МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

«Российский экономический университет им. Г.В. Плеханова»

Московский приборостроительный техникум

**Курсовой проект**

ПМ 02 Осуществление интеграции программных модулей

МДК 11.01 «Технология разработки и защиты баз данных»

Специальность 09.02.07 «Информационные системы и программирование»

Квалификация: Программист

Тема: «Разработка системы для детского нейроцентра «Природа разума»»

**Пояснительная записка**

Листов: 51

|  | Руководитель  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Е.Ю. Бойцова  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2025 год  Исполнитель  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ / С.А. Мельников  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2025 год |
| --- | --- |

2025

Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Российский экономический университет имени Г.В. Плеханова»

Московский приборостроительный техникум

«Утверждаю»

Заместитель директора по учебной работе

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Д.А. Клопов

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2025 г.

Задание

на выполнение курсового проекта (курсовой работы)

Мельникову Сергею Александровичу

(фамилия, имя, отчество студента — полностью)

студенту группы П50-6-21 специальности 09.02.07 «Информационные системы и программирование» по МДК 11.01 «Технология разработки и защиты баз данных»

1. Исходные данные к проекту (работе):
   1. Тема: «Разработка программного комплекса для интернет магазина «FlowBow»
   2. Состав курсового проекта:
      1. Задание КП
      2. Пояснительная записка
      3. Программа (исходные данные) на электронном носителе
      4. Презентация и инсталляционный пакет программы на электронном носителе
   3. Содержание пояснительной записки:

ВВЕДЕНИЕ

1. ОБЩАЯ ЧАСТЬ
   1. Цель разработки
   2. Средства разработки
2. СПЕЦИАЛЬНАЯ ЧАСТЬ
   1. Постановка задачи
      1. Входные данные предметной области
      2. Выходные данные предметной области
      3. Требования к проекту
   2. Внешняя спецификация
      1. Описание задачи
      2. Входные и выходные данные
      3. Методы
      4. Тесты
      5. Контроль целостности данных
   3. Проектирование
      1. Схема архитектуры приложения
      2. Логическая схема данных
      3. Физическая схема данных
      4. Структурная схема
      5. Функциональная схема
      6. Диаграмма классов
      7. Схема тестирования
      8. Схема пользовательского интерфейса
   4. Результат работы программы
3. ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ
   1. Инструментальные средства
   2. Отладка программы
   3. Защитное программирование
   4. Характеристики программы

ЗАКЛЮЧЕНИЕ

СПИСОК ИСПОЛЬЗУЕМЫХ МАТЕРИАЛОВ

1. ПРИЛОЖЕНИЕ А. Сценарий тестовых испытаний.

2. ПРИЛОЖЕНИЕ А.1. Результат тестовых испытаний.

3. ПРИЛОЖЕНИЕ Б. Руководство пользователя.

4. ПРИЛОЖЕНИЕ В. Скрипт базы данных.

5. ПРИЛОЖЕНИЕ Г. Текст программы.

2. Содержание задания по проекту (работе) ⁠— перечень вопросов, подлежащих разработке

|  | Разрабатываемый вопрос | Объем от всего задания, % | Срок выполнения |
| --- | --- | --- | --- |
| А | Описательная часть проекта (введение, общее описание и т. д.) | 5 | 16.01.2025 |
| 1. | Введение | - | 16.01.2025 |
| 2. | Цель разработки | - | 16.01.2025 |
| 3. | Средства разработки | - | 16.01.2025 |
| Б | Анализ задачи и её постановка | 5 | 23.01.2025 |
| 1. | Определение требований к программе | - | 23.01.2025 |
| 2. | Спецификация программы (описание задачи, описание входных и выходных данных, методы) | - | 23.01.2025 |
| 3. | Тесты, контроль целостности данных | - | 23.01.2025 |
| В | Проектирование и реализация | 55 | 03.02.2025 |
| 1. | Схемы проекта (схема архитектуры, логическая схема данных, физическая схема данных, функциональная и структурная схемы, диаграмма классов, схема тестирования, схема пользовательского интерфейса) | - | 03.02.2025 |
| 2. | Реализация в инструментальной среде | - | 03.02.2025 |
| Г | Технологическая часть проекта | 5 | 25.02.2025 |
| 1. | Инструментальные средства разработки | - | 25.02.2025 |
| 2. | Отладка программа | - | 25.02.2025 |
| 3. | Защитное программирование | - | 25.02.2025 |
| 4. | Характеристика программы | - | 25.02.2025 |
| Д | Программная документация | 20 | 09.03.2025 |
| 1. | Приложение А. Сценарий тестовых испытаний | - | 19.03.2025 |
| 2. | Приложение А.1. Результаты тестовых испытаний | - | 19.03.2025 |
| 3. | Приложение Б. Руководство пользователя | - | 23.03.2025 |
| 4. | Приложение В. Скрипт базы данных | - | 23.03.2025 |
| 5. | Приложение Г. Текст программы | - | 23.03.2025 |
| 1. | Программа на машинном носителе.  Информация на носителе разбита на разделы: эксплуатационный пакет, тексты программы, документация. | - | 11.04.2025 |

Руководитель курсового проекта (работы) Екатерина Юрьевна Бойцова, преподаватель

«11» апреля 2025 года \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Е. Ю. Бойцова /

Дата выдачи курсового задания «13» января 2025 года

Срок сдачи законченного проекта (работы) «11» апреля 2025 года

Задание принял к исполнению

«13» января 2025 года \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Мельников. С. А./

СОДЕРЖЕНИЕ

[1. ОБЩАЯ ЧАСТЬ 6](#_Toc1)

[1.1 Цель Разработки 6](#_Toc2)

[1.2 Средства разработки 6](#_Toc3)

[2. СПЕЦИАЛЬНАЯ ЧАСТЬ 9](#_Toc4)

[2.1 Постановка задачи 9](#_Toc5)

[2.1.1 Входные данные предметной области 9](#_Toc6)

[2.2. Внешняя спецификация 11](#_Toc7)

[2.3 Проектирование 19](#_Toc8)

[2.4. Результаты работы программы 29](#_Toc9)

[3. ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ 35](#_Toc10)

[3.1. Инструментальные средства разработки 35](#_Toc11)

[3.2. Отладка программы 36](#_Toc12)

[3.3. Защитное программирование 37](#_Toc13)

[3.4. Характеристики программы 37](#_Toc14)

[ЗАКЛЮЧЕНИЕ 46](#_Toc15)

### ВВЕДЕНИЕ

В условиях стремительного развития технологий, где прозрачность, безопасность и скорость взаимодействия становятся основными требованиями, всё больше людей обращаются к программным комплексам, таким как "Природа разума", вместо традиционных решений. Это особенно актуально для сферы нейроинтеграции и развития интеллектуальных систем, где необходимы доверие и защита данных.

Программный комплекс "Природа разума" не только упрощает процесс анализа и обработки информации, но и предлагает инновационные решения, такие как алгоритмы нейросетевого обучения для автоматизации принятия решений и интеграцию различных источников данных. Такие системы обеспечивают высокий уровень безопасности, минимизируют риски ошибок и предоставляют пользователям доступ к мощным аналитическим инструментам в любое время.

Одной из главных проблем традиционных систем остаётся непрозрачность процессов и высокая уязвимость к сбоям или манипуляциям. Из-за этого пользователи часто сталкиваются с недостоверными результатами и высокими затратами на обслуживание. Кроме того, традиционные платформы не всегда способны обеспечить должный уровень автоматизации и надёжности, что снижает удобство и безопасность работы с данными.

Чтобы решить эти проблемы, был создан программный комплекс "Природа разума". Платформа предлагает пользователям доступ к центру развития и нейроинтеграции, а также функции, упрощающие и ускоряющие процесс анализа данных. "Природа разума" позволяет интегрировать различные нейросетевые модели и отслеживать статус процессов в реальном времени.

Техническая основа комплекса построена на современных технологиях, таких как нейронные сети и распределённые вычисления, что гарантирует высокую безопасность и надёжность сервиса. Интеграция с аналитическими инструментами и системами мониторинга позволяет отслеживать историю операций и предлагать персонализированные рекомендации пользователям.

"Природа разума" — это не просто программный комплекс, а надёжный партнёр, обеспечивающий прозрачность, безопасность и эффективность каждого процесса, будь то анализ данных, разработка интеллектуальных систем или интеграция нейросетевых решений в бизнес-процессы.

# ОБЩАЯ ЧАСТЬ

## Цель Разработки

Создать автоматизированную систему для формирования заключений и рекомендаций для детей после диагностики, которая позволит педагогам эффективно использовать базу данных с упражнениями и товарами. Система должна обеспечивать удобный интерфейс для ввода индивидуальных заключений, интеграцию с базой данных для добавления упражнений и товаров, а также обеспечить безопасный доступ для сотрудников через общую авторизацию.

## Средства разработки

В качестве средств вычислительной техники использовался персональный компьютер, характеристики которого представлены в Таблице 1

Таблица 1 - Технические средства для разработки

| № | Тип оборудования | Наименование оборудования |
| --- | --- | --- |
| 1 | 2 | 3 |
| Персональный компьютер | | |
| 1 | Размер экрана: | 26" |
| 2 | Разрешение экрана: | 1920х1080 |
| 3 | Линейка процессора: | Inter Core i5 10300H 4.5GHz |
| 4 | Количество ядер процессора: | 8 |
| 5 | Оперативная память: | 16 ГБ |
| 6 | Тип видеокарты: | Дискретный |
| 7 | Видеокарта: | Nvidia GeForce 1650 |
| 8 | Конфигурация накопителей: | SSD |
| 9 | Общий объем всех накопителей: | 512 ГБ |

Для разработки программного продукта инвентаризации и учета оборудования использовались программные средства, представленные в Таблице 2.

Таблица 2 - Программный средства

| № | Тип средства | Название средства | Назначение |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| 1 | Операционная система | Arch Linux | Организация  взаимодействия программ  и пользователя |
| 2 | Инструментальное  средство разработки | Visual Studio Code 1.96 | Разработка клиентского приложения и API |
| 3 | Средство разработки мобильного приложения | Android Studio Meerkat 2024.3.1 | Разработка мобильного приложения |
| 4 | Средство разработки базы данных | PgAdmin4 | Проектирование и разработка базы данных |
| 5 | Пакет офисных  программ | ONLYOFFICE | Реализация технической  документации |

# СПЕЦИАЛЬНАЯ ЧАСТЬ

## Постановка задачи

Разработать базу данных, мобильное и клиентское приложение для центра разваития

Создать пользовательский интерфейс: разработать удобный, интуитивно понятный и визуально привлекательный интерфейс для комфортного использования веб-сервиса.

