)

[2. РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ 12](#_Toc199237518)

[2.1. Разработка веб-интерфейса 12](#_Toc199237519)

[2.2. Верстка страниц проекта 13](#_Toc199237520)

[2.3. Подключение к базе данных 19](#_Toc199237521)

[2.4. Авторизация пользователя 20](#_Toc199237522)

[2.5. Безопасность страниц приложений 21](#_Toc199237523)

[2.6. Добавление документов 22](#_Toc199237524)

[2.7. Управление правами доступа пользователей в проекте 23](#_Toc199237525)

[3. ОЦЕНКА ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ 25](#_Toc199237526)

[ЗАКЛЮЧЕНИЕ 28](#_Toc199237527)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 29](#_Toc199237528)

# ВВЕДЕНИЕ

В современных условиях развития информационных технологий и цифровизации бизнес-процессов эффективность работы крупных организаций во многом зависит от качества и скорости обработки информации. Железнодорожная отрасль, представленная такими крупными структурами, как ОАО «Российские железные дороги» (далее — РЖД), требует внедрения современных решений для автоматизации документооборота и повышения производительности труда.

В рамках прохождения производственной практики в Информационно-вычислительном центре РЖД была поставлена задача разработки программного обеспечения, обеспечивающего удобный и быстрый поиск документов с возможностью фильтрации по различным параметрам. Целью проекта является создание эффективного инструмента, позволяющего сотрудникам оперативно находить необходимые документы, что способствует снижению временных затрат и повышению качества управленческих решений.

Особенностью разработанной системы является интеграция функции оптического распознавания текста (OCR), которая позволяет преобразовывать содержимое загруженных PDF-документов в текстовый формат. Это значительно расширяет возможности поиска, позволяя осуществлять полнотекстовый поиск по содержимому документов, включая отсканированные материалы.

Система обеспечивает поиск по названию и распознанному тексту документов, а также фильтрацию по категориям и дате добавления. Особое внимание уделялось удобству использования и регистронезависимому поиску, что способствует повышению эффективности работы пользователей.

Реализация данного проекта способствует оптимизации рабочих процессов в ИВЦ РЖД, снижению временных затрат на поиск информации и повышению качества управленческих решений, что в конечном итоге повышает экономическую эффективность деятельности филиала.

Целями практики являются:

* + изучение опыта создания и применения информационных технологий для решения реальных задач в условиях конкретной организации;
  + сбор необходимого материала для выполнения работы.

Задачи практики включают:

* создание и использование информационных технологий в конкретной организации;
* получение практического опыта применения технологий разработки программного обеспечения;
* выбор технологии для реализации программного обеспечения;
* разработка программно-информационного обеспечения в условиях конкретной организации.

# ВЫБОР ИНСТРУМЕНТАЛЬНЫХ СРЕДСТВ

## Инструменты создания веб-интерфейса

Пользовательский интерфейс является важной частью разрабатываемого проекта. С помощью него осуществляется взаимодействие пользователя с технической системой, поэтому он должен быть простым и интуитивно понятным. При разработке веб-интерфейса будут использоваться такие технологии как HTML, CSS и JavaScript, которые обеспечивают структуру, стилизацию и интерактивность страниц. Помимо этого, применялись современные фреймворки, значительно упрощающие и ускоряющие процесс разработки, позволяя создавать более удобный, динамичный и отзывчивый интерфейс.

HTML (от англ. HyperText Markup Language – «язык гипертекстовой разметки») – это стандартизированный язык разметки документов для просмотра веб-страниц в браузере.  Он задаёт структуру и содержание документа, используя теги для обозначения заголовков, абзацев, списков, изображений, ссылок и других элементов. HTML формирует каркас сайта, который затем стилизуется и «оживляется» с помощью других технологий.

CSS (от англ. Cascading Style Sheets – «каскадные таблицы стилей») – это формальный язык описания внешнего вида веб-страницы. Он отвечает за оформление элементов HTML: цвета, шрифты, размеры, отступы, расположение, анимации и адаптивность под разные устройства. CSS позволяет отделить структуру (HTML) от визуального представления, что упрощает поддержку и развитие сайта.

JavaScript – это язык программирования, который добавляет интерактивность и динамическое поведение веб-страницам. С помощью JavaScript можно реагировать на действия пользователя (клики, ввод текста), изменять содержимое страницы без её перезагрузки, создавать анимации, работать с данными и взаимодействовать с сервером. Это ключевой инструмент для создания современных веб-приложений.

Объектная модель документа (DOM, Document Object Model) – это программный интерфейс, который представляет HTML- или XML-документ в виде иерархического дерева объектов (узлов). Каждый элемент страницы, такой как тег, текст или атрибут, становится отдельным объектом в этом дереве. DOM позволяет браузеру структурировать содержимое веб-страницы и предоставляет программам (чаще всего JavaScript) возможность получать доступ к этим объектам, изменять их, добавлять новые или удалять существующие.

Благодаря DOM веб-страницы перестают быть статичными - JavaScript может динамически изменять структуру, содержимое и стили страницы уже после её загрузки, создавая интерактивный и отзывчивый интерфейс. Например, с помощью DOM можно изменять текст, скрывать или показывать элементы, реагировать на действия пользователя (клики, ввод данных и т.д.).

Таким образом, DOM – это своего рода «каркас» или «модель» веб-страницы в памяти браузера, который связывает HTML-разметку с возможностями программного управления через скрипты. Это ключевой компонент для создания современных динамичных веб-приложений.

Bootstrap – это популярный CSS-фреймворк с набором готовых стилей и компонентов (кнопки, формы, навигация, сетки и др.). Он значительно ускоряет разработку адаптивных и кроссбраузерных интерфейсов, обеспечивая единый стиль и удобство использования на разных устройствах - от мобильных телефонов до десктопов. Bootstrap также включает JavaScript-плагины для расширения функциональности.

Основными преимуществами данного фреймворка являются:

* большая библиотека компонентов – предлагает готовые и протестированные элементы интерфейса, что значительно облегчает работу разработчикам и позволяет разрабатывать проекты быстрее;
* адаптивность – встроенная 12-колоночная сетка и готовые утилиты обеспечивают корректное отображение сайта на устройствах с разными размерами экранов - от смартфонов до десктопов;
* кроссбраузерность и кроссплатформенность – веб-приложения будут одинаково отображаться во всех современных браузерах;
* низкий порог входа – основы Bootstrap базируются на привычных HTML, CSS и JavaScript, что позволяет быстро освоить фреймворк даже начинающим разработчикам.
* широкое сообщество и документация – наличие большого количества обучающих материалов, форумов и примеров облегчает изучение и решение возникающих вопросов.
* бесплатность и открытый исходный код – Bootstrap распространяется бесплатно и может использоваться в любых проектах, включая коммерческие.

## Язык программирования

Помимо пользовательского интерфейса, важнейшей частью разрабатываемой системы является серверная составляющая, которая служит основой всей платформы. Именно на сервере реализуется ключевой функционал программного продукта, и выбор языка программирования здесь оказывает значительное влияние на качество и эффективность разработки.

В качестве основного языка для создания серверной части онлайн-платформы был выбран Python [1].

Python – это высокоуровневый, интерпретируемый язык программирования, который завоевал широкую популярность благодаря своей простоте и гибкости. Он широко применяется в различных сферах, включая веб-разработку, научные исследования, анализ данных, искусственный интеллект и автоматизацию.

Одним из главных преимуществ Python является его понятный и лаконичный синтаксис, который облегчает чтение и поддержку кода, что особенно важно при командной разработке и масштабировании проектов. Кроме того, Python обладает обширной стандартной библиотекой и поддерживается большим сообществом разработчиков, что обеспечивает доступ к множеству сторонних библиотек и фреймворков.

Для веб-разработки на Python существует ряд мощных инструментов, среди которых особенно выделяются фреймворки Flask и Django. Эти инструменты позволяют быстро создавать надёжные и масштабируемые серверные приложения, обеспечивая гибкость и удобство разработки. В рамках данного проекта в качестве основного средства разработки серверной части был выбран Flask - лёгкий и минималистичный фреймворк, который идеально подходит для создания эффективных и легко поддерживаемых веб-приложений [2].

Веб-фреймворк Flask поддерживает следующие ключевые возможности:

* модульное тестирование, позволяющее проверять корректность отдельных частей исходного кода;
* интеграцию с Google App Engine - мощной платформой для развертывания приложений в инфраструктуре Google;
* реализацию RESTful-архитектуры, обеспечивающей создание гибких и масштабируемых веб-сервисов.

Flask обеспечивает защиту от XSS-атак, при которых злоумышленник пытается внедрить вредоносные скрипты на страницу, а также работает с безопасными cookie-файлами. В состав фреймворка входит инструмент Werkzeug, предоставляющий простой встроенный веб-сервер для разработки и тестирования.

Среди преимуществ Flask выделяются:

* простой и понятный синтаксис, схожий с языком Python;
* удобная система шаблонов, позволяющая быстро создавать прототипы веб-приложений;
* широкий набор инструментов для гибкой настройки и расширения функционала сайтов;
* высокая расширяемость благодаря множеству готовых расширений и плагинов, которые упрощают добавление таких функций, как авторизация, работа с базами данных и обработка форм.

Благодаря таким возможностям Python и его экосистемы, он является оптимальным выбором для разработки серверной части современной онлайн-платформы.

## Работа с базами данных

Разрабатываемая онлайн-платформа предназначена для эффективного управления документооборотом и обработки большого объёма данных. В системе хранятся различные документы, которые необходимо не только сохранять, но и обеспечивать быстрый поиск и анализ их содержимого. Для этого в проекте реализована технология оптического распознавания текста (OCR), позволяющая преобразовывать отсканированные и PDF-документы в текстовый формат для полнотекстового поиска.

Поскольку документы и связанные с ними метаданные тесно взаимосвязаны, возникает задача точного и надёжного управления всей информацией, чтобы исключить ошибки и обеспечить целостность данных. Для этого используется удобный и надёжный инструмент управления базой данных, который обеспечивает структурированное хранение документов, их классификацию и быстрый доступ.

СУБД должна обеспечивать высокую производительность, безопасность и удобство администрирования, а также поддерживать масштабируемость по мере роста объёмов информации. В качестве основного инструмента для работы с базой данных в данном проекте используется MySQL - одна из самых популярных и проверенных временем реляционных СУБД [3].

MySQL – это популярная реляционная система управления базами данных (СУБД) с открытым исходным кодом, основанная на языке SQL. Она широко применяется для создания, хранения и обработки данных в самых разных сферах - от небольших веб-проектов до крупных корпоративных систем.

СУБД построена на клиент-серверной архитектуре. Сервер отвечает за хранение и управление базами данных, а клиент - за отправку запросов на сервер с помощью языка SQL. Основная задача MySQL - эффективно создавать таблицы, устанавливать связи между ними и обеспечивать быстрый доступ к данным.

Преимущества MySQL:

* бесплатность и открытый исходный код - доступна для скачивания и использования без ограничений;
* кроссплатформенность - работает на Linux, Windows, macOS и других ОС;
* высокая производительность и масштабируемость - способна эффективно обрабатывать большие объёмы данных;
* безопасность - поддерживает шифрование данных и разграничение доступа;
* поддержка стандартного SQL и расширений - позволяет выполнять сложные запросы и оптимизировать работу с данными;
* широкая экосистема инструментов - например, MySQL Workbench для удобного администрирования и визуализации баз данных, а также phpMyAdmin для управления через браузер.

Для обеспечения безопасности данных и предотвращения атак типа SQL-инъекций в проекте используются дополнительные программные библиотеки. Одной из таких является SQLAlchemy - мощный инструмент для работы с реляционными базами данных на языке Python. Эта библиотека поддерживает основные операции с базами данных, включая создание, удаление и изменение структур.

SQLAlchemy реализует технологию ORM (Object-Relational Mapping), которая позволяет синхронизировать объекты Python с записями в реляционной базе данных. Благодаря этому разработчик может описывать структуру базы данных и взаимодействовать с ней, используя привычный синтаксис Python, без необходимости писать сложные SQL-запросы вручную. Кроме того, SQLAlchemy автоматически экранирует параметры запросов, что значительно снижает риск SQL-инъекций и повышает безопасность приложения.

Преимущества использования SQLAlchemy:

1. Безопасность. Параметры запросов экранируются, что делает атаки типа внедрение SQL-кода маловероятными.
2. Производительность. Повышается вероятность повторного использования запроса к серверу базы данных, что может позволить ему в некоторых случаях применить повторно план выполнения запроса.
3. Переносимость. SQLAlchemy, при должном подходе, позволяет писать код на Python, совместимый с несколькими backend СУБД. Несмотря на стандартизацию языка SQL, между базами данных имеются различия в его реализации, абстрагироваться от которых и помогает SQLAlchemy.

# РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ

## Разработка веб-интерфейса

Ключевым элементом любого веб-приложения является пользовательский интерфейс - средство, через которое пользователь взаимодействует с системой. Клиентская часть включает не только привлекательный дизайн (цветовые решения, шрифты, анимации), но и удобный, понятный функционал.

При создании программного продукта важно уделять равное внимание обеим этим аспектам. Для этого необходимо понимать основные концепции «пользовательского опыта» (User Experience, UX) и «пользовательского интерфейса» (User Interface, UI).

UX отвечает за логику взаимодействия пользователя с элементами интерфейса. В его задачи входит организация архитектуры веб-приложения, определение иерархии и расположения элементов и блоков на страницах. Хороший UX-дизайн должен быть простым и интуитивно понятным, а размещение кнопок, изображений, меню и других элементов - логичным и соответствующим контексту. Кроме того, важна адаптивность интерфейса для корректного отображения на различных устройствах.

UI, в свою очередь, описывает визуальную сторону проекта. Здесь решающую роль играют цветовая палитра, общий стиль оформления, выбор шрифтов, иконок и другие визуальные компоненты. Внешний вид должен создавать положительное впечатление, не перегружать зрение, а шрифты - быть легко читаемыми и контрастными по отношению к фону. Важным является правильный подбор цветовых сочетаний, использование визуальных эффектов и анимаций.

При разработке онлайн-платформы для проведения Конкурса особое внимание было уделено следующим аспектам UX/UI-дизайна:

1. Простота. Интерфейс разработан таким образом, чтобы быть максимально понятным и удобным для пользователя, а последовательность действий - логичной и предсказуемой. Внешний вид и функциональность платформы не перегружены лишними элементами, отсутствует избыточная визуальная информация. Основные разделы собраны в меню, при переходе по которым открываются вкладки с необходимой информацией.
2. Контрастность. Отдельные элементы и блоки чётко выделены и не сливаются друг с другом. Цветовая палитра преимущественно светлая, а текст и кнопки имеют яркий контрастный оттенок, что помогает привлечь внимание и облегчает восприятие.
3. Расположение элементов. Все компоненты интерфейса - меню, формы, поля для ввода - выровнены по единой сетке, имеют равномерные отступы и не перекрывают друг друга. Каждый элемент занимает своё место в иерархии важности, при этом ключевые блоки расположены в наиболее заметных зонах страницы. Для этого использовался принцип «Z-образного» паттерна, согласно которому пользователь просматривает страницу сверху вниз и слева направо.
4. Логическая структура и соответствие контексту. Содержимое страниц полностью соответствует названиям разделов и ожиданиям пользователей. Например, при переходе на страницу авторизации пользователь сразу видит привычные поля для ввода «Логин» и «Пароль», а также кнопку «Войти», что облегчает навигацию и взаимодействие.

## Верстка страниц проекта

В ходе выполнения проекта был проведён этап согласования дизайна с организацией-заказчиком с целью обеспечения соответствия корпоративным стандартам и требованиям пользователя.

Для разработки шаблонов страниц и прототипов использовался программный продукт Adobe Acrobat [4]. Этот инструмент позволяет создавать интерактивные PDF-документы с визуально проработанными макетами, что обеспечивает наглядное представление структуры страниц, оформления элементов интерфейса и навигации.

Использование Adobe Acrobat позволило:

* обеспечить высокое качество визуализации макетов;
* упростить процесс внесения правок и корректировок;
* организовать удобную демонстрацию и обсуждение дизайна с заказчиком.

Шаблоны, созданные в Adobe Acrobat, послужили основой для дальнейшей разработки проекта. После утверждения макетов начался этап программной реализации, в ходе которого были разработаны функциональные страницы системы, соответствующие согласованным шаблонам.

Разработанное приложение представляет собой внутреннюю корпоративную систему, обеспечивающую удобный и безопасный доступ к функционалу и данным организации. После авторизации пользователи получают возможность работать с различными разделами системы, каждый из которых отвечает за определённые задачи.

Так, для просмотра и взаимодействия пользователям будут доступны следующие страницы:

* Главная страница – отображает полный список всех документов, доступных в системе;
* Профиль пользователя – содержит документы, загруженные и созданные конкретным пользователем;
* Добавление документа – предоставляет интерфейс для загрузки и создания новых документов в системе;
* О программе – содержит информацию о приложении, его назначении и основных функциях.

Страница входа является первым шагом для доступа к системе. Она обеспечивает безопасную авторизацию пользователей, позволяя подтвердить их личность и предоставить доступ к функционалу приложения в соответствии с правами и ролями. На странице предусмотрены поля для ввода логина и пароля, а также кнопка для подтверждения данных и входа в систему. Такой подход гарантирует защиту информации и персонализированный доступ для каждого пользователя (Рисунок 1).

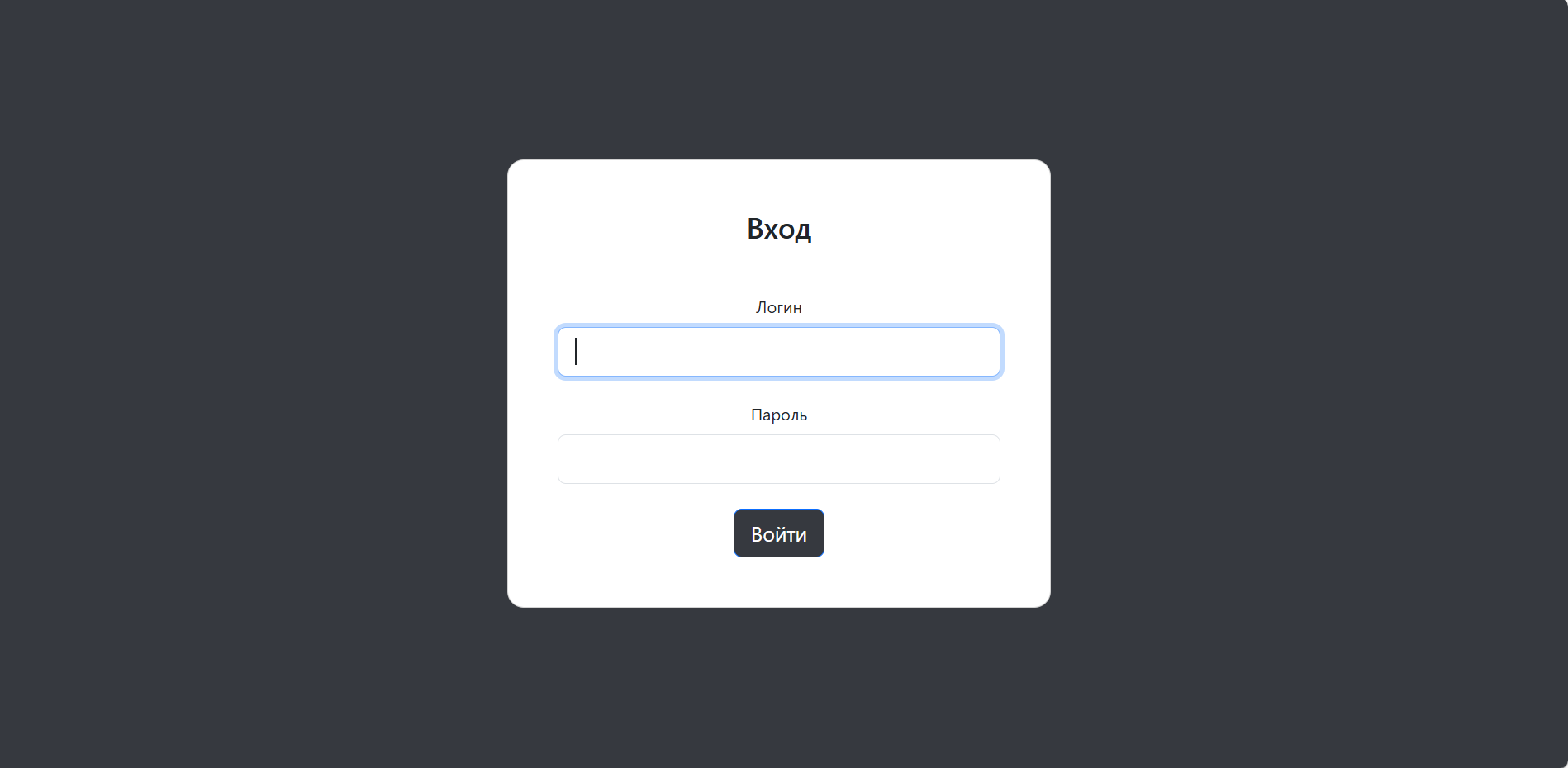


Рисунок 1 – Страница авторизации

Главная страница служит центральным элементом интерфейса и предназначена для отображения полного списка всех документов, доступных в системе. Здесь пользователи могут быстро просмотреть, найти и выбрать необходимые материалы для работы. При этом функционал страницы варьируется в зависимости от уровня доступа пользователя - разные роли получают соответствующие им возможности и инструменты (Рисунки 2.1 и 2.2).

На главной странице реализованы функции сортировки и фильтрации документов, что позволяет эффективно управлять большим объёмом данных. Интерфейс выполнен с учётом принципов простоты и логичности, чтобы обеспечить комфортное взаимодействие пользователя с системой.