Разработать административную панель: предоставить администраторам удобный инструмент для управления данными в базе данных, включая функции редактирования, добавления и удаления информации.

Обеспечить безопасность системы: внедрить меры защиты для безопасного взаимодействия пользователей и администраторов с веб-сервисом и его API.

Выполнить тестирование и оптимизацию: провести всестороннее тестирование для выявления и устранения ошибок, а также оптимизировать производительность системы для быстрого и стабильного взаимодействия пользователей с веб-сервисом.

# 2.1.1 Входные данные предметной области

Входными данными веб-сервиса являются персональные данные клиента, данные о заказе, адрес доставки

2.1.2 Выходные данные предметной области

Выходными данными веб-сервиса являются информация о Пациентах, Пользователях, Ролях и Документах.

2.1.3 Подробные требования к проекту

Требования к функциям:

Программа должна реализовать следующие функции:

* Создание Ролей.
* Изменение Ролей.
* Удаление Ролей.
* Создание Пользователя.
* Изменение Пользователя.
* Удаление Пользователя.
* Создание Пациента.
* Изменение Пациента.
* Удаление Пациента.
* Создание Документа.
* Изменение Документа.
* Удаление Документа.
* Просмотр Пользователей.
* Просмотр Ролей.
* Просмотр Пациентов.
* Просмотр Документов.
* Скачивание Документа.
* Поиск Пользователей.
* Поиск Ролей.
* Поиск Документов.
* Поиск Пациентов.
* Фильтрация Пользователей
* Фильтрация Документов
* Фильтрация Пациентов

Условия эксплуатации:

* Программный комплекс должен быть доступен в локальной сети

Требования к составу и параметрам технических средств:

* Сайт должен корректно работать во всех популярных современных браузерах, включая Google Chrome, Mozilla Firefox, Safari и Microsoft Edge, поддерживающих стандарты HTML5 и CSS3.
* Интерфейс должен быть адаптивным, обеспечивая удобное и корректное отображение на различных устройствах, таких как смартфоны, планшеты и настольные компьютеры, независимо от разрешения экрана.
* Минимальные технические требования для комфортного использования сайта включают:
* Интернет-соединение со скоростью не менее 10 Мбит/с.
* Устройства с оперативной памятью от 2 ГБ и разрешением экрана 360x640 пикселей или выше для мобильных устройств.
* Сайт должен обеспечивать быструю загрузку страниц, минимизируя время ожидания пользователей за счет оптимизации изображений, скриптов и стилей.
* Поддержка доступности для пользователей с ограниченными возможностями, включая совместимость с экранными дикторами, контрастными цветовыми схемами и возможностью навигации с клавиатуры.

2.2. Внешняя спецификация

2.2.1 Описание задачи

Основной задачей является разработка программного комплекса “Природа Разума” для удобного взаимодействия с клиентами и реализации менеджмента документов.

Данное решение состоит из:

Реляционной базы данных для хранения информации о товара и клиентах, веб – приложения для взаимодейтсвия с веб - сайтом, API для взаимодейтсвия базы данных с backend – стороной приложения, мобильного приложения для представления информации о данной организации.

База Данных Хранит следующие данные:

* Поддиректории (Название)
* Пользователь (ФИО, Логин, Пароль, Активен ли, Роль, Документы)
* Роль (Название, Описание)
* Документ (Название, Данные, Пациент, Автор)
* Пациент (ФИО, Возраст)
* Пользовательские роли:
* Администратор
* Педагог

На этапе анализа предметной области были определены 2 роли – Администратор и Педагог. Если Пользователь не зарегистрирован в системе, то для дальнейших действий ему необходимо обратиться к Администратору для создания его аккаунта, если же он уже прошёл данный этап, то ему необходимо войти в систему при помощи логина и пароля, где он получит возможность использовать основной функционал программы (Рисунок 1).

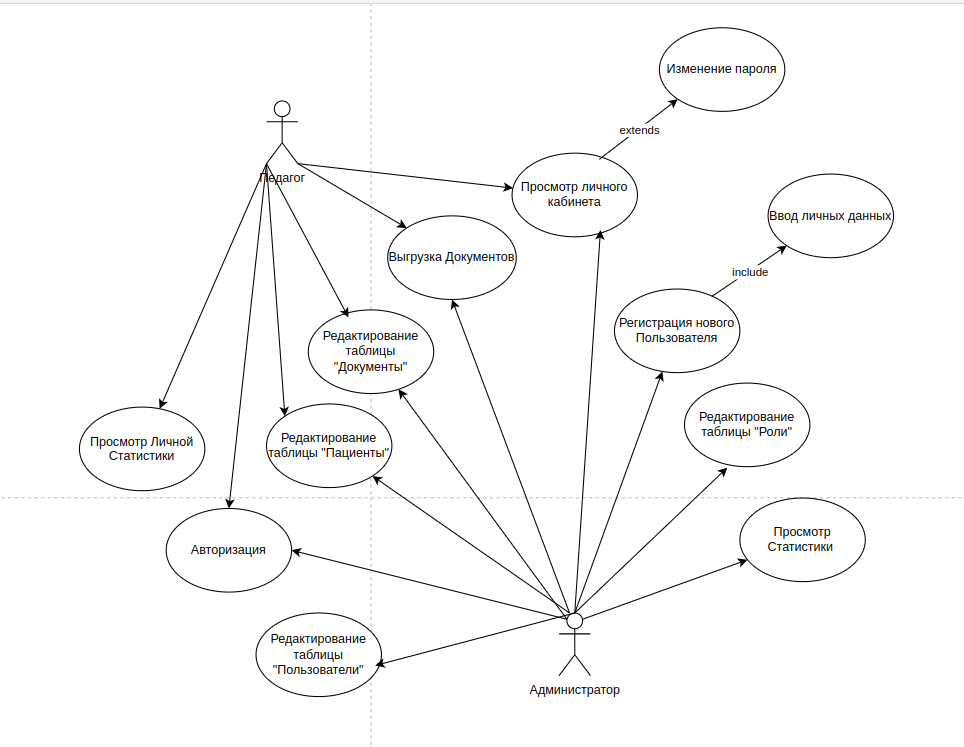


Рисунок 1 - Диаграмма прецедентов

2.2.2. Входные и выходные данные

В таблице 3 представлены входные данные, вводимые пользователем при взаимодействии с программой. При вводе информации пользователем в поля происходит проверка валидации данных, на соответствие формата данных, количество введенных символов, а также дополнительных проверок на ввод спецсимволов.

Таблица 3 - Входные данные

| № | Поле | Тип данных | Ограничения | Формат ввода | Описание |
| --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 |
| Страница «Авторизация» | | | | | |
| 1 | Login | строка | [a-z, A-Z, 0-9] {5,50} | Поле для ввода логина | Ввод логина для авторизации |
| 2 | Password | строка | [a-z, A-Z, 0-9, !@#$%^&\*()\_+\-=[\]{};':"\\|,.<>/?] {6,100} | Поле для ввода пароля | Ввод пароля для авторизации |
| Страница «Пользователи» | | | | | |
| 1 | Login | Строка | [a-z, A-Z, 0-9] {5,50} | Поле ввода для логина | Ввод логина для создания пользователя |
| 2 | Password | Строка | [a-z, A-Z, 0-9, !@#$%^&\*()\_+\-=[\]{};':"\\|,.<>/?] {6,100} | Поле ввода для пароля | Ввод пароля для создания пользователя |
| 4 | Role | Сущность | - | Выпадающий список для выбора роли | Выбор роли для создания пользователя |
| 5 | isActive | Булево-значение | - | Свитч для изменения статуса Пользователя | Изменение статуса для создания пользователя |
| 6 | Name | Строка | [а-я, А-Я]  {2, 100} | Поле для ввода имени | Ввод имени для создания пользователя |
| 7 | Surname | Строка | [а-я, А-Я]  {2, 100} | Поле для ввода фамилии | Ввод фамилии для создания пользователя |
| 8 | ThirdName | Строка | [а-я, А-Я]  {0, 100} | Поле для ввода Отчества | Ввод отчества для создания пользователя |
| Страница «Пациенты» | | | | | |
| 1 | Name | Строка | [а-я, А-Я]  {2, 100} | Поле для ввода имени | Ввод имени для создания пациента |
| 2 | Surname | Строка | [а-я, А-Я]  {2, 100} | Поле для ввода фамилии | Ввод фамилии для создания пациента |
| 3 | ThirdName | Строка | [а-я, А-Я]  {0, 100} | Поле для ввода Отчества | Ввод отчества для создания пациента |
| 4 | Age | Цифра | [0-9] | Поле для ввода возраста | Ввод возраста для создания пациента |
| Страница «Документы» | | | | | |
| 1 | Name | Строка | [а-я, А-Я, a-z, A-Z, 0-9] {3, 255} | Поле для ввода названия | Ввод названия для создания документа |
| 2 | File | Байты | - | Input для выбора документа | Выбор файла для создания документа |
| 3 | DirectoryType | Сущность | - | Выпадающее меню для выбора директории | Выбор поддиректории для создания документа |
| 4 | Patient | Сущность | - | Выпадающее меню для выбора Пацента | Выбор пациента для создания документа |
| Страница «Роли» | | | | | |
| 1 | Name | Строка | [a-z, A-Z, 0-9] {3, 255} | Поле для ввода названия | Ввод названия для создания роли |
| 2 | Description | Строка | [a-z, A-Z, а-я, А-Я, 0-9, !@#$%^&\*()\_+\-=[\]{};':"\\|,.<>/?] {6,1000} | Поле для ввода описания | Ввод описания для создания роли |

Выходные данные указаны в таблице 4.

Таблица 4 - Выходные данные

| № | Поле | Тип данных | Ограничения | Описание |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| Страница «Пользователи» | | | | |
| 1 | Login | Строка | [a-z, A-Z, 0-9] {5,50} | Вывод логина |
| 2 | Role | Сущность | - | Вывод роли |
| 3 | isActive | Булево-значение | - | Вывод статуса |
| 4 | Name | Строка | [а-я, А-Я]  {2, 100} | Вывод имени |
| 5 | Surname | Строка | [а-я, А-Я]  {2, 100} | Вывод фамилии |
| 6 | ThirdName | Строка | [а-я, А-Я]  {0, 100} | Вывод отчества |
| Страница «Документы» | | | | |
| 1 | Name | Строка | [а-я, А-Я, a-z, A-Z, 0-9] {3, 255} | Вывод названия |
| 2 | DirectoryType | Сущность | - | Вывод поддиректории |
| 3 | Patient | Сущность | - | Вывод пациента |
| Страница «Роли» | | | | |
| 1 | Name | Строка | [a-z, A-Z, 0-9] {3, 255} | Вывод названия |
| 2 | Description | Строка | [a-z, A-Z, а-я, А-Я, 0-9, !@#$%^&\*()\_+\-=[\]{};':"\\|,.<>/?] {6,1000} | Вывод описания |
| Страница «Пациенты» | | | | |
| 1 | Name | Строка | [а-я, А-Я]  {2, 100} | Вывод имени |
| 2 | Surname | Строка | [а-я, А-Я]  {2, 100} | Вывод фамилии |
| 3 | ThirdName | Строка | [а-я, А-Я]  {0, 100} | Вывод отчества |
| 4 | Age | Цифра | [0-9] | Вывод возраста |

2.2.3 Методы

При разработке данного программного комплекса была использована "Чистая Архитектура", которая обеспечивает четкое разделение ответственности между компонентами приложения и способствует созданию гибкой и поддерживаемой системы. Этот подход позволяет изолировать бизнес-логику от деталей реализации, что упрощает тестирование и модификацию приложения.