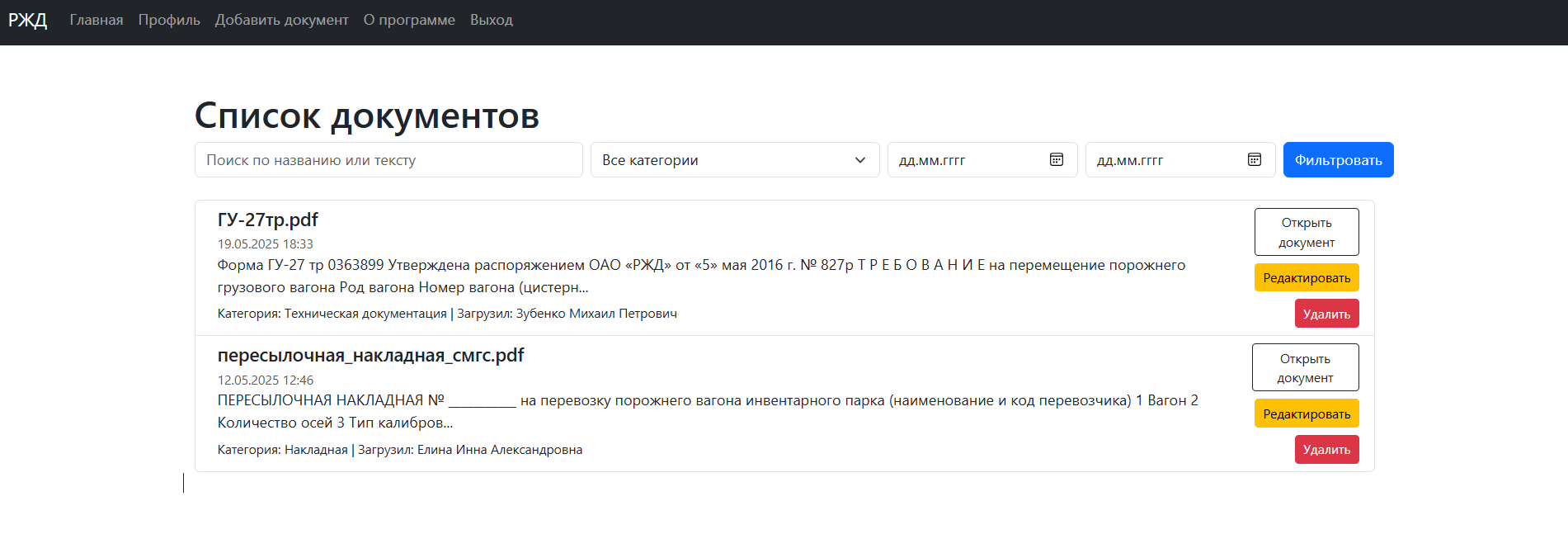


Рисунок 2.1 – Главная страница. Администратор

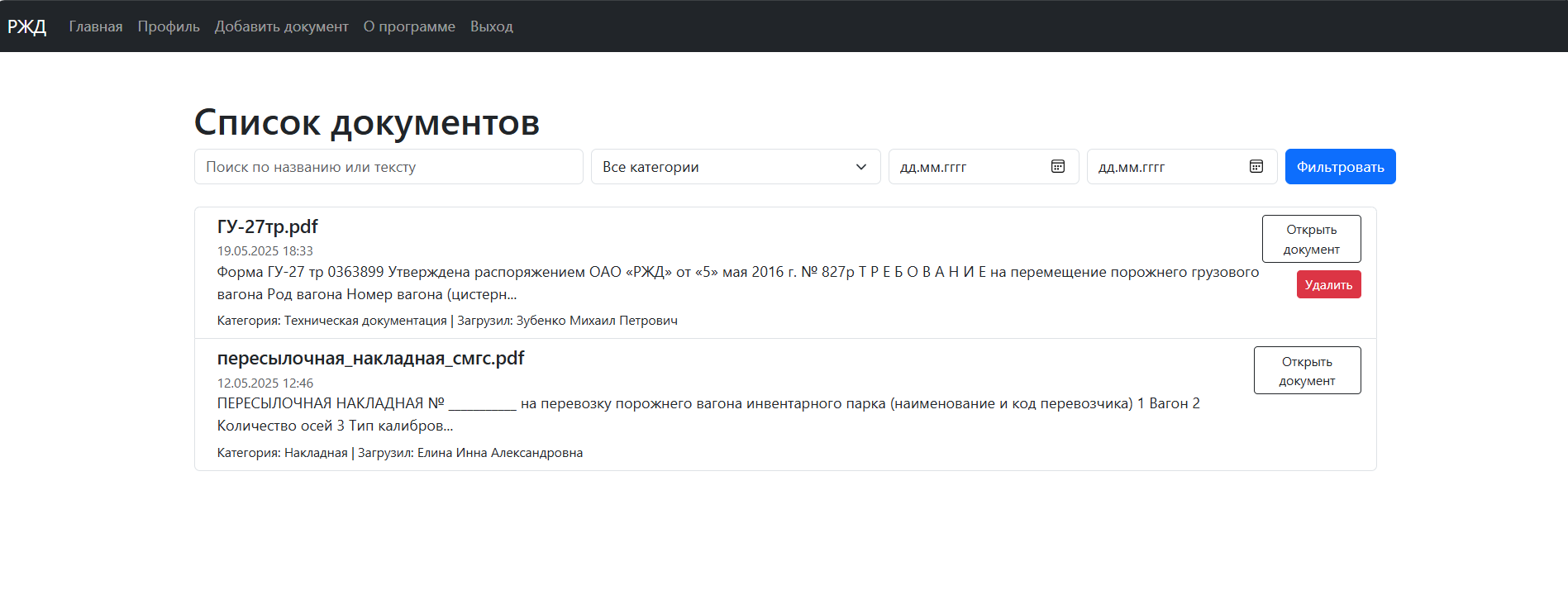


Рисунок 2.2 – Главная страница. Пользователь

Страница профиля предназначена для отображения документов, которые он загрузил или создал в системе. Здесь пользователь может просмотреть список своих материалов, отсортировать их по дате или другим параметрам, а также получить быстрый доступ к редактированию и управлению этими документами.

Интерфейс страницы профиля разработан с учётом удобства и прозрачности: все элементы расположены логично, что облегчает навигацию и работу с файлами. Кроме того, на странице реализованы функции фильтрации и поиска, позволяющие быстро найти нужный документ среди большого количества.

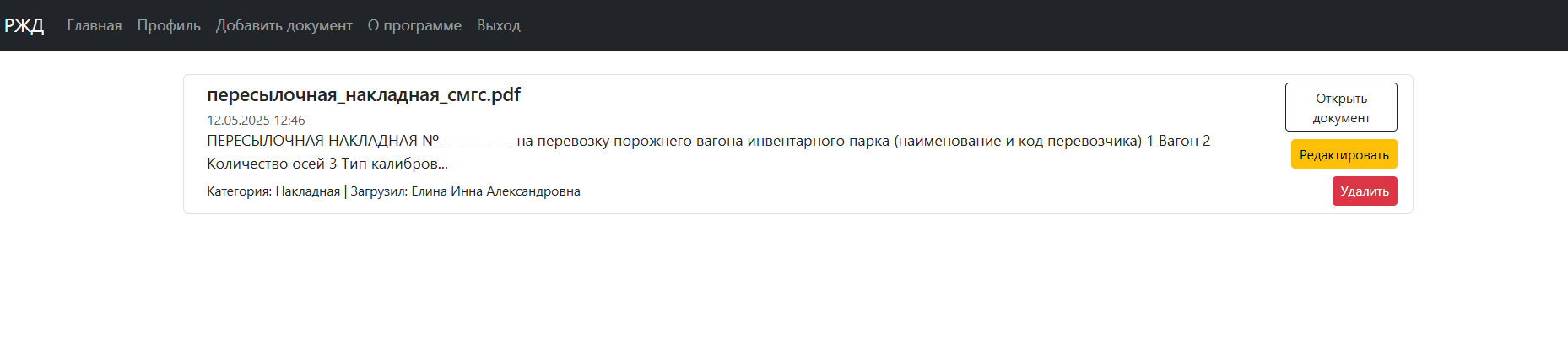


Рисунок 3.1 – Страница профиля. Администратор

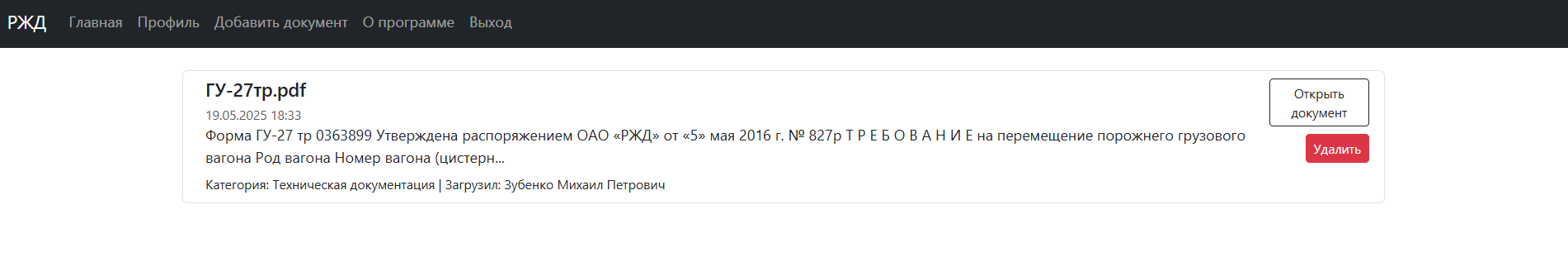


Рисунок 3.2 – Страница профиля. Пользователь

В рамках разработанной корпоративной системы страница добавления документа предоставляет пользователю возможность загружать новые файлы в систему с минимальными усилиями. Интерфейс включает поле выбора файла с устройства. (Рисунок 4)

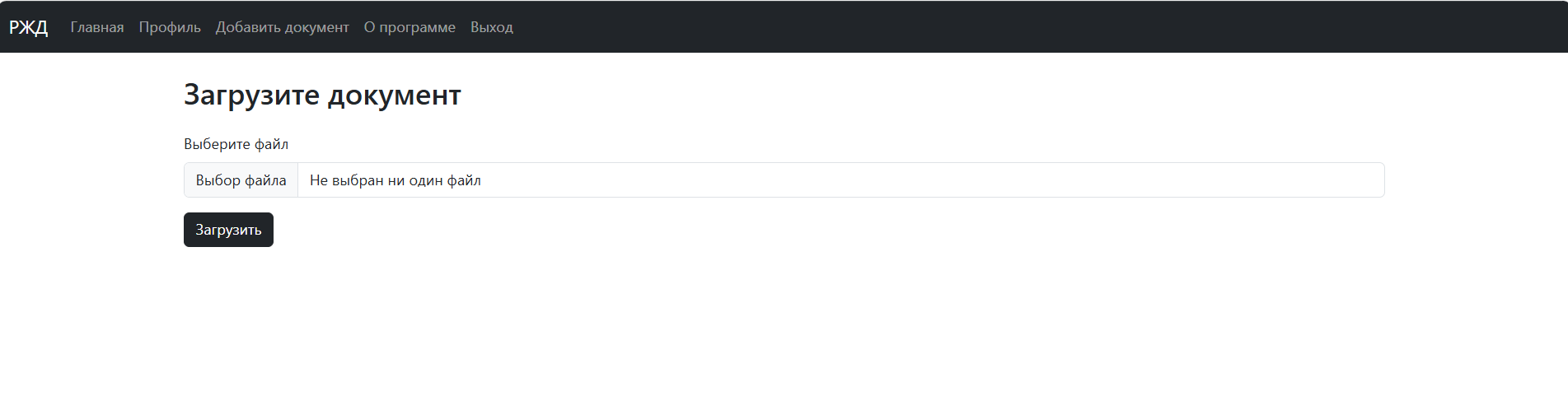


Рисунок 4 – Страница добавления документа

По завершении распознавания пользователь перенаправляется на страницу редактирования, где может внести необходимые корректировки в текст документа. Загруженный и отредактированный документ становится доступен для просмотра и дальнейшей работы как на главной странице, так и в профиле пользователя (Рисунок 5).

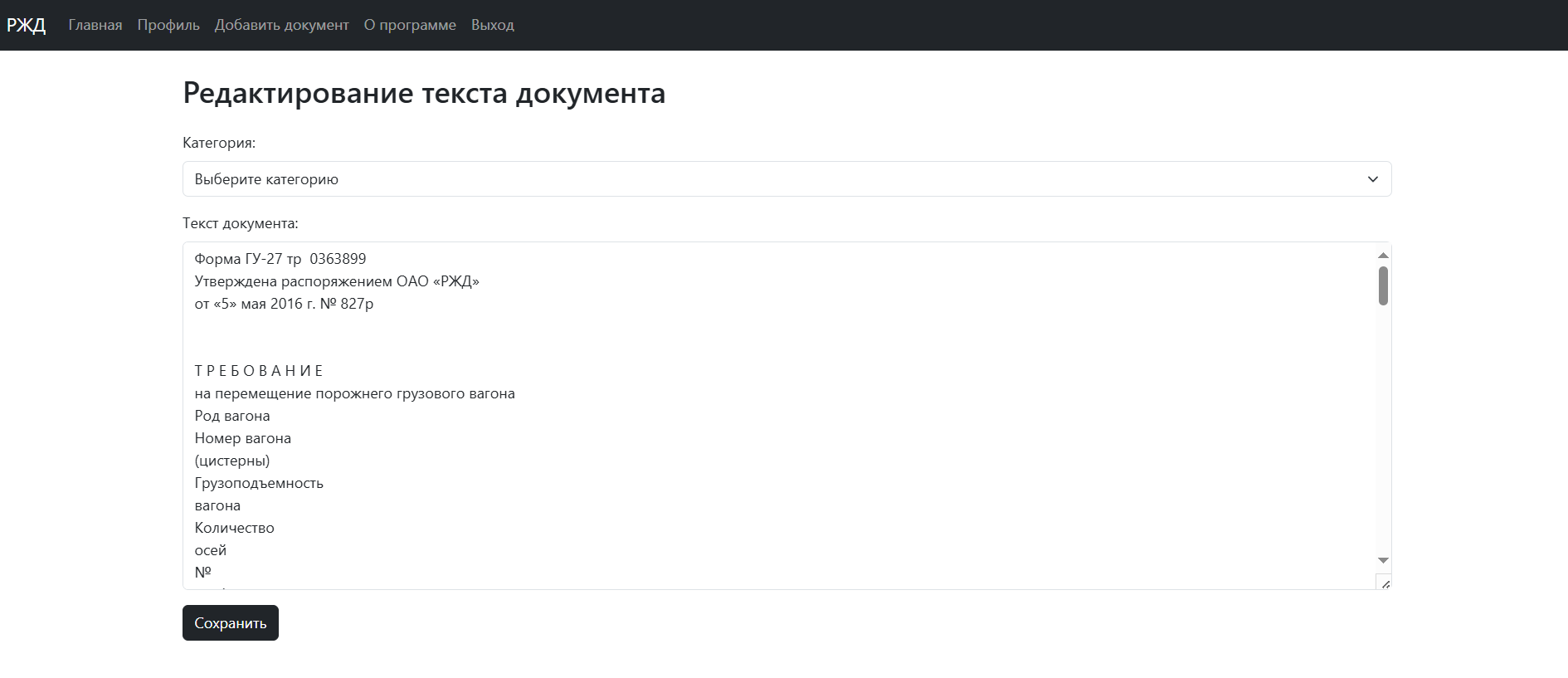


Рисунок 5 – Страница редактирования документа

Страница «О программе» предназначена для предоставления пользователям общей информации о разработанном приложении, его целях и основных возможностях. Здесь кратко описывается назначение системы как внутренней корпоративной платформы для управления документами и автоматизации документооборота.

На странице изложены ключевые функции приложения, включая загрузку, распознавание и редактирование документов, а также особенности используемых технологий, таких как OCR для преобразования изображений в редактируемый текст. Пользователям объясняется, как система помогает оптимизировать рабочие процессы, повысить эффективность взаимодействия сотрудников и обеспечить удобный доступ к корпоративным данным.

Дизайн страницы выполнен в простом и информативном стиле, чтобы обеспечить лёгкость восприятия и быстрое ознакомление с функционалом программы. Размещение текста и иллюстраций организовано так, чтобы подчеркнуть основные преимущества системы и её роль в автоматизации документооборота внутри организации (Рисунок 6).

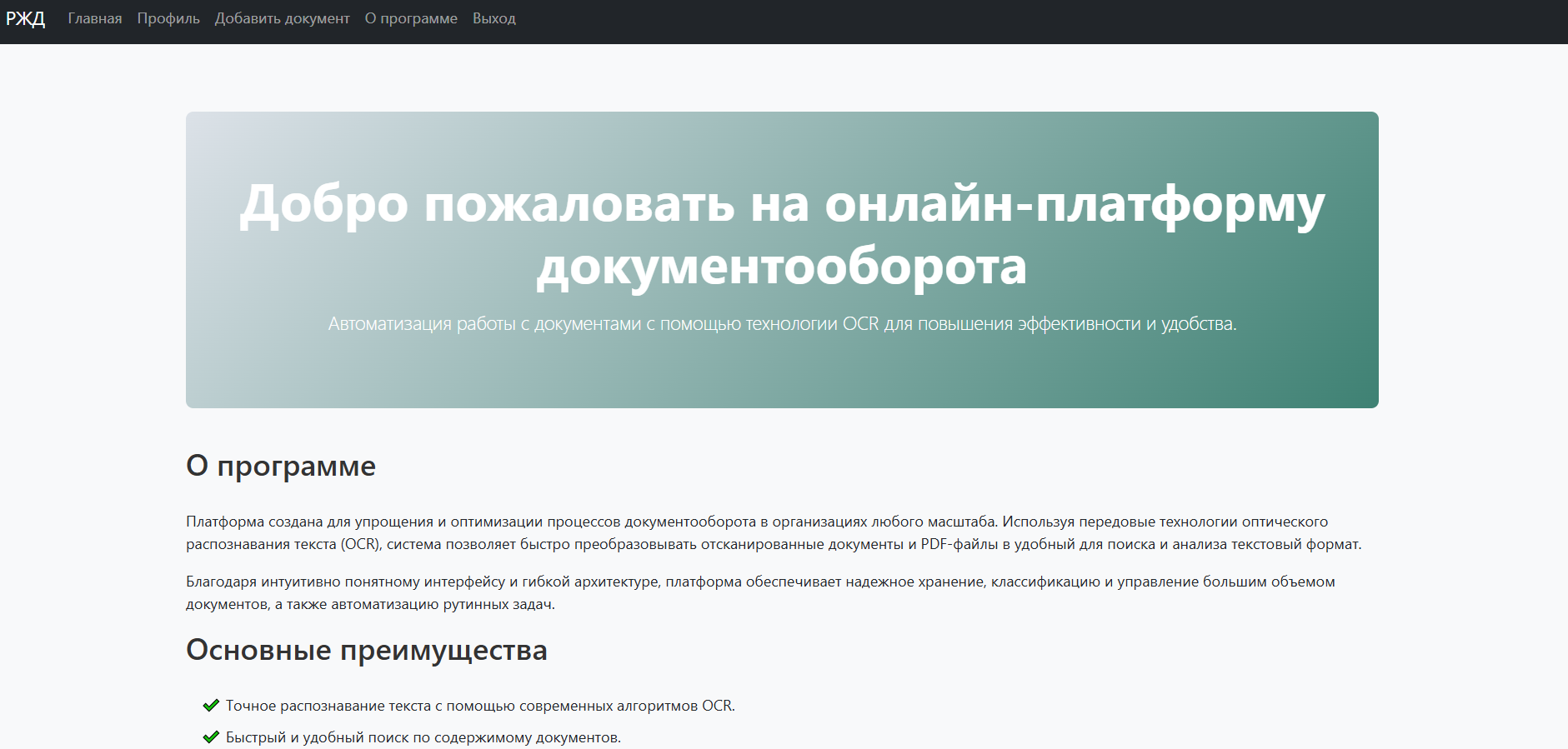


Рисунок 6 – Страница описания программы

## Подключение к базе данных

Flask поддерживает работу с различными системами управления базами данных, включая MySQL. Для удобного взаимодействия с базой данных через объекты Python в проекте используется ORM-библиотека SQLAlchemy, а именно её адаптация для Flask - Flask-SQLAlchemy [5]. Это обеспечивает плавную и бесшовную интеграцию, значительно упрощая выполнение основных операций с базой данных: создание (Create), чтение (Read), обновление (Update) и удаление (Delete).

Пример настройки подключения к заранее созданной базе данных MySQL, расположенной на локальном сервере:

app.config['SQLALCHEMY\_DATABASE\_URI'] = 'mysql+pymysql://username:password@localhost/database\_doc'

db = SQLAlchemy(app)

Перед началом работы важно установить необходимый драйвер для взаимодействия Flask с MySQL. В данном случае используется PyMySQL - популярный и надёжный драйвер, обеспечивающий стабильную работу с MySQL из Python-приложений.

Таким образом, использование Flask-SQLAlchemy вместе с подходящим драйвером позволяет эффективно управлять базой данных MySQL и реализовывать все необходимые операции с данными в приложении.

## Авторизация пользователя

В проекте для реализации авторизации пользователей использовалась библиотека Flask-Login [6], которая обеспечивает удобное и безопасное управление процессом входа и выхода из системы. Flask-Login позволяет работать с сессиями пользователей, хранить информацию о текущем пользователе.

Маршрут /login обрабатывает как GET-запросы для отображения страницы входа, так и POST-запросы для обработки введённых пользователем данных. При обращении к странице входа пользователь видит форму с полями для ввода имени пользователя и пароля (Рисунок 7).

При отправке формы (POST-запрос) происходит следующая логика:

* из базы данных извлекается пользователь с указанным именем;
* проверяется корректность введённого пароля;
* если данные неверны, пользователь получает сообщение об ошибке и остаётся на странице входа;
* если проверка успешна, вызывается функция login\_user(user, remember=form.remember\_me.data), которая создаёт сессию пользователя и сохраняет его состояние;
* после успешного входа происходит перенаправление на страницу, с которой пользователь пытался попасть в систему (аргумент next в URL), либо на главную страницу приложения.

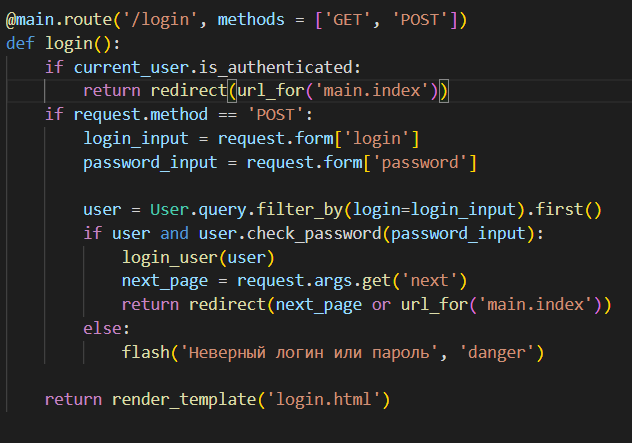


Рисунок 7 – маршрут входа в систему

## Безопасность страниц приложений

Для обеспечения безопасности и контроля доступа ко всем страницам приложения, кроме страницы входа, реализован централизованный механизм с помощью функции before\_request. Эта функция вызывается перед выполнением каждого запроса и проверяет, имеет ли пользователь право доступа к запрашиваемому маршруту (Рисунок 8).

В функции require\_login, зарегистрированной через декоратор @main.before\_request, реализована следующая логика:

* определен список маршрутов, доступных без авторизации.
* при каждом запросе проверяется, является ли текущий маршрут из этого списка;
* если маршрут не разрешён для неавторизованных пользователей и текущий пользователь не прошёл авторизацию (current\_user.is\_authenticated возвращает False), происходит перенаправление на страницу входа;
* при этом в параметр next передаётся исходный URL запроса, что позволяет после успешного входа вернуть пользователя на изначально запрошенную страницу.

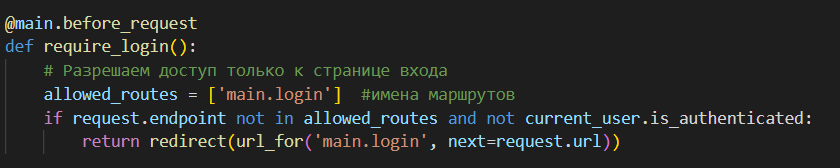


Рисунок 8 – проверка доступа к системе

## Добавление документов

В проекте реализована возможность загрузки различных типов документов, включая графические файлы, такие как PDF, PNG и другие форматы изображений. После загрузки происходит автоматическое распознавание текста с помощью встроенного OCR-модуля, что позволяет преобразовать содержимое в редактируемый формат для дальнейшей работы пользователя.

Кроме того, предусмотрена поддержка текстовых форматов документов (например, DOCX, TXT и др.). Для таких файлов распознавание текста не требуется, так как содержимое уже представлено в текстовом виде, что ускоряет процесс обработки и редактирования.

На странице добавления документа реализован предпросмотр загруженного файла, который позволяет пользователю ознакомиться с содержимым перед сохранением. В рамках предпросмотра пользователь также может выбрать категорию документа, что облегчает последующую классификацию и поиск (Рисунок 9).