"Чистая Архитектура" организована в несколько уровней, каждый из которых выполняет свою конкретную роль:

1. Внешний уровень (Interface Adapters: Этот уровень отвечает за взаимодействие с пользователем и внешними системами. Он включает в себя контроллеры и представления, которые обрабатывают входящие запросы и формируют ответы. Здесь происходит преобразование данных в формат, удобный для использования на более глубоких уровнях.

2. Уровень приложений (Application Layer: На этом уровне реализуется бизнес-логика приложения. Он управляет процессами и правилами, определяющими, как данные должны обрабатываться. Этот уровень не зависит от внешних библиотек или фреймворков, что позволяет легко изменять его без влияния на другие компоненты.

3. Доменный уровень (Domain Layer: Этот уровень содержит основные бизнес-объекты и правила, которые определяют поведение системы. Он является сердцем приложения, где сосредоточена вся бизнес-логика. Доменный уровень не зависит от других уровней, что обеспечивает его независимость и возможность повторного использования.

4. Уровень инфраструктуры (Infrastructure Layer: Этот уровень отвечает за взаимодействие с внешними системами, такими как базы данных, API и другие сервисы. Он реализует конкретные детали, необходимые для работы приложения, но не содержит бизнес-логики. Это позволяет легко заменять или изменять инфраструктурные компоненты без влияния на бизнес-логику.

Благодаря такому разделению, "Чистая Архитектура" способствует улучшению тестируемости, поддерживаемости и расширяемости системы. Каждый уровень может быть изменён или протестирован независимо, что минимизирует риски и упрощает процесс разработки. Это также позволяет командам работать параллельно над различными аспектами приложения, что ускоряет его создание и внедрение.

### 2.2.4. Тесты

Для тестирования центра развития был разработан и выполнен набор тестовых сценариев, направленных на проверку соответствия системы её функциональным требованиям. Тестирование проводилось с использованием позитивных и негативных сценариев, применялись методы белого ящика и черного ящика, а также были созданы и выполнены модульные тесты для проверки отдельных компонентов.

Детальная информация о тест-кейсах и результатах тестирования представлена в разделе «Приложение В. Сценарии тестирования и результаты испытаний».

### 2.2.5. Контроль целостности данных

Была проведена работа для обеспечения целостности данных. Во все поля была добавлена защита от некорректно-вводимой информации, а также обработаны все исключения приложения.

В Таблице 5 представлены аномалии, которые могут возникнуть в процессе эксплуатации программного продукта.

Таблица 5 – Контроль целостности данных

| №  п/п | Ситуация | Аномалия | Реакция программы | Описание |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| 1 | Вход в личный кабинет | Не заполнены или введены неправильно данные для одного из обязательных для заполнения полей | Отображается окно с текстом о неудачной авторизацией и ссылкой на ошибку | Данные должны быть верно заполнены |
| 2 | Добавление или изменение таблицы Пользователи | Не заполнены или введены неправильно данные для одного из обязательных для заполнения полей | Отображается подсказка под полем с неправильными данными или же пустым значением | Данные должны быть верно заполнены |
| 3 | Добавление или изменение таблицы Роли | Не заполнены или введены неправильно данные для одного из обязательных для заполнения полей | Отображается подсказка под полем с неправильными данными или же пустым значением | Данные должны быть верно заполнены |
| 4 | Добавление или изменение таблицы Документы | Не заполнены или введены неправильно данные для одного из обязательных для заполнения полей | Отображается подсказка под полем с неправильными данными или же пустым значением | Данные должны быть верно заполнены |
| 5 | Добавление или изменение таблицы Пациенты | Не заполнены или введены неправильно данные для одного из обязательных для заполнения полей | Отображается подсказка под полем с неправильными данными или же пустым значением | Данные должны быть верно заполнены |
| 6 | Изменение пароля в личном кабиенет | Не заполнены или введены неправильно данные для одного из обязательных для заполнения полей | Отображается тоаст с ошибкой | Данные должны быть верно заполнены |

## 2.3 Проектирование

2.3.1 Архитектурная схема программы

На рисунке 2 представлена архитектурная схема программного комплекса. Она позволяет понять взаимодействие между модулями комплекса.

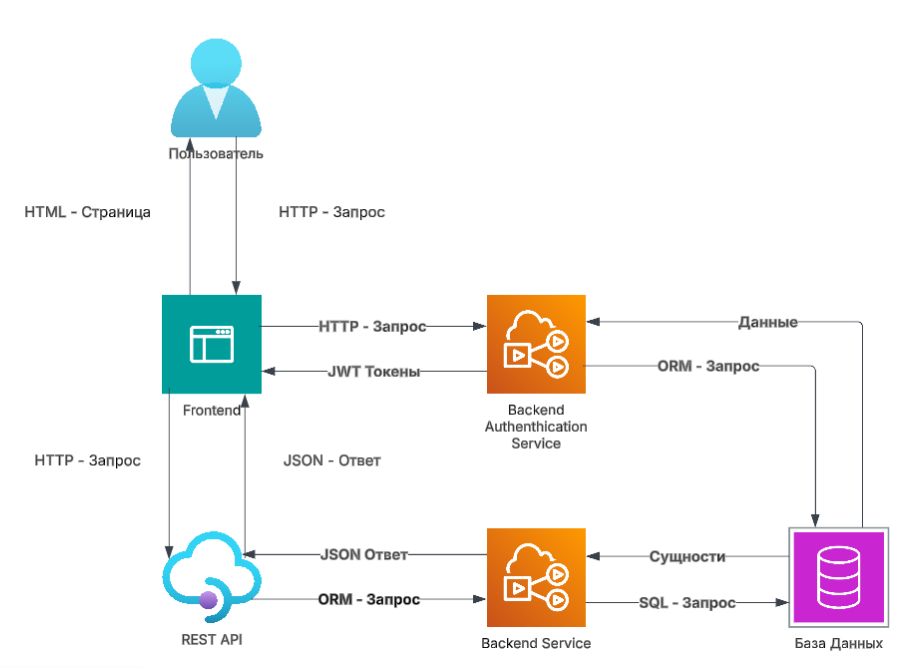


Рисунок 2 – Архитектурная схема

2.3.2. Функциональная схема программы

На рисунке 3 представлена функциональная схема программного продукта веб-приложения интернет, на которой графически продемонстрированы все доступные функции каждой из ролей Web приложения.

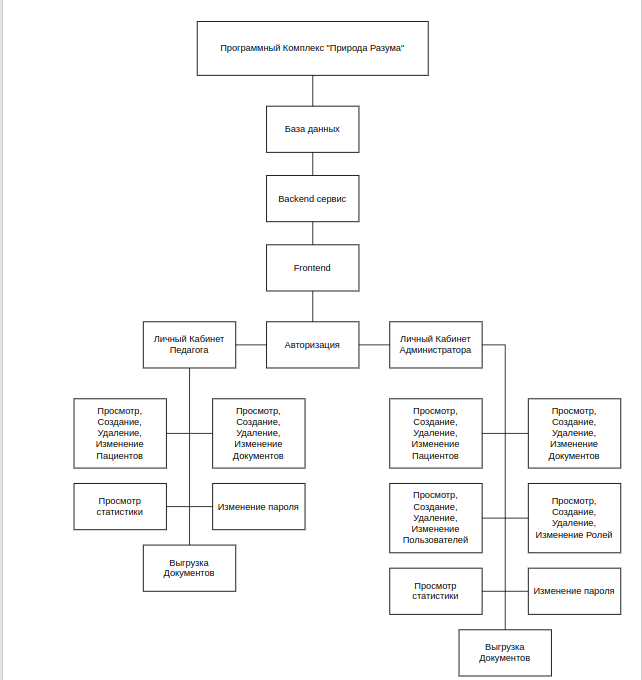


Рисунок 3 - Функциональная схема

### 2.3.3. Схема данных

При проектировании базы данных была составлена логическая схема данных с учетом третьей нормализированной формы, с которой взаимодействует программный комплекс. На рисунке 4 представлена логическая схема данных

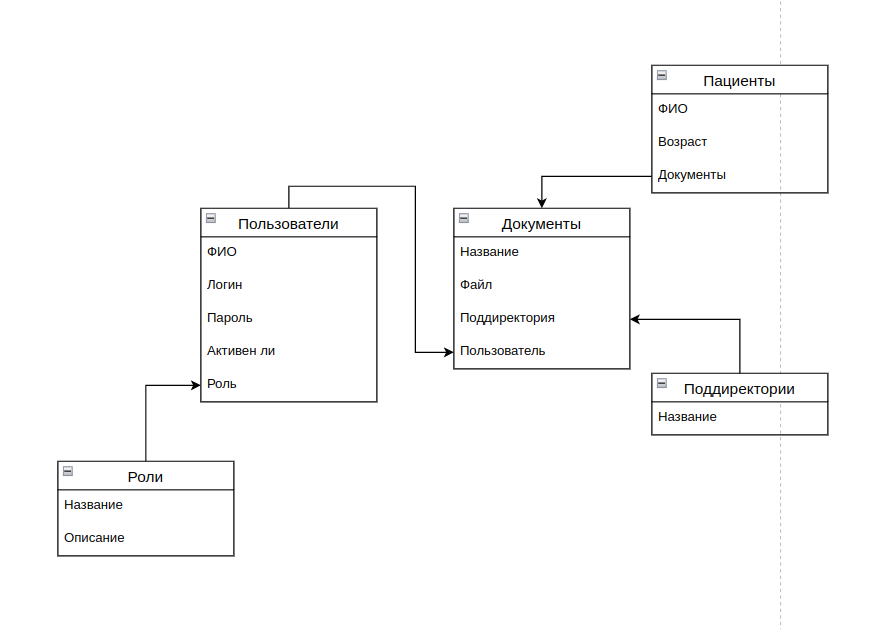


Рисунок 4 - Логическая схема

База данных состоит из 11 сущностей (таблиц), которые имеют определённый набор атрибутов (полей) для записи информации. Каждая таблица имеет своё уникальное поле идентификатор, по которым формируются связи между таблицами.

При проектировании базы данных была составлена физическая схема данных, которая отображает типы данных атрибутов каждой сущности.

На Рисунке 5 представлена физическая схема данных.

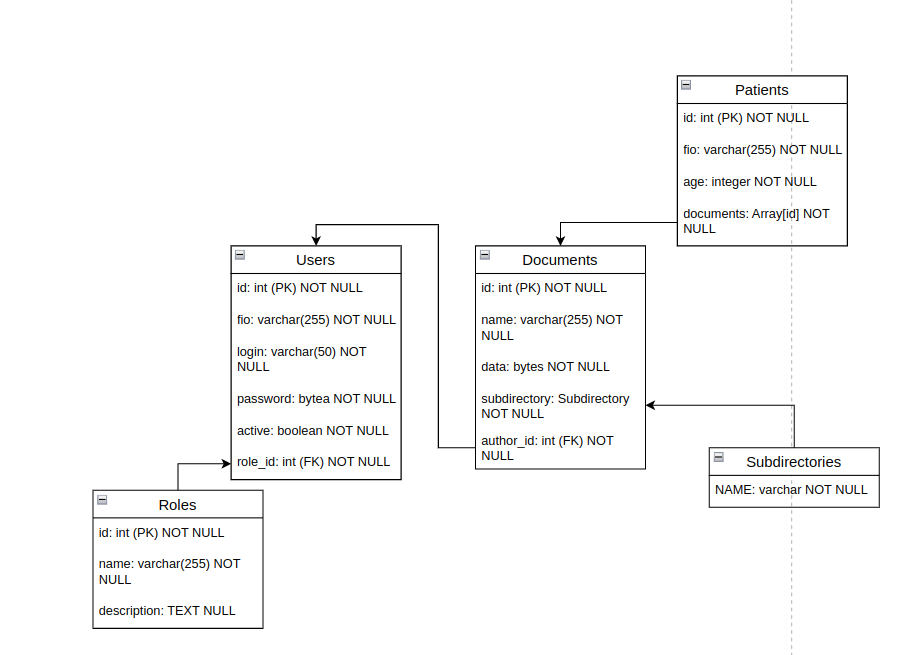


Рисунок 5 - Физическая схема

В приложении использовалась система управления реляционными базами данных Pg Admin4 , связи между сущностями были реализованы посредством SQLAlchemy ORM в связке с Pydantic. База данных состоит из 5 таблиц, в которых реализованы связи «Один ко многим», «Многие ко многим», «Один к одному». В Таблице 6 представлен словарь данных для разработанной базы данных, с которой взаимодействует приложение.

Таблица 6 - Словарь данных

| Ключ | Наименование | Тип данных | Обязательность заполнения | Назначение |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| Users – таблица с пользователями | | | | |
| PK | Id | Int | Not null | Уникальный идентификатор с  атрибутом Auto increment |
|  | Login | Varchar (50) | Not null | Логин |
|  | password | Varchar (64) | Not null | пароль |
|  | uuid | uuid | Not null | универсальный уникальный идентификатор |
|  | address | Varchar(60) | Not null | Адрес кошелька |
| FK | role | Id | Not null | Роль |
| Roles – таблица с ролями | | | | |
| Ключ | Наименование | Тип данных | Обязательность заполнения | Назначение |
| 1 | 2 | 3 | 4 | 5 |
| PK | Id | Int | Not null | Уникальный идентификатор с  атрибутом Auto increment |
|  | Name | Varchar (255) | Not null | Название |
| Transactions – таблица с транзакциями | | | | |
| PK | Id | Int | Not null | Уникальный идентификатор с  атрибутом Auto increment |
|  | Tx\_hash | Varchar (64) | Not null | Хэш |
|  | Tx\_date | timestamp | Not null | Дата |
|  | Owner\_address | Varchar(60) | Not null | Адрес владельца |
|  | Receiver\_address | Varchar(60) | Not null | Адрес получателя |
|  | Smart\_contract\_id | Int | Not null | Смарт контракт |
|  | Product\_id | Int | Not null | Товар |
|  | type | Varchar(255) | Not null | Тип |
| Smart Contracts – таблица с смарт контрактами | | | | |
| PK | id | Int | Not null | Уникальный идентификатор с  атрибутом Auto increment |
|  | onchain\_address | Varchar(60) | Not null | Ончейн адрес |
|  | ABI | JSON | null | ABI |
|  | Owner\_address | Varchar(60) | Not null | Адрес владельца |
|  | status | Varchar(100) | null | Статус |
|  | Created\_at | timestamp | Not null | Дата создания |
|  | uuid | uuid | Not null | Универсальный уникальный идентификатор |
| Products – таблица с товарами | | | | |
| PK | id | Int | Not null | Уникальный идентификатор с  атрибутом Auto increment |
|  | Name | Varchar(255) | Not null | Название |
|  | Description | Varchar(1000) | Not null | Описание |
|  | Onchain\_address | Varchar(60) | Not null | Ончейн адрес |
|  | Owner\_address | Varchar(60) | Not null | Адрес владельца |
|  | metadata | text | null | Метадата |
| FK | Type\_id | Int | Not null | Тип продукта |
| Product Types – таблица с типами продуктов | | | | |
| PK | id | Int | Not null | Уникальный идентификатор с  атрибутом Auto increment |
|  | name | Varchar(255) | Not null | Название |
| Logs – таблица с логами | | | | |
| PK | id | Int | Not null | Уникальный идентификатор с  атрибутом Auto increment |
|  | Interaction\_type | Varchar(255) | Not null | Тип взаимодействия |
|  | Created\_at | timestamp | Not null | Дата создания |
|  | Log\_type | Varchar(255) | Not null | Тип |
|  | name | Varchar(1000) | Not null | Название |
| UserLogs – таблицами с логами пользователей | | | | |
| PK | id | Int | Not null | Уникальный идентификатор с  атрибутом Auto increment |
|  | uuid | uuid | Not null | Универсальный уникальный идентификатор |
| FK | User\_id | int | Not null | Пользователь |
| FK | Log\_id | int | Not null | Лог |
| UserTransactions – таблица с транзакциями пользователей | | | | |
| PK | id | Int | Not null | Уникальный идентификатор с  атрибутом Auto increment |
|  | uuid | uuid | Not null | Универсальный уникальный идентификатор |
| FK | User\_id | int | Not null | Пользователь |
| FK | Tx\_id | int | Not null | Транзакция |
| Requests – Таблица с запросами | | | | |
| PK | id | Int | Not null | Уникальный идентификатор с  атрибутом Auto increment |
|  | created\_at | timestamp | Null | Дата создания |
|  | description | TEXT | Not null | Описание |
|  | name | Varchar(255) | Not null | Название |
|  | email | Varchar(255) | Not null | Электронная почта |
| UserRequests – Таблица с запросами пользователей | | | | |
| PK | id | Int | Not null | Уникальный идентификатор с  атрибутом Auto increment |
|  | uuid | uuid | Not null | Универсальный уникальный идентификатор |
| FK | User\_id | int | Not null | Пользователь |
| FK | request\_id | int | Not null | Запрос |

### 2.3.5 Схема пользовательского интерфейса

На Рисунке 6 продемонстрированы схемы интерфейсов веб-приложения. На данной схеме графически продемонстрированы все страницы, а также переходы между ними.

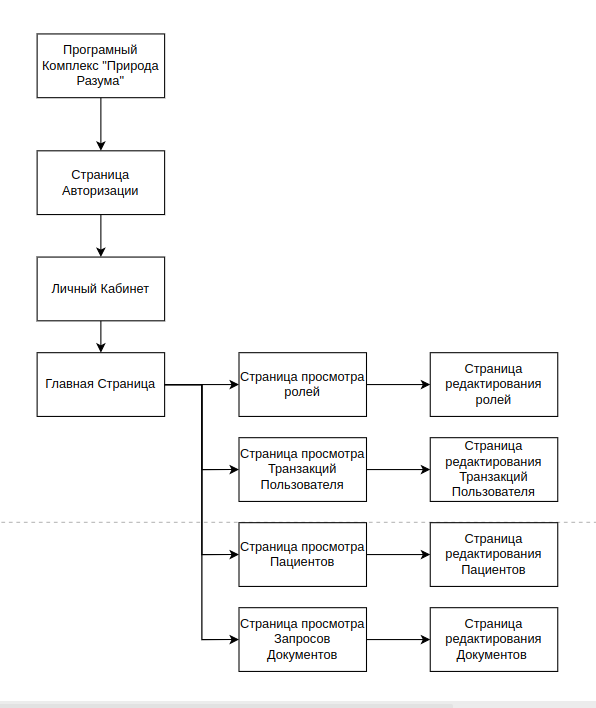


Рисунок 6 - Схема интерфейса

Также представим схему интерфейса всего программного комплекса

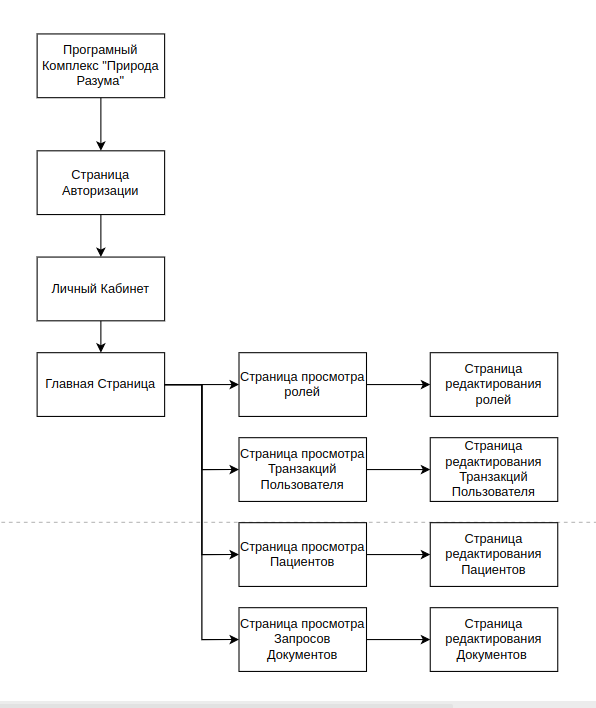


Рисунок 7 - Схема Интерфейса

# 2.4. Результаты работы программы

В результате выполнения поставленной задачи, было разработано требуемое программное решение, которое является программным комплексом центра развития “Природа Разума”. Данное программное решение является клиент-серверным и состоит из:

• Реляционной базы данных;

• Веб - приложения;

• API, которое позволяет получать данные базы данных;

• Мобильного приложения;

На рисунках 9-18 представлены результаты работы разработанного веб-приложения, а именно: Страница Авторизации, Страница Личного Кабинета, Страница Документов, Страница Редактирования Документа, Страница Пациентов, Страница Редактирования Пациента, Страница Пользователей, Страница Редактирования Пользователя, Страница Ролей, Страница Редактирования Роли



Рисунок 8 - Страница Авторизации

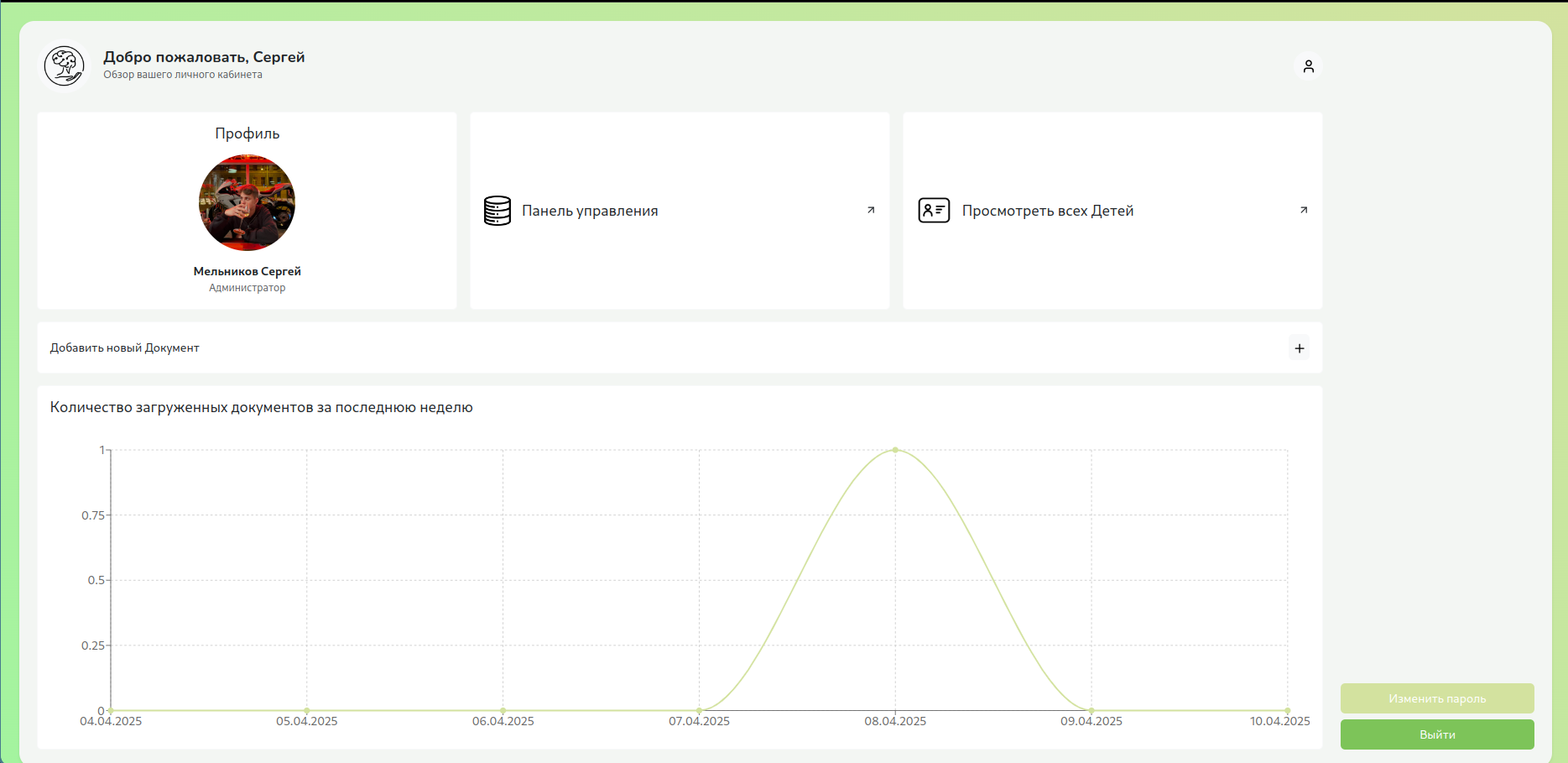


Рисунок 9 - Страница Личного Кабинета

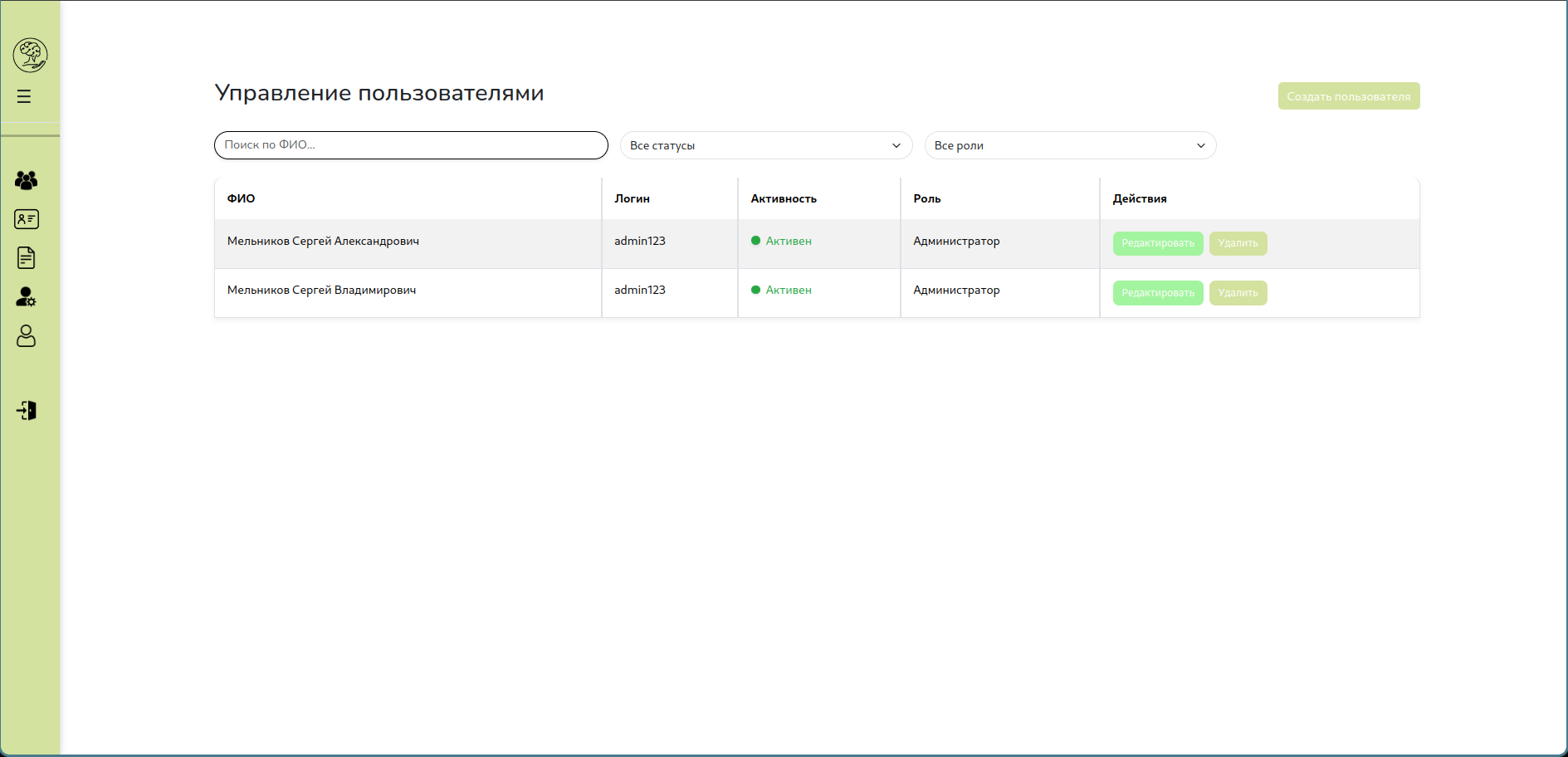


Рисунок 10 - Страница Пользователи

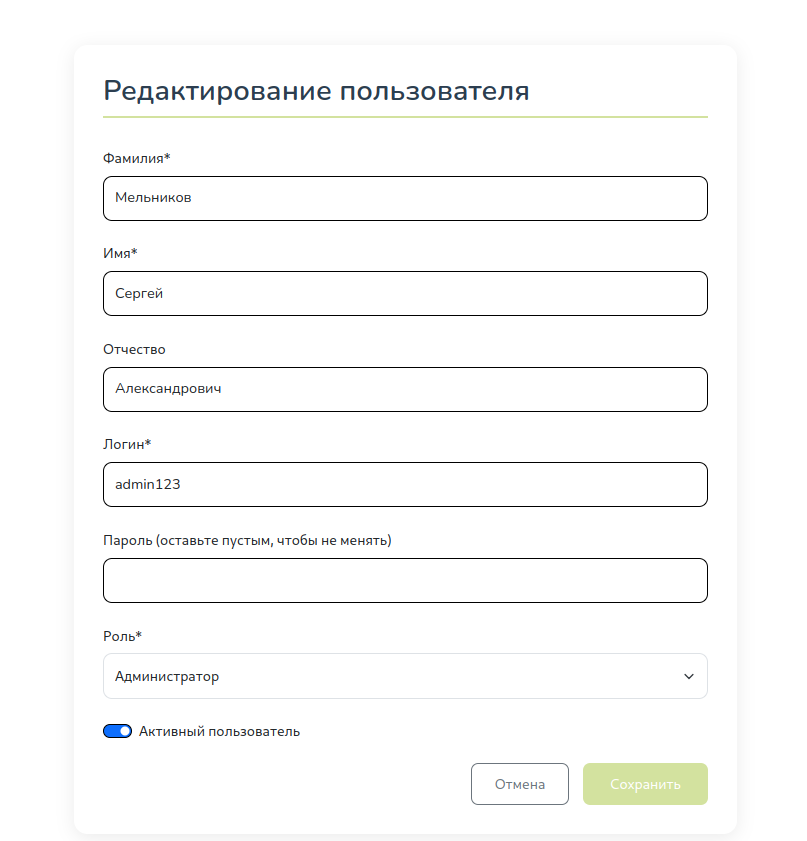


Рисунок 11 - Страница Редактирования Пользователя

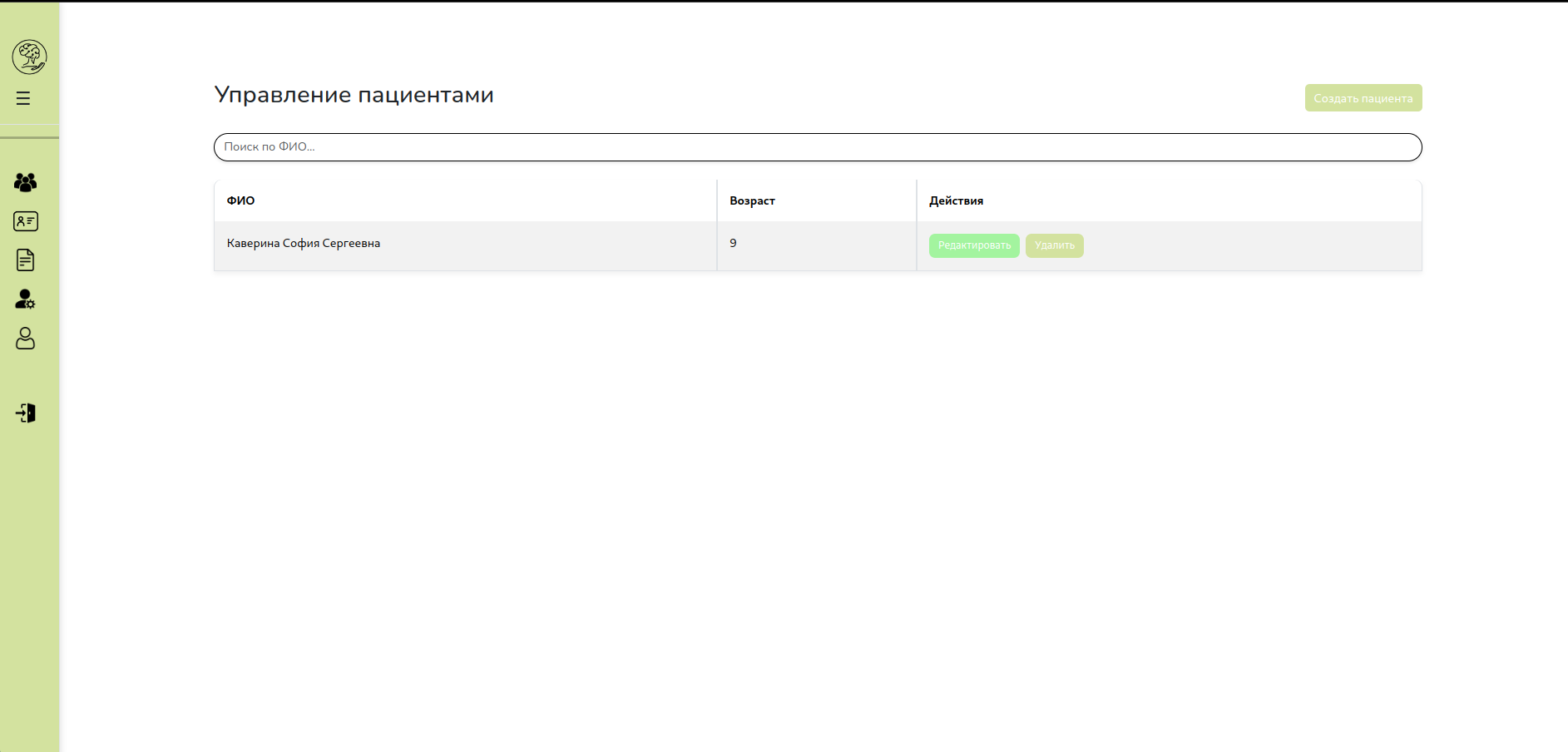


Рисунок 12 - Страница Пациентов

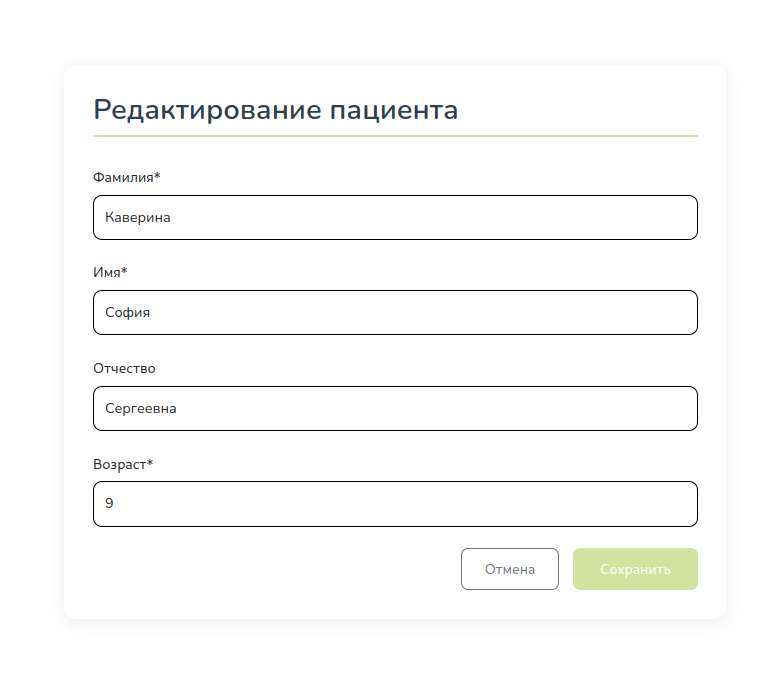


Рисунок 13 - Страница Редактирования Пациента

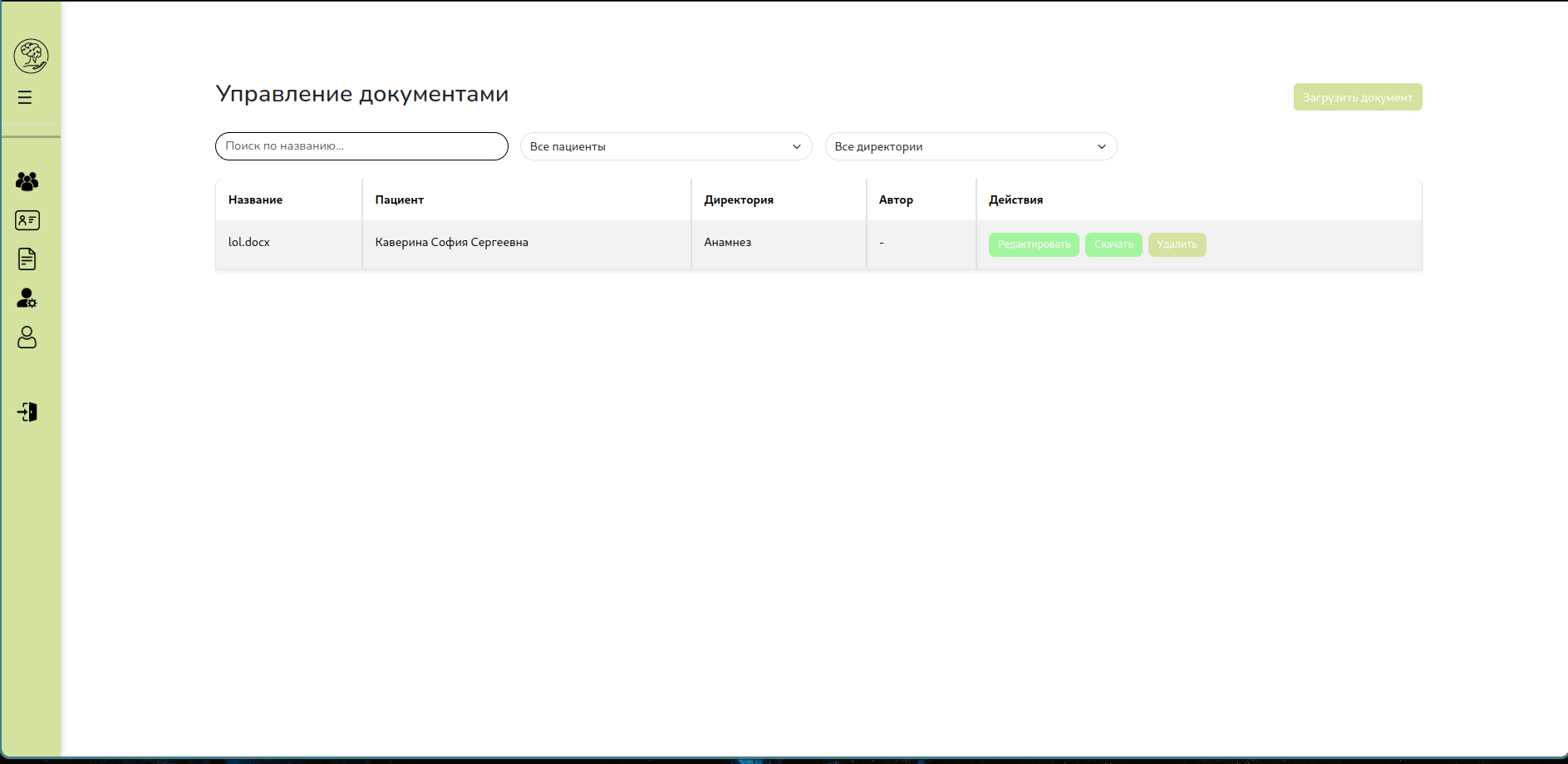


Рисунок 14 - Страница Документов

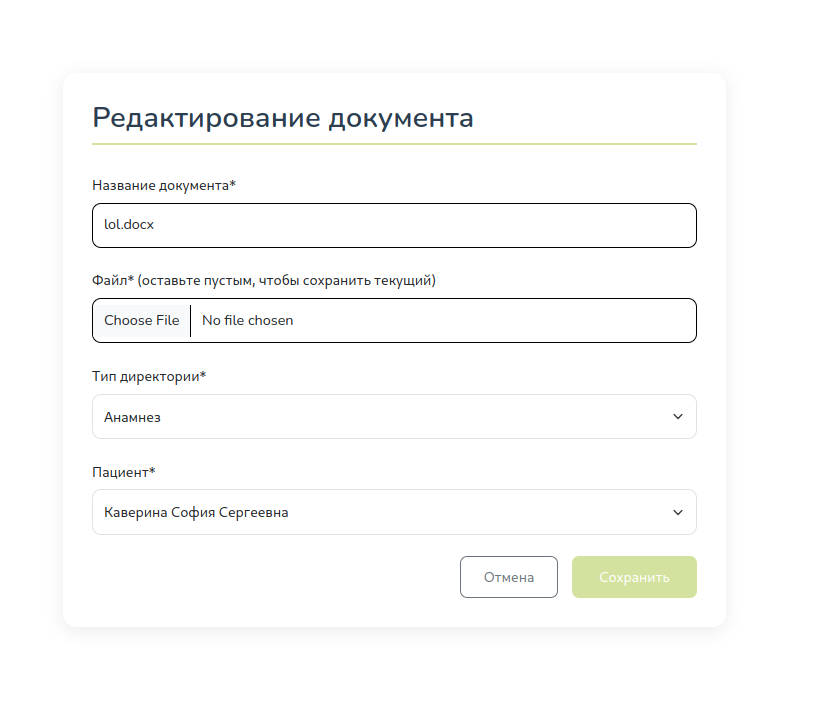


Рисунок 15 - Страница Редактирования Документа

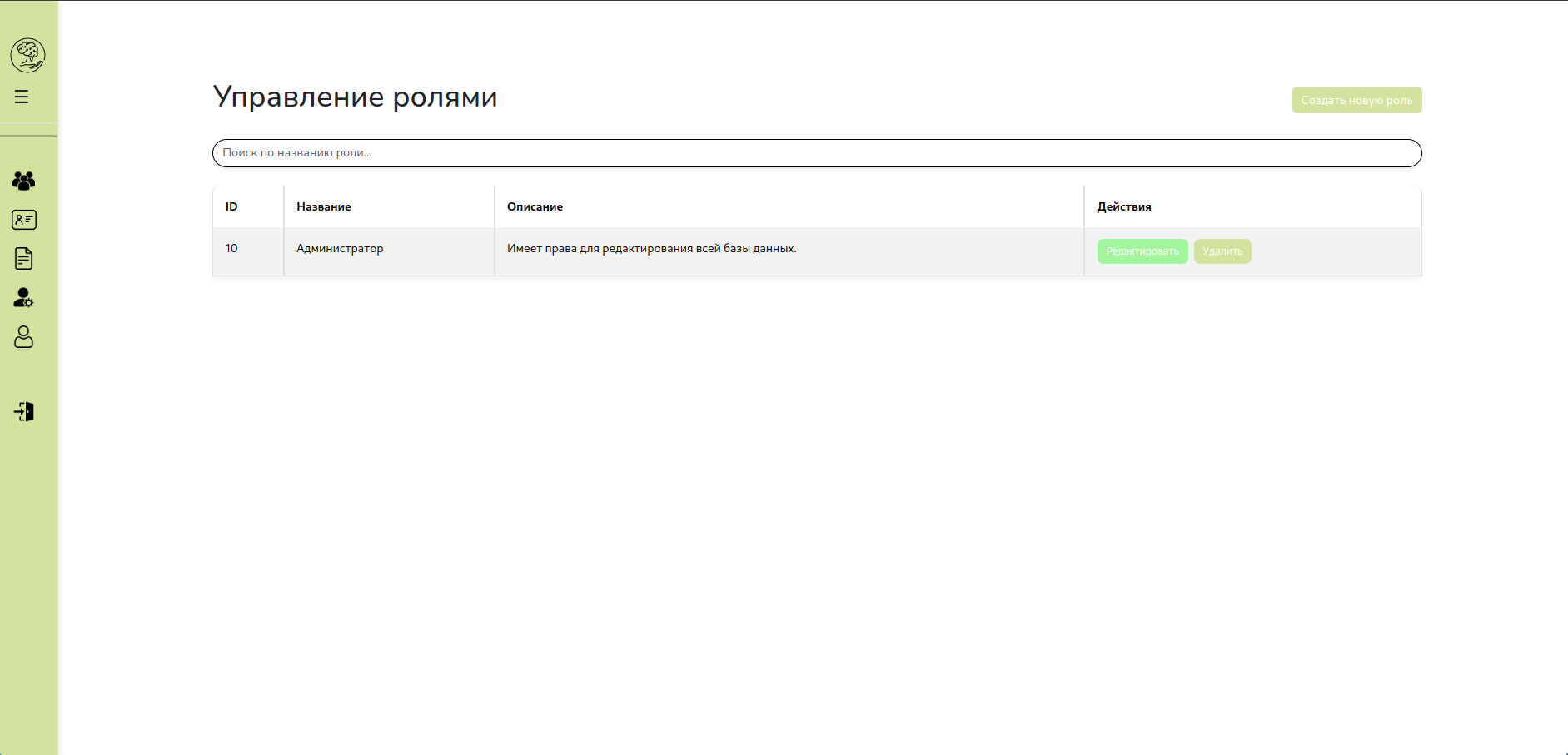


Рисунок 16 - Страница ролей

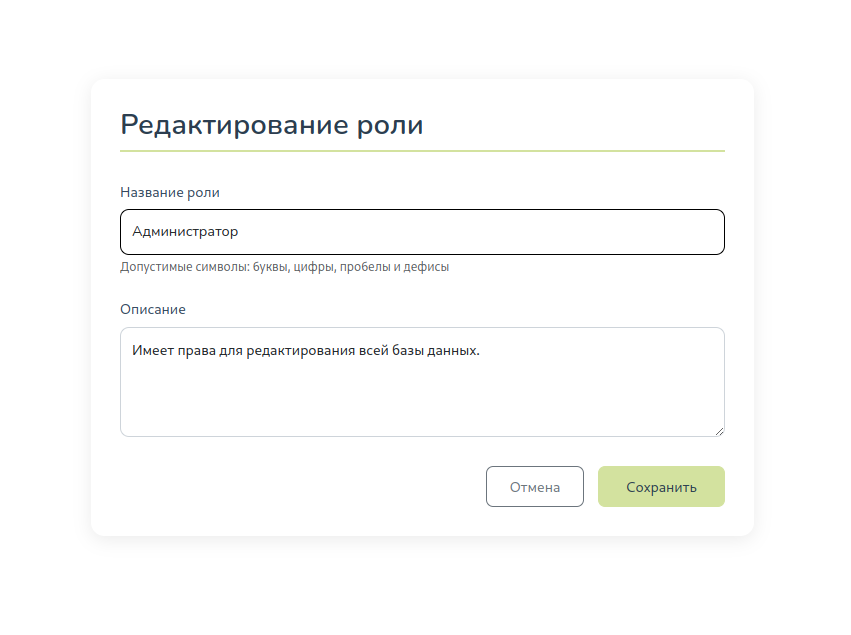