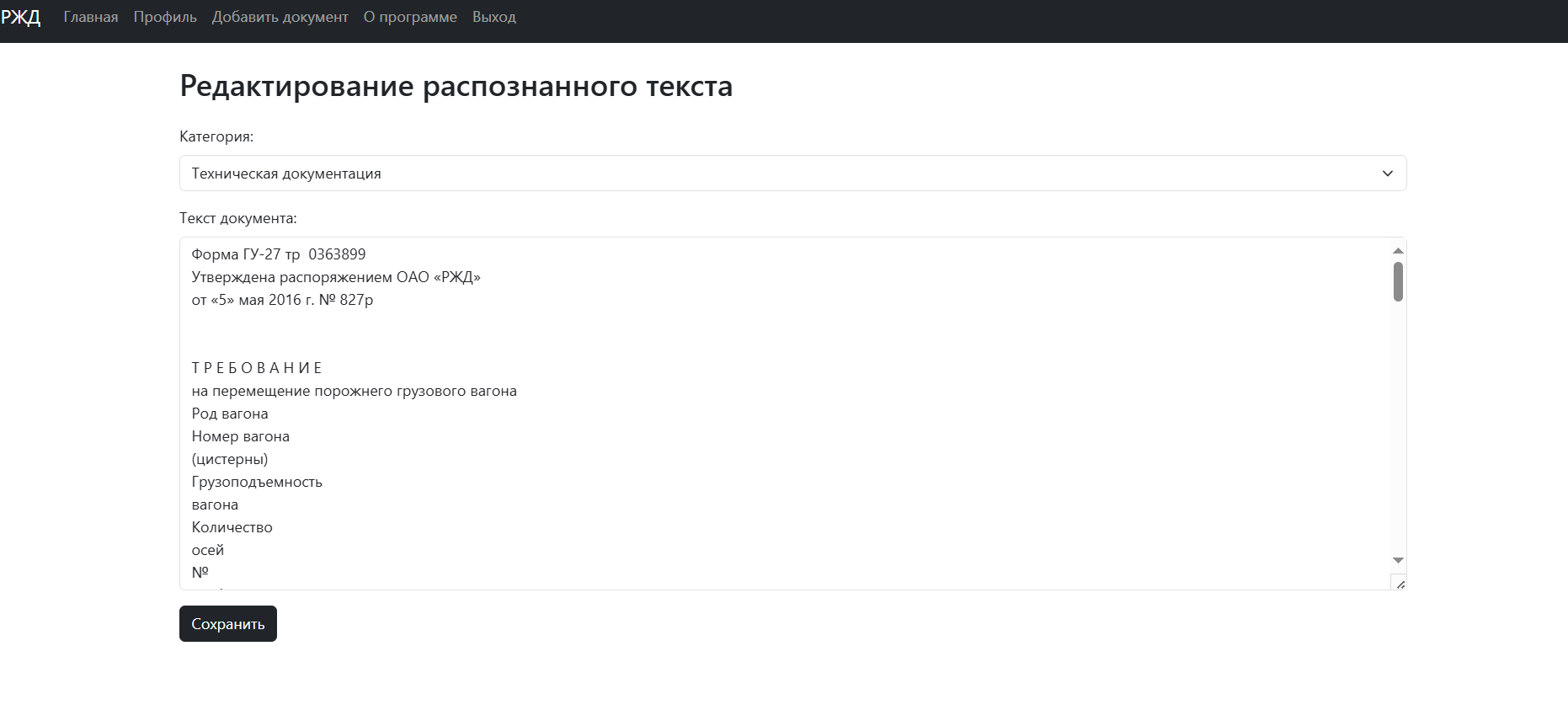


Рисунок 9 – предпросмотр документа

После подтверждения выбора и внесения необходимых данных, вся информация, включая содержимое документа и выбранную категорию, сохраняется в базе данных, обеспечивая полноценное управление документами в системе.

Таким образом, функционал страницы добавления документа в проекте уже обеспечивает универсальный и удобный инструмент для загрузки, просмотра, классификации и обработки различных типов документов, позволяя пользователям эффективно работать с содержимым и управлять материалами в системе.

## Управление правами доступа пользователей в проекте

В проекте реализована двухуровневая система прав доступа, которая обеспечивает разграничение функционала и безопасность при работе с документами.

* Администратор. Обладает полным доступом к системе. Он может создавать, редактировать и удалять любые документы. Администратор отвечает за поддержку и безопасность корпоративной системы;
* Пользователь. Имеет ограниченный доступ: может просматривать документы, загружать новые файлы и работать с собственным профилем.

Однако пользователь не имеет прав на удаление и редактирование чужих документов.

Контроль прав доступа реализован через проверку роли пользователя при каждом запросе к защищённым страницам и функциям. Если пользователь не обладает необходимыми правами, доступ к соответствующим функциям ограничивается, что обеспечивает безопасность данных и корректное распределение полномочий.

# ОЦЕНКА ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ

Современный рынок систем электронного документооборота представлен множеством готовых решений, каждое из которых обладает своими преимуществами и недостатками. В таблице 1 приведено сравнение ключевых характеристик популярных российских и зарубежных систем, таких как ELMA365, Directum RX, 1С:Документооборот и других.

Как видно из таблицы, многие из этих решений предлагают широкий функционал, включающий автоматизацию бизнес-процессов и интеграцию с внешними сервисами, однако зачастую они избыточны и требуют значительных затрат на внедрение и сопровождение. Однако для специфических задач Читинского ИВЦ РЖД они оказываются избыточными и зачастую неэффективными. Во-первых, сложность и обширность функционала таких систем приводит к увеличению времени обучения сотрудников и росту затрат на внедрение и сопровождение. Во-вторых, стоимость лицензирования и дополнительных модулей, например, для оптического распознавания текста (OCR), значительно повышает общие расходы, что не всегда оправдано для задач центра с ограниченным бюджетом. Кроме того, готовые решения часто требуют длительной и дорогостоящей кастомизации, чтобы адаптироваться под уникальные требования ИВЦ, что усложняет и замедляет процесс внедрения.

В отличие от этих систем, разработанное решение ориентировано на ключевые потребности Читинского ИВЦ РЖД — упорядоченное хранение, загрузку, редактирование и чтение документов с встроенной функцией сканирования и OCR. Такой подход позволяет избежать избыточности и сосредоточиться на действительно важных функциях, что значительно снижает затраты на внедрение и поддержку. Встроенный OCR обеспечивает быстрый и удобный переход бумажных документов в электронный формат без необходимости приобретения дополнительных модулей. Простота интерфейса и закрытый профиль доступа гарантируют безопасность данных и минимизируют время обучения сотрудников, что повышает общую эффективность работы.

Разработанное решение отличается высокой экономической эффективностью, поскольку его внедрение обойдётся организации всего в 15 тысяч рублей. При этом система позволит сократить время сотрудников, затрачиваемое на обработку и поиск документов, примерно на 30–40% благодаря централизованному хранению и встроенной функции оптического распознавания текста (OCR). Это существенное сокращение времени приведёт к заметному повышению производительности труда и снижению операционных затрат без необходимости крупных финансовых вложений. Таким образом, предложенная система является оптимальным и выгодным решением для автоматизации документооборота в Читинском ИВЦ РЖД, обеспечивая значительный качественный прорыв в организации рабочих процессов.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Критерий / Система | Разработанное решение | ELMA365 CSP (Российская) | Directum RX (Российская) | 1С:Документооборот (Российская) | Зарубежные СЭД (SAP, Microsoft) |
| Основной функционал | Загрузка, хранение, редактирование, чтение документов, встроенный OCR | Полный документооборот + BPM, интеграции, совместная работа | Полный документооборот, интеграции, управление проектами | Базовый и расширенный документооборот, модули согласования | Комплексные ERP и ECM системы, масштабируемость |
| Встроенный OCR | Есть (встроенный при загрузке) | Частично, через модули | Есть, но требует лицензирования | Отдельный модуль | Встроенный, но дорогостоящий |
| Простота использования | Минималистичный интерфейс | Сложный, требует обучения | Сложный, много функций | Средний уровень сложности | Высокая сложность, обучение |
| Стоимость внедрения и поддержки | Низкая, экономичное решение | Высокая, требует ресурсов | Высокая, комплексное внедрение | Средняя, зависит от версии | Очень высокая, дорогостоящее внедрение |
| Адаптация под нужды организации | Быстрая и простая | Гибкая, но сложная настройка | Гибкая, требует специалистов | Ограниченная гибкость | Требует глубокого кастомизации |
| Безопасность и доступ | Закрытый профиль, ограниченный доступ | Высокий уровень безопасности | Высокий уровень безопасности | Средний уровень | Очень высокий уровень безопасности |
| Интеграции | Минимальные | Широкие (ERP, ЭДО, AI) | Широкие (ЭДО, ERP, другие системы) | Интеграция с 1С и ЭДО | Широкие, но сложные |

Таблица 1 – сравнение готовых решений и разработанного решения

# ЗАКЛЮЧЕНИЕ

В ходе выполнения данного проекта были подробно изучены основные концепции и подходы к разработке веб-приложений. Проведен анализ и осуществлен выбор различных технологий, охватывающих весь спектр разработки — от создания пользовательского интерфейса до организации системы управления базами данных. На базе проведенного исследования был спроектирован и реализован пользовательский интерфейс, а также внедрены функция авторизации пользователей на платформе.

В процессе работы были достигнуты следующие результаты:

* определён и выбран набор программных инструментов для реализации проекта;
* создан и оформлен пользовательский интерфейс приложения;
* разработаны и документированы основные функциональные возможности на языке Python с применением веб-фреймворка Flask, обеспечивающие полноценную работу веб-приложения.

Кроме того, в ходе проекта успешно реализованы практические задачи программирования на Python. Изучены и применены возможности CSS-фреймворка Bootstrap 5.3 для создания современного и удобного графического интерфейса.

Таким образом, реализация проекта позволила получить ценный практический опыт в использовании современных технологий веб-разработки и сформировать базу знаний, необходимую для успешного выполнения поставленных задач.

# СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Python для новичков: сферы применения и возможности. [Электронный ресурс]: https://skillbox.ru/media/code/dlya\_chego\_nuzhen\_python/?ysclid=mb5wfviswm276828781
2. Фреймворк Flask: как он работает и зачем нужен [Электронный ресурс]: https://skillbox.ru/media/code/freymvork-flask-kak-on-rabotaet-i-zachem-nuzhen/
3. Что такое MySQL – как работает СУБД, области применения и преимущества. [Электронный ресурс]: https://selectel.ru/blog/mysql/?ysclid=mb5wjbttjv993995606
4. Что такое Adobe Acrobat. [Электронный ресурс]: https://speechify.com/ru/blog/what-is-adobe-acrobat/
5. Руководство по библиотеке SqlAlchemy в Python. [Электронный ресурс]: https://blog.skillfactory.ru/rukovodstvo-po-sqlalchemy-v-python/?ysclid=mb5wrftlwr98423854
6. Flask-Login 0.7.0 documentation. [Электронный ресурс]: https://flask-login.readthedocs.io/en/latest/

Разработка веб-интерфейса — это процесс создания удобной и визуально привлекательной среды взаимодействия пользователя с веб-ресурсом или приложением. Веб-интерфейс состоит из двух главных компонентов: фронтенда (front-end) и бэкенда (back-end), каждый из которых выполняет собственные задачи [16].

Компоненты веб-интерфейса

* Фронтенд (Front-end) — это всё, что видит и с чем взаимодействует пользователь, включая дизайн, графику, анимацию и интерфейсные элементы. Фронтенд-разработчики занимаются созданием HTML, CSS и JavaScript-кода, формирующего внешний облик страницы и её поведение.
* Бэкенд (Back-end) — это внутренняя логика и техническая сторона ресурса, обрабатывающая запросы пользователей, сохраняющая и возвращающая данные, реализующая правила безопасности и многое другое. Бэкенд-разработчики пишут серверный код на языках для разработанного приложения язык Python., организуя хранение данных и контролируя доступ к ним.

Процесс разработки включает в себя несколько этапов. Cначала проводится сбор требований и анализ нужд заказчика (организации), формируется техническое задание; затем начинается проектирование UI/UX — создаются макеты страниц, определяется расположение элементов интерфейса, выбираются цветовая гамма и стиль оформления; далее идет этап верстки и оформления, создаётся необходимая анимация и эффекты; параллельно ведется программирование серверной части приложения, пишутся обработчики запросов и API для взаимодействия с пользователями; после этого проводятся тесты и отладка интерфейса, устраняются баги и повышается производительность.

* 1. **Верстка страниц проекта.**

В ходе выполнения проекта был проведён этап согласования дизайна с организацией-заказчиком с целью обеспечения соответствия корпоративным стандартам и требованиям пользователя.

Для разработки шаблонов страниц и прототипов использовался программный продукт Adobe Acrobat [4]. Этот инструмент позволяет создавать интерактивные PDF-документы с визуально проработанными макетами, что обеспечивает наглядное представление структуры страниц, оформления элементов интерфейса и навигации.

Использование Adobe Acrobat позволило:

* обеспечить высокое качество визуализации макетов;
* упростить процесс внесения правок и корректировок;
* организовать удобную демонстрацию и обсуждение дизайна с заказчиком.