Рисунок 17 - Страница редактирования роли

# ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ

## 3.1. Инструментальные средства разработки

Для разработки программного комплекса были использованы следующие языки программировния: Python и Typescript совместно с фреймворкоми FastAPI и React для создания бэкэнда и фронтэнда приложения.

Для реализации интерфейса на стороне клиента применялся HTML, CSS, а также специальные файлы tsx, для реализации верстки на frontend. Данные технологии позволили сделать красивый и очень удобный интерфейс для пользователяю

В качестве системы управления базами данных использовался Pg Admin 4. Для построения и взаимодействия с базой данных применялся Django ORM и SQLAlchemy ORM. Взаимодействие с базой данных происходило через модели, которые были разработаны с учетом требований третьей нормальной формы.   
Для работы с API на стороне клиента использовалась библиотека ApolloClient, для реализации самого API на стороне бэкэнда использовалась библиотека Strawberry. Для реализации авторизации при помощи JWT токена использовалась библиотека jwt

Проект был разработан в среде Visual Studio Code, которая обеспечивает удобность и гибкость разработки для множества фреймворков и технологий.

# 3.2. Отладка программы

Для отладки программы использовались:

Python Debugger для дебагга приложения, а также точки остановы для отслеживания артефактов и ошибок в программном коде.

В ходе разработки возникла ошибка при попытке переходе на страницу “Пользователи”

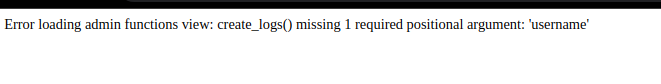


Рисунок 18 - ошибка при получении данных из БД

Решение заключалось в том, что было необходимо вставить недостающий аргумент в функцию для создания Лога.

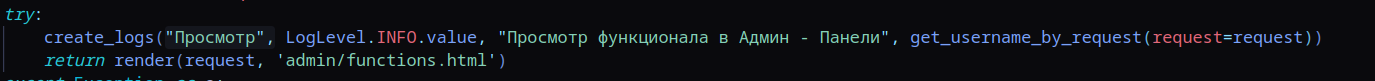


Рисунок 19 - Добавление недостающего аргумента в функцию

# 3.3. Защитное программирование

Защитное программирование применялось на этапе проектирования приложения, были созданы проверки на корректность введенных данных. В моделях были реализованы валидаторы для проверки правильности ввода данных.

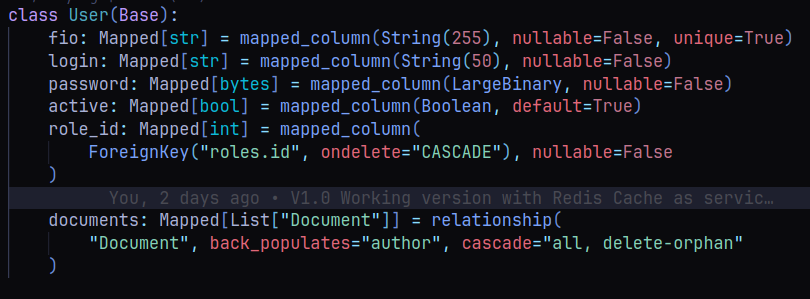


Рисунок 20 - Валидация

# 3.4. Характеристики программы

Данный программный комплекс состоит из FrontEnd и BackEnd, а также API. API необходимо для реализации связи между базрй данных и FrontEnd частью программного комлекса. Главным критерием для запуска данного веб – приложения является нахождения приложения в одной локальной сети с компьютером.