Шаблоны, созданные в Adobe Acrobat, послужили основой для дальнейшей разработки проекта. После утверждения макетов начался этап программной реализации, в ходе которого были разработаны функциональные страницы системы, соответствующие согласованным шаблонам.

Разработанное приложение представляет собой внутреннюю корпоративную систему, обеспечивающую удобный и безопасный доступ к функционалу и данным организации. После авторизации пользователи получают возможность работать с различными разделами системы, каждый из которых отвечает за определённые задачи.

Так, для просмотра и взаимодействия пользователям будут доступны следующие страницы:

* Главная страница – отображает полный список всех документов, доступных в системе;
* Профиль пользователя – содержит документы, загруженные и созданные конкретным пользователем;
* Добавление документа – предоставляет интерфейс для загрузки и создания новых документов в системе;
* О программе – содержит информацию о приложении, его назначении и основных функциях.

Страница входа является первым шагом для доступа к системе. Она обеспечивает безопасную авторизацию пользователей, позволяя подтвердить их личность и предоставить доступ к функционалу приложения в соответствии с правами и ролями. На странице предусмотрены поля для ввода логина и пароля, а также кнопка для подтверждения данных и входа в систему. (Рисунок 1).

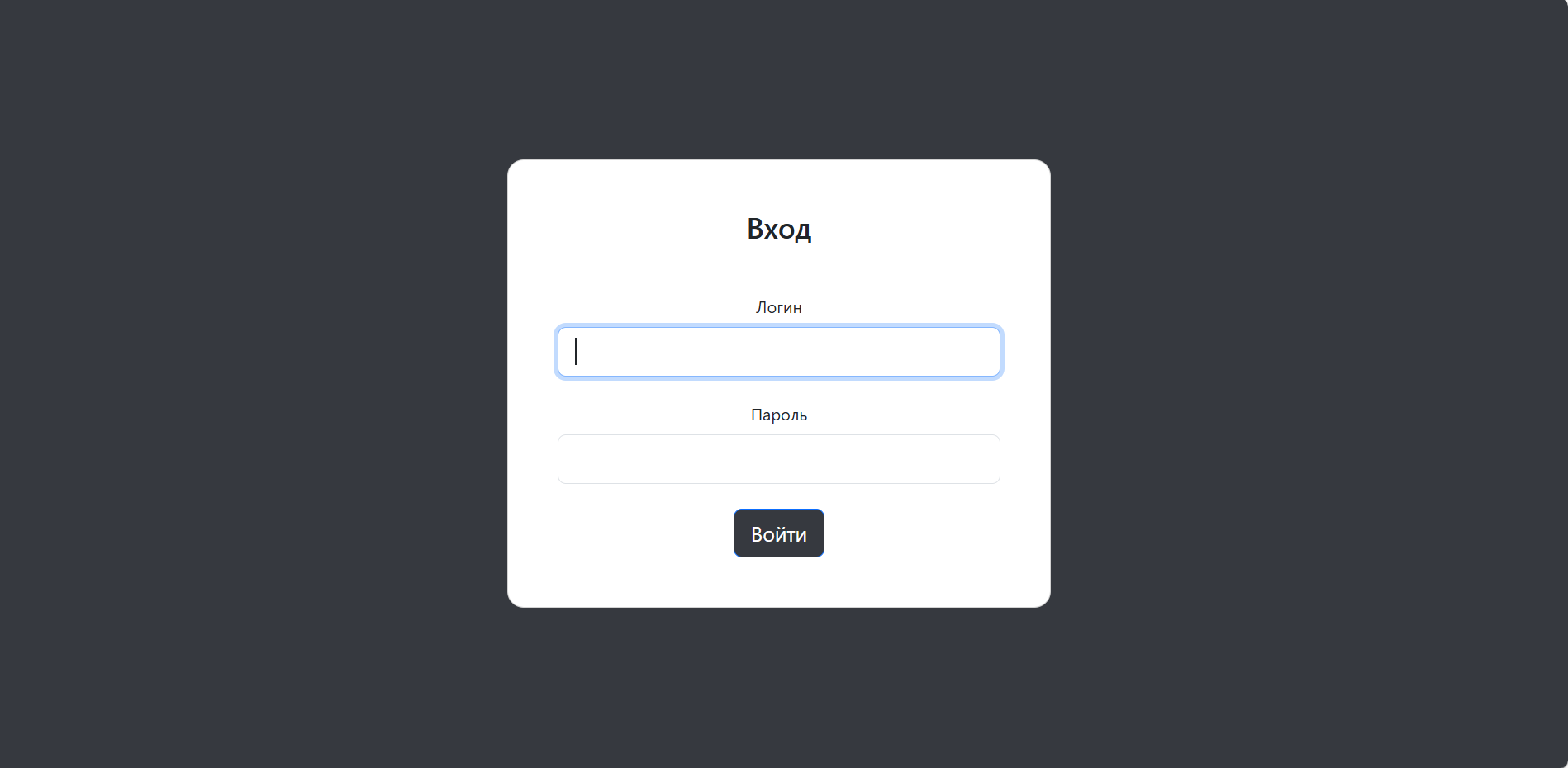


Рисунок 1 – Страница авторизации

Главная страница служит центральным элементом интерфейса и предназначена для отображения полного списка всех документов, доступных в системе. Здесь пользователи могут быстро просмотреть, найти и выбрать необходимые материалы для работы. При этом функционал страницы варьируется в зависимости от уровня доступа пользователя - разные роли получают соответствующие им возможности и инструменты (Рисунки 2.1 и 2.2).

На главной странице реализованы функции сортировки и фильтрации документов, что позволяет эффективно управлять большим объёмом данных. Интерфейс выполнен с учётом принципов простоты и логичности, чтобы обеспечить комфортное взаимодействие пользователя с системой.

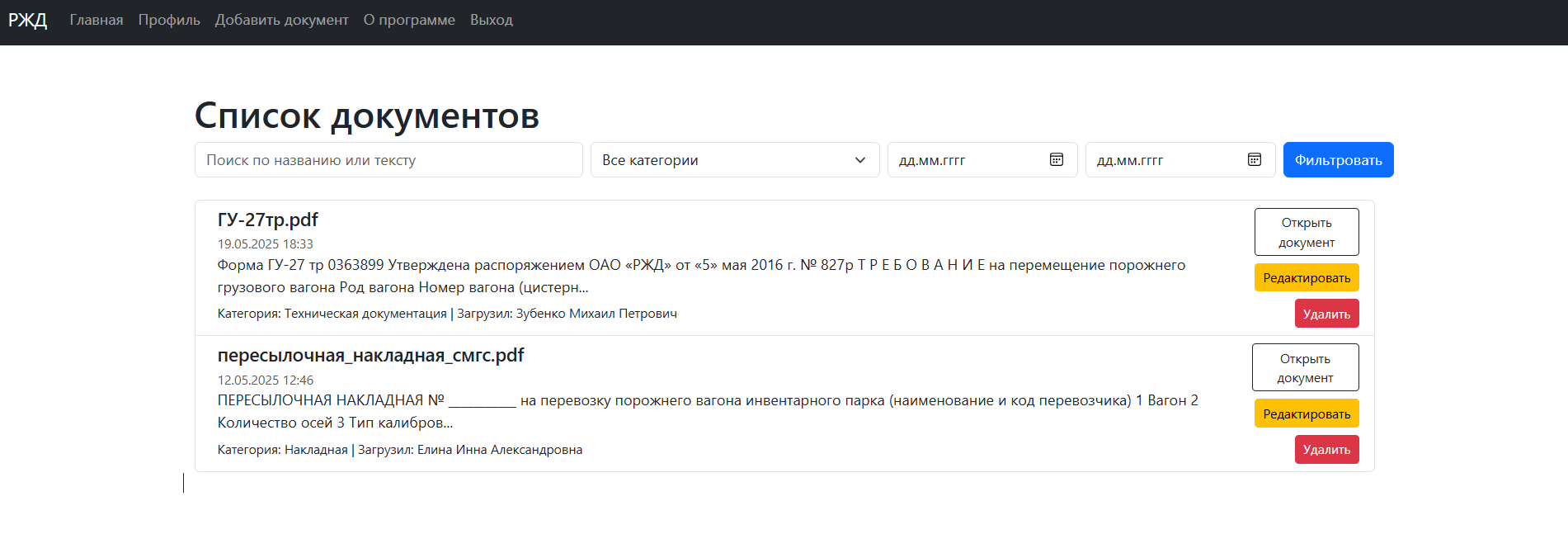


Рисунок 2.1 – Главная страница. Администратор

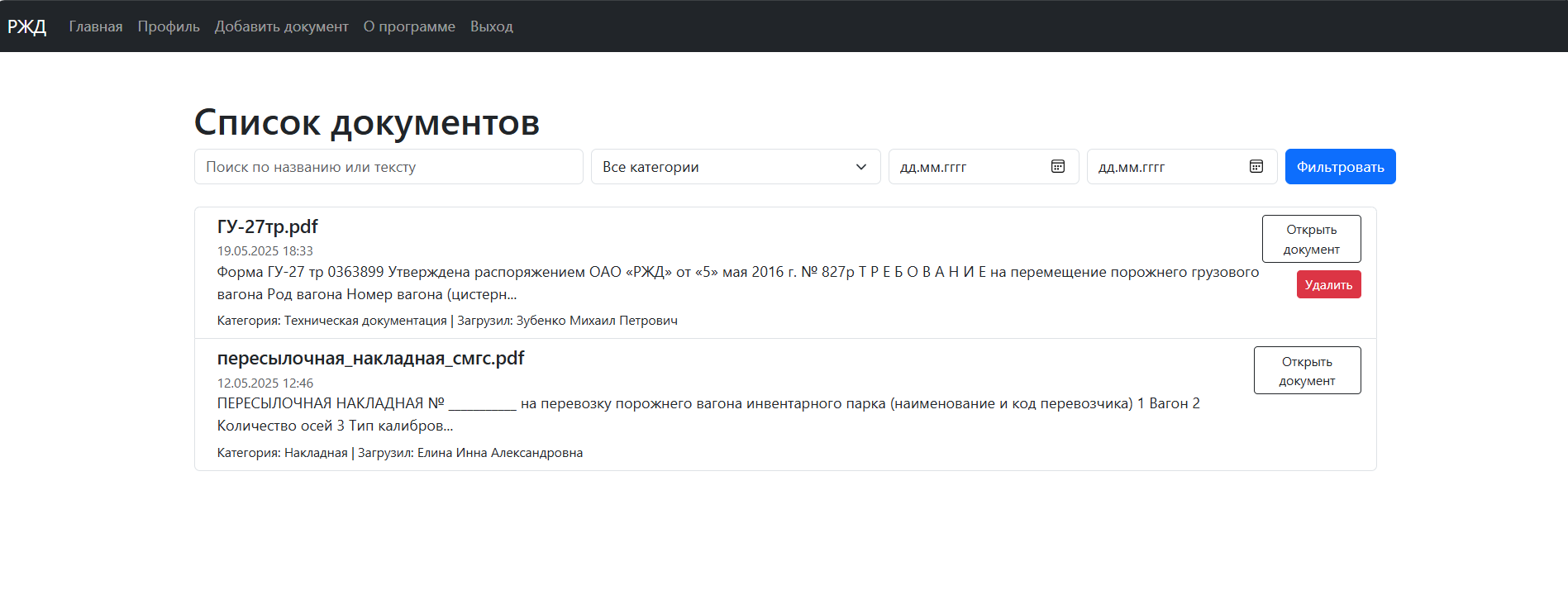


Рисунок 2.2 – Главная страница. Пользователь

Страница профиля предназначена для отображения документов, которые он загрузил или создал в системе. Здесь пользователь может просмотреть список своих материалов, отсортировать их по дате или другим параметрам, а также получить быстрый доступ к редактированию и управлению этими документами.

Интерфейс страницы профиля разработан с учётом удобства и прозрачности: все элементы расположены логично, что облегчает навигацию и работу с файлами. Кроме того, на странице реализованы функции фильтрации и поиска, позволяющие быстро найти нужный документ среди большого количества.

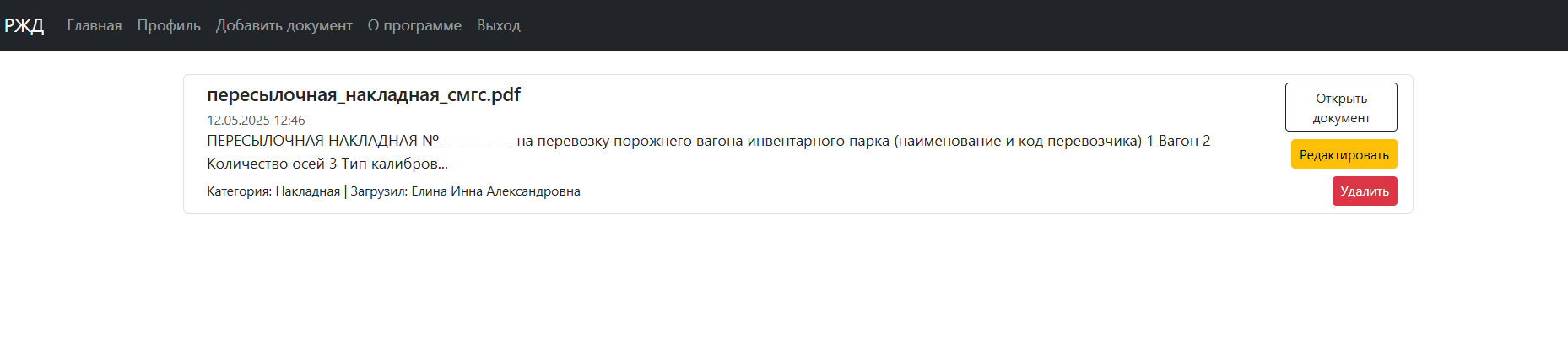


Рисунок 3.1 – Страница профиля. Администратор

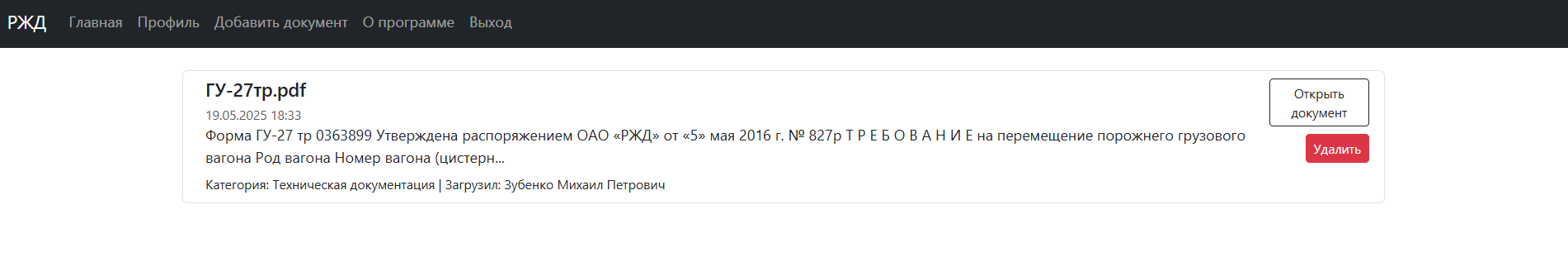


Рисунок 3.2 – Страница профиля. Пользователь

В рамках разработанной корпоративной системы страница добавления документа предоставляет пользователю возможность загружать новые файлы в систему с минимальными усилиями. Интерфейс включает поле выбора файла с устройства. (Рисунок 4)

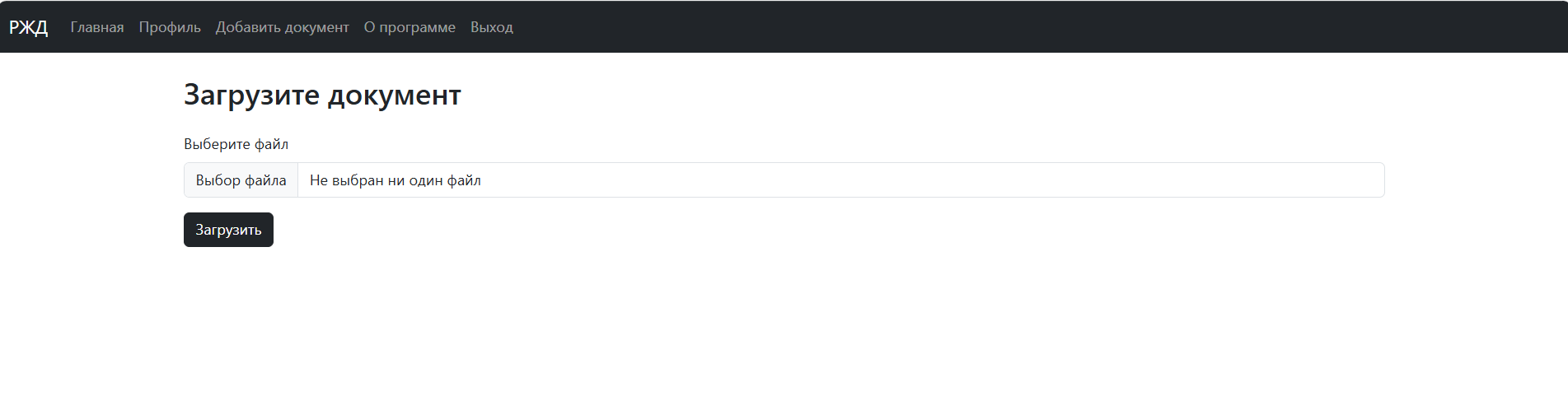


Рисунок 4 – Страница добавления документа

По завершении распознавания пользователь перенаправляется на страницу редактирования, где может внести необходимые корректировки в текст документа. Загруженный и отредактированный документ становится доступен для просмотра и дальнейшей работы как на главной странице, так и в профиле пользователя (Рисунок 5).

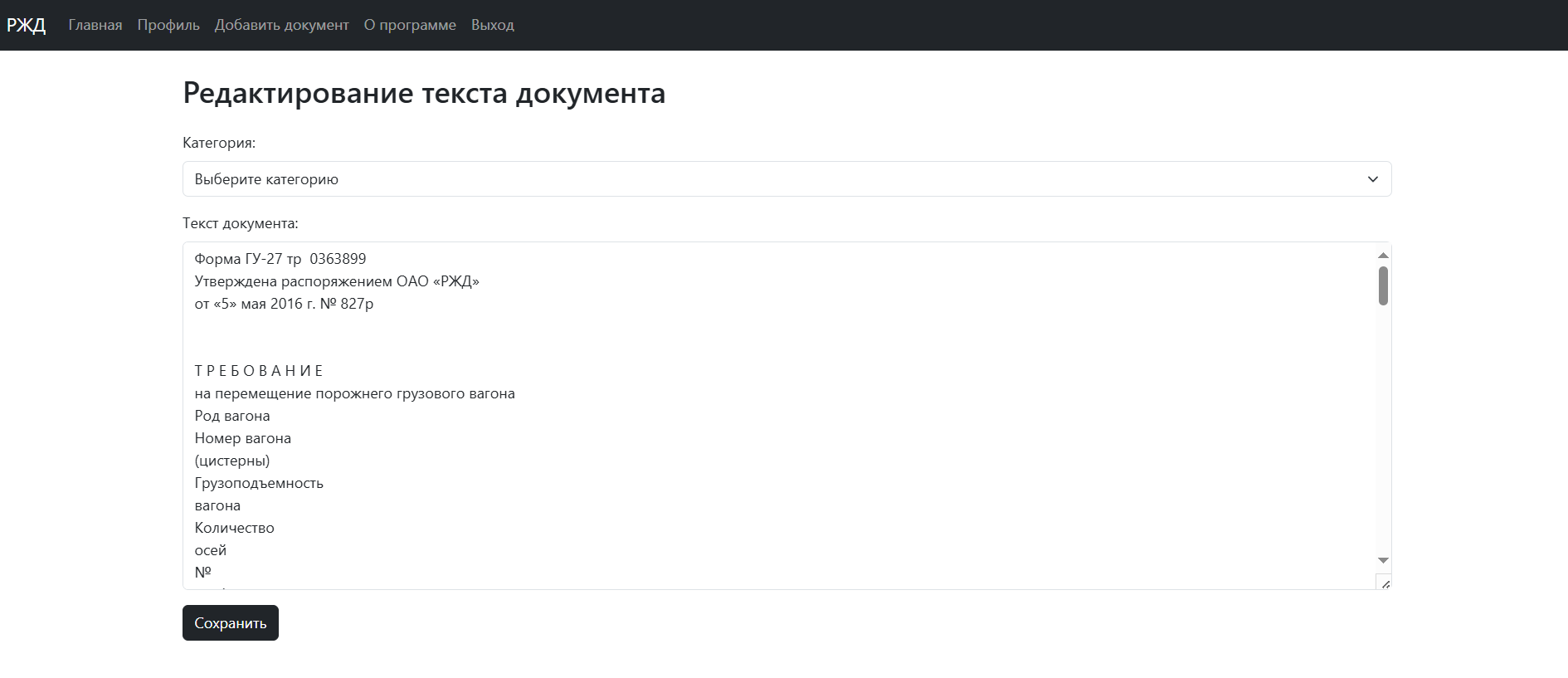


Рисунок 5 – Страница редактирования документа

Страница «О программе» предназначена для предоставления пользователям общей информации о разработанном приложении, его целях и основных возможностях. Здесь кратко описывается назначение системы как внутренней корпоративной платформы для управления документами и автоматизации документооборота.

На странице изложены ключевые функции приложения, включая загрузку, распознавание и редактирование документов, а также особенности используемых технологий, таких как OCR для преобразования изображений в редактируемый текст. Пользователям объясняется, как система помогает оптимизировать рабочие процессы, повысить эффективность взаимодействия сотрудников и обеспечить удобный доступ к корпоративным данным.

Дизайн страницы выполнен в простом и информативном стиле, чтобы обеспечить лёгкость восприятия и быстрое ознакомление с функционалом программы. Размещение текста и иллюстраций организовано так, чтобы подчеркнуть основные преимущества системы и её роль в автоматизации документооборота внутри организации (Рисунок 6).