Таблица 7 - Модули

| Модуль | Описание | Количество строк кода | Размер (в Кбайтах) |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| src/app.py | В данном модуле представлена основная точка входа в приложение FastAPI. Здесь происходит инициализация сервера, регистрация всех маршрутов и настройка промежуточного ПО. | 76 | 159 |
| src/config.py | В данном модуле реализованы настройки приложения, загрузка конфигурационных переменных из окружения и установка логгера. | 43 | 222 |
| src/depends.py | В данном модуле представлены зависимости FastAPI для внедрения сервисов, репозиториев и других компонентов в обработчики запросов. | 35 | 69 |
| src/auth/auth.py | В данном модуле реализована система аутентификации и авторизации, включая работу с JWT-токенами и проверку прав доступа. | 87 | 10 |
| src/auth/schema.py | В данном модуле представлены схемы Pydantic для аутентификации и авторизации, определяющие формат запросов и ответов для этих операций. | 209 | 16 |
| src/auth/utils.py | В данном модуле реализованы вспомогательные функции для работы с аутентификацией, такие как хеширование паролей и проверка токенов. | 254 | 19 |
| src/cache/utils.py | В данном модуле представлены утилиты для работы с кешированием, включая кодировщик Base64 для сериализации/десериализации кешированных данных. | 187 | 18 |
| src/db/db.py | В данном модуле реализована конфигурация и подключение к базе данных, создание пула соединений и управление транзакциями. | 287 | 16 |
| src/migration/env.py | В данном модуле представлена среда для выполнения миграций Alembic, определяющая как генерировать и применять миграции к базе данных. | 21 | 60 |
| src/migration/script.py.mako | В данном модуле реализован шаблон Mako для генерации скриптов миграции Alembic. | 183 | 7 |
| src/models/models.py | В данном модуле представлены модели SQLAlchemy для всех таблиц базы данных, определяющие схему данных. | 39 | 7 |
| src/repositories/base.py | В данном модуле реализован базовый репозиторий с общими методами CRUD для работы с различными сущностями. | 56 | 6 |
| src/repositories/documents.py | В данном модуле представлен репозиторий для работы с документами, наследующий от базового репозитория с расширенной функциональностью. | 501 | 73 |
| src/repositories/patients.py | В данном модуле реализован репозиторий для работы с пациентами, включающий специализированные методы для этой сущности. | 103 | 154 |
| src/repositories/roles.py | В данном модуле представлен репозиторий для работы с ролями пользователей, управляющий правами доступа в системе. | 94 | 32 |
| src/repositories/users.py | В данном модуле реализован репозиторий для работы с пользователями системы, включая методы аутентификации. | 21 | 20 |
| src/routing/analitics.py | В данном модуле представлены маршруты API для аналитических запросов и отчетов. | 43 | 50 |
| src/routing/base.py | В данном модуле реализована функция создания базовых маршрутов для CRUD-операций над сущностями. | 32 | 23 |
| src/routing/documents.py | В данном модуле представлены маршруты API для работы с документами, включая загрузку и скачивание файлов. | 27 | 53 |
| src/routing/patients.py | В данном модуле реализованы маршруты API для управления данными пациентов. | 16 | 154 |
| src/routing/roles.py | В данном модуле представлены маршруты API для управления ролями и правами доступа. | 35 | 47 |
| src/routing/users.py | В данном модуле реализованы маршруты API для управления пользователями системы. | 108 | 67 |
| src/routing/utils.py | В данном модуле представлены вспомогательные функции для маршрутизации и обработки запросов. | 74 | 45 |
| src/schemas/analitics.py | В данном модуле реализованы схемы Pydantic для аналитических данных и отчетов. | 60 | 32 |
| src/schemas/documents.py | В данном модуле представлены схемы Pydantic для валидации и сериализации данных документов. | 112 | 71 |
| src/schemas/patients.py | В данном модуле реализованы схемы Pydantic для валидации и сериализации данных пациентов. | 59 | 43 |
| src/schemas/roles.py | В данном модуле представлены схемы Pydantic для валидации и сериализации данных ролей. | 67 | 54 |
| src/schemas/users.py | В данном модуле реализованы схемы Pydantic для валидации и сериализации данных пользователей. | 49 | 79 |
| src/services/base.py | В данном модуле представлен базовый сервисный слой с бизнес-логикой для работы с сущностями. | 45 | 13 |
| src/services/documents.py | В данном модуле реализована бизнес-логика для работы с документами, включая валидацию и преобразование данных. | 87 | 21 |
| src/services/patients.py | В данном модуле представлена бизнес-логика для работы с пациентами. | 41 | 11 |
| src/services/roles.py | В данном модуле реализована бизнес-логика для работы с ролями и правами доступа. | 101 | 54 |
| src/services/users.py | В данном модуле представлена бизнес-логика для работы с пользователями, включая регистрацию и управление аккаунтами. | 48 | 34 |
| App.tsx | В данном модуле представлен корневой компонент приложения, отвечающий за маршрутизацию и общую структуру интерфейса. | 76 | 159 |
| main.tsx | В данном модуле реализована точка входа в приложение, где происходит рендеринг корневого компонента в DOM. | 43 | 222 |
| features/auth/components/LoginForm.tsx | В данном модуле реализована форма авторизации пользователей с полями для ввода учетных данных. | 35 | 69 |
| features/auth/services/api.ts | В данном модуле представлены функции для взаимодействия с API аутентификации, включая запросы на вход в систему. | 87 | 10 |
| features/auth/services/utils.ts | В данном модуле реализованы вспомогательные утилиты для процессов аутентификации и авторизации. | 209 | 16 |
| features/auth/styles/index.css | В данном модуле представлены стили для компонентов аутентификации, определяющие их внешний вид. | 254 | 19 |
| features/auth/types/auth.types.ts | В данном модуле реализованы типы данных, связанные с аутентификацией пользователей. | 187 | 18 |
| features/panel/components/DocumentChart.tsx | В данном модуле представлен компонент для визуализации данных о документах в виде графика. | 287 | 16 |
| features/panel/components/PasswordChangeModal.tsx | В данном модуле реализовано модальное окно для изменения пароля пользователя. | 21 | 60 |
| features/panel/components/PersonalDataModal.tsx | В данном модуле представлено модальное окно для редактирования личных данных пользователя. | 183 | 7 |
| features/panel/components/SideBar.tsx | В данном модуле реализована боковая панель навигации административной части приложения. | 39 | 7 |
| features/panel/MainPage.tsx | В данном модуле представлена главная страница панели управления с общей информацией и статистикой. | 56 | 6 |
| features/panel/PersonalCabinet.tsx | В данном модуле реализован личный кабинет пользователя с его персональными данными и настройками. | 501 | 73 |
| features/panel/models/documents/DocumentForm.tsx | В данном модуле представлена форма для создания и редактирования документов. | 103 | 154 |
| features/panel/models/documents/DocumentList.tsx | В данном модуле реализован список документов с возможностями сортировки и фильтрации. | 94 | 32 |
| features/panel/models/documents/types/documents.types.ts | В данном модуле представлены типы данных, связанные с документами в системе. | 21 | 20 |
| features/panel/models/patients/PatientForm.tsx | В данном модуле реализована форма для добавления и редактирования данных пациентов. | 43 | 50 |
| features/panel/models/patients/PatientList.tsx | В данном модуле представлен список пациентов с инструментами управления и поиска. | 32 | 23 |
| features/panel/models/patients/types/patients.types.ts | В данном модуле реализованы типы данных для работы с информацией о пациентах. | 27 | 53 |
| features/panel/models/roles/RoleForm.tsx | В данном модуле представлена форма для создания и изменения ролей в системе. | 16 | 154 |
| features/panel/models/roles/RoleList.tsx | В данном модуле реализован компонент для отображения и управления списком ролей пользователей. | 35 | 47 |
| features/panel/models/roles/types/roles.types.ts | В данном модуле представлены типы данных, описывающие роли и права доступа. | 108 | 67 |
| features/panel/models/users/UserForm.tsx | В данном модуле реализована форма для создания и редактирования пользователей системы. | 74 | 45 |
| features/panel/models/users/UserList.tsx | В данном модуле представлен список пользователей с возможностями управления их учетными записями. | 60 | 32 |
| features/panel/models/users/types/users.types.ts | В данном модуле реализованы типы данных для работы с пользователями системы. | 112 | 71 |
| features/panel/styles/cabinet.css | В данном модуле представлены стили для компонентов личного кабинета пользователя. | 59 | 43 |
| features/panel/styles/index.css | В данном модуле реализованы общие стили для компонентов панели управления. | 67 | 54 |
| features/panel/types/panel.types.ts | В данном модуле представлены типы данных, используемые компонентами панели управления. | 49 | 79 |
| features/services/api.ts | В данном модуле реализованы общие функции для взаимодействия с API всего приложения. | 45 | 13 |
| MainActivity.kt | В данном модуле реализован весь функционал и UI – изайн для мобильного приложения | 356 | 21 |

# ЗАКЛЮЧЕНИЕ

В результате выполнения проекта был разработан программный продукт «Природа разума», который отвечает всем предъявленным требованиям в области нейроинтеграции и анализа данных. В ходе работы была проведена тщательная экспертиза и анализ предметной области, связанной с управлением данными и процессами в образовательной среде. Были выявлены основные проблемы и особенности, которые необходимо учитывать при разработке решения. Ключевым аспектом данной системы стала автоматизация процессов анализа данных и формирования рекомендаций через веб-приложение, а также улучшение и упрощение пользовательского опыта для педагогов и администраторов.

Для решения поставленной задачи было разработано веб-приложение, предназначенное для управления процессами в образовательной среде. В основе системы лежит реляционная база данных, включающая набор сущностей и взаимосвязей между ними. База данных предназначена для хранения информации о пользователях, их ролях, заключениях, рекомендациях, упражнениях взаимодействия.

Веб-приложение взаимодействует с базой данных посредством REST API и предоставляет удобный интерфейс для просмотра, управления и анализа информации. Пользователи системы могут легко работать с данными благодаря интуитивно понятному интерфейсу и функциональным возможностям, доступным как педагогам, так и администраторам.

Разработка программного продукта включала выполнение всех этапов жизненного цикла: от анализа требований до тестирования и оценки результатов. На этапе анализа были определены ключевые функции и требования, которые должен удовлетворять программный комплекс. Также были выбраны технологии и методы программирования, которые использовались для реализации системы. Были проанализированы потенциальные проблемы и предложены способы их решения.

Проектирование включало создание функциональных, интерфейсных, логических и физических схем, отражающих структуру взаимодействия классов и функциональность системы. При разработке веб-приложения использовался паттерн Clear Architecture, который обеспечил разделение логики, интерфейса и управления данными. Это позволило повысить производительность системы при работе с внешними модулями и улучшить запись данных в базу.

На этапе тестирования были определены методы и схемы тестирования, составлены сценарии тестов и проведено модульное тестирование системы для проверки её работоспособности.

По итогам работы было сделано заключение о том, что разработанный продукт обладает рядом преимуществ: удобный интерфейс для педагогов, автоматизация обработки данных и рекомендаций, а также удобство управления и анализа информации.

Несмотря на успешную разработку, программный продукт имеет потенциал для дальнейшего развития. Возможности для улучшений включают расширение базы данных, внедрение дополнительных модулей и новых функций