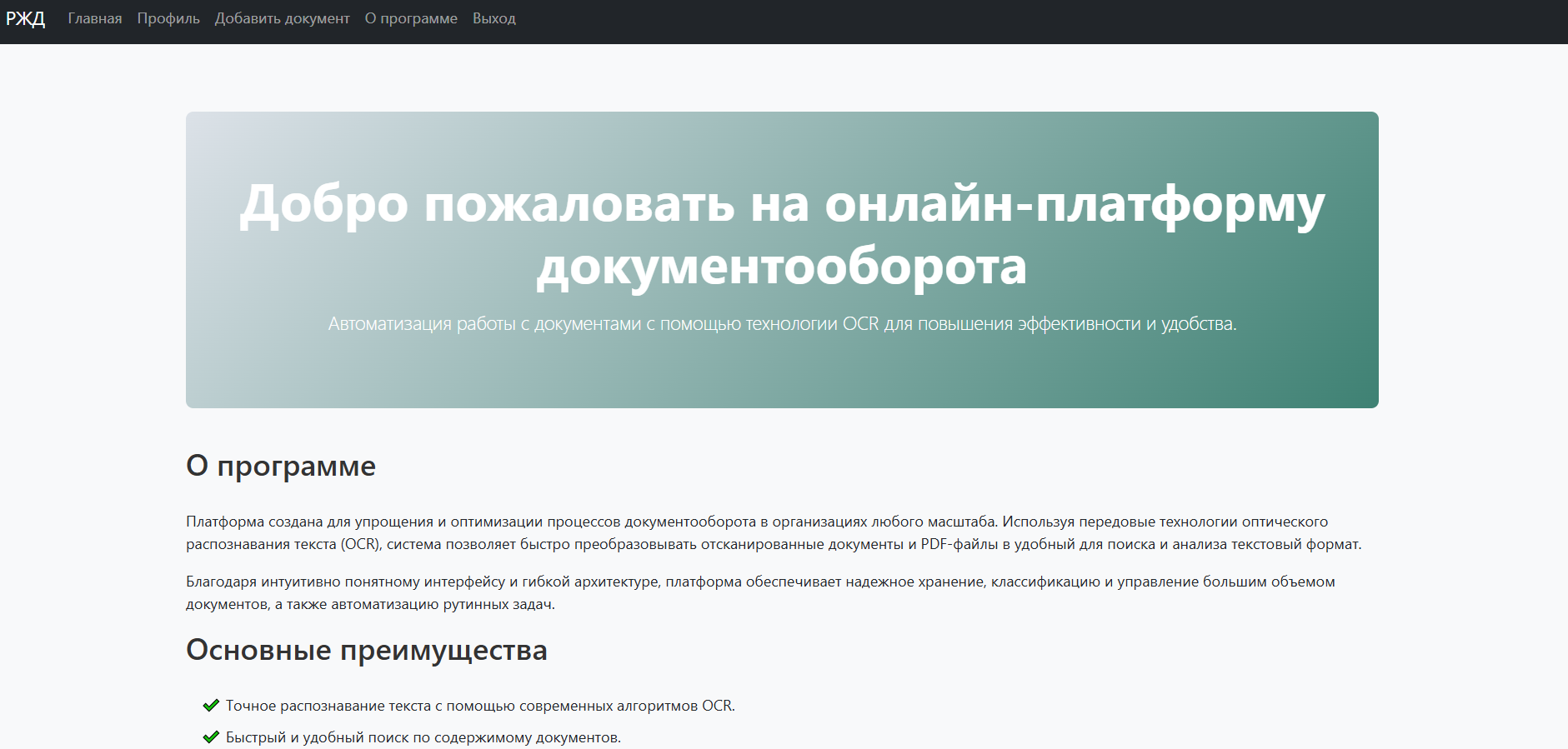


Рисунок 6 – Страница описания программы

* 1. **Подключение к базе данных**

Перед началом работы с базой данных необходимо предварительно создать нужные таблицы и установить устойчивое соединение с самой базой данных

Создали модели (прикрепить код баз), картинка с подключением

Flask поддерживает работу с различными системами управления базами данных, включая MySQL. Для удобного взаимодействия с базой данных через объекты Python в проекте используется ORM-библиотека SQLAlchemy, а именно её адаптация для Flask - Flask-SQLAlchemy [5]. Это обеспечивает плавную и бесшовную интеграцию, значительно упрощая выполнение основных операций с базой данных: создание (Create), чтение (Read), обновление (Update) и удаление (Delete).

Пример настройки подключения к заранее созданной базе данных MySQL, расположенной на локальном сервере:

app.config['SQLALCHEMY\_DATABASE\_URI'] = 'mysql+pymysql://username:password@localhost/database\_doc'

db = SQLAlchemy(app)

Перед началом работы важно установить необходимый драйвер для взаимодействия Flask с MySQL. В данном случае используется PyMySQL - популярный и надёжный драйвер, обеспечивающий стабильную работу с MySQL из Python-приложений.

Таким образом, использование Flask-SQLAlchemy вместе с подходящим драйвером позволяет эффективно управлять базой данных MySQL и реализовывать все необходимые операции с данными в приложении.

* 1. **Авторизация пользователя**

В проекте для реализации авторизации пользователей использовалась библиотека Flask-Login [6], которая обеспечивает удобное и безопасное управление процессом входа и выхода из системы. Flask-Login позволяет работать с сессиями пользователей, хранить информацию о текущем пользователе.

Маршрут /login обрабатывает как GET-запросы для отображения страницы входа, так и POST-запросы для обработки введённых пользователем данных. При обращении к странице входа пользователь видит форму с полями для ввода имени пользователя и пароля (Рисунок 7).

При отправке формы (POST-запрос) происходит следующая логика:

* из базы данных извлекается пользователь с указанным именем;
* проверяется корректность введённого пароля;
* если данные неверны, пользователь получает сообщение об ошибке и остаётся на странице входа;
* если проверка успешна, вызывается функция login\_user(user, remember=form.remember\_me.data), которая создаёт сессию пользователя и сохраняет его состояние;
* после успешного входа происходит перенаправление на страницу, с которой пользователь пытался попасть в систему (аргумент next в URL), либо на главную страницу приложения.

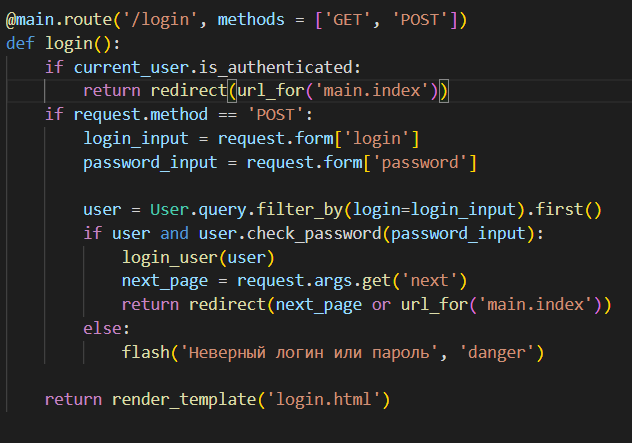


Рисунок 7 – маршрут входа в систему

* 1. **Безопасность страниц приложения**

Для обеспечения безопасности и контроля доступа ко всем страницам приложения, кроме страницы входа, реализован централизованный механизм с помощью функции before\_request. Эта функция вызывается перед выполнением каждого запроса и проверяет, имеет ли пользователь право доступа к запрашиваемому маршруту (Рисунок 8).

В функции require\_login, зарегистрированной через декоратор @main.before\_request, реализована следующая логика:

* определен список маршрутов, доступных без авторизации.
* при каждом запросе проверяется, является ли текущий маршрут из этого списка;
* если маршрут не разрешён для неавторизованных пользователей и текущий пользователь не прошёл авторизацию (current\_user.is\_authenticated возвращает False), происходит перенаправление на страницу входа;
* при этом в параметр next передаётся исходный URL запроса, что позволяет после успешного входа вернуть пользователя на изначально запрошенную страницу.

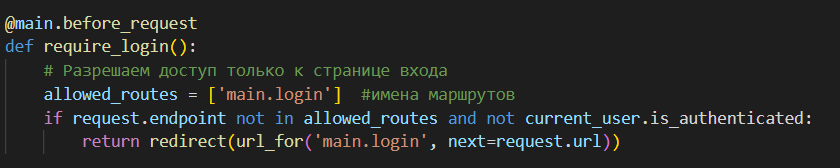


Рисунок 8 – проверка доступа к системе

* 1. **Добавление документов**

В проекте реализована возможность загрузки различных типов документов, включая графические файлы, такие как PDF, PNG и другие форматы изображений. После загрузки происходит автоматическое распознавание текста с помощью встроенного OCR-модуля, что позволяет преобразовать содержимое в редактируемый формат для дальнейшей работы пользователя.

Кроме того, предусмотрена поддержка текстовых форматов документов (например, DOCX, TXT и др.). Для таких файлов распознавание текста не требуется, так как содержимое уже представлено в текстовом виде, что ускоряет процесс обработки и редактирования.

На странице добавления документа реализован предпросмотр загруженного файла, который позволяет пользователю ознакомиться с содержимым перед сохранением. В рамках предпросмотра пользователь также может выбрать категорию документа, что облегчает последующую классификацию и поиск (Рисунок 9).

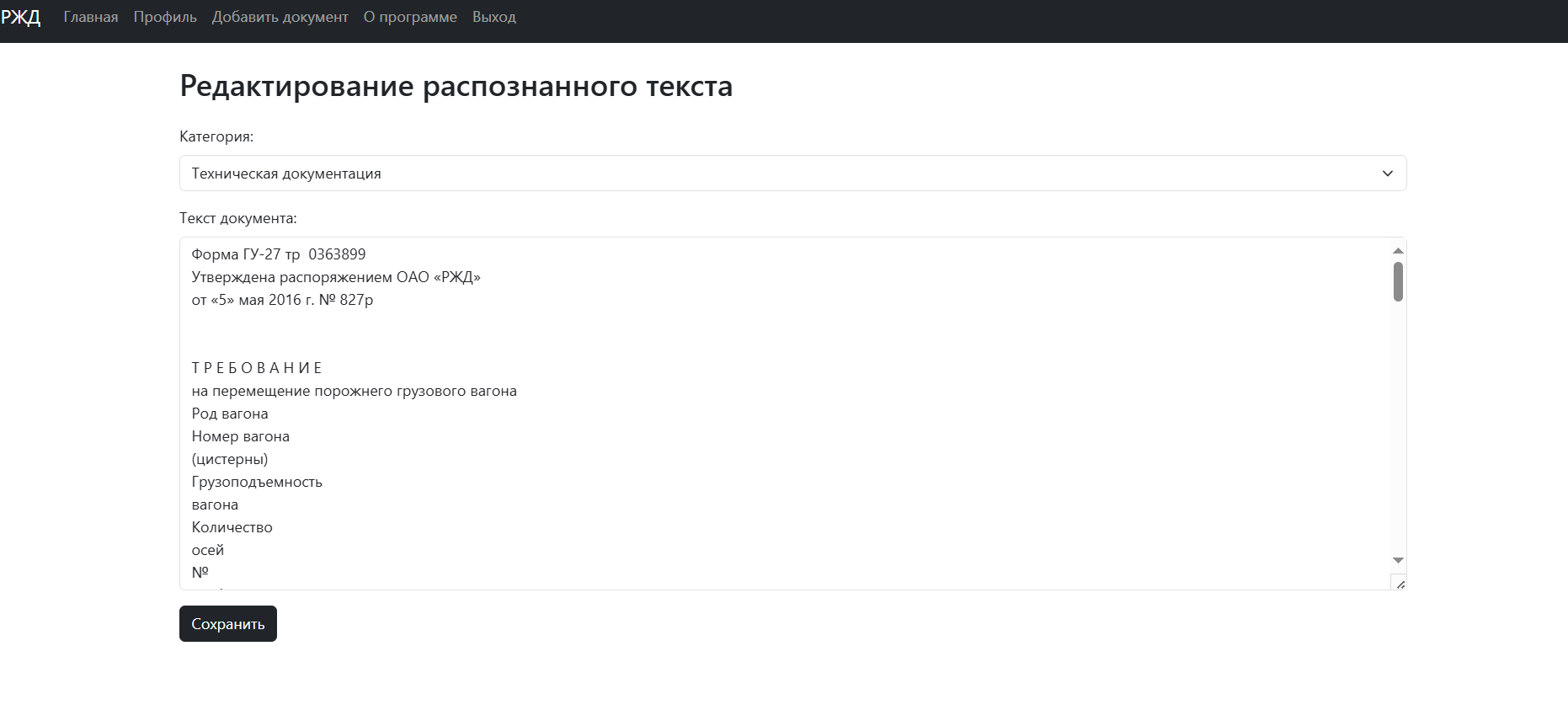


Рисунок 9 – предпросмотр документа

После подтверждения выбора и внесения необходимых данных, вся информация, включая содержимое документа и выбранную категорию, сохраняется в базе данных, обеспечивая полноценное управление документами в системе.

Таким образом, функционал страницы добавления документа в проекте уже обеспечивает универсальный и удобный инструмент для загрузки, просмотра, классификации и обработки различных типов документов, позволяя пользователям эффективно работать с содержимым и управлять материалами в системе.

* 1. **Управление правами доступа пользователей в проекте**

В проекте реализована двухуровневая система прав доступа, которая обеспечивает разграничение функционала и безопасность при работе с документами.

* Администратор. Обладает полным доступом к системе. Он может создавать, редактировать и удалять любые документы. Администратор отвечает за поддержку и безопасность корпоративной системы;
* Пользователь. Имеет ограниченный доступ: может просматривать документы, загружать новые файлы и работать с собственным профилем.

Однако пользователь не имеет прав на удаление и редактирование чужих документов.

Контроль прав доступа реализован через проверку роли пользователя при каждом запросе к защищённым страницам и функциям. Если пользователь не обладает необходимыми правами, доступ к соответствующим функциям ограничивается, что обеспечивает безопасность данных и корректное распределение полномочий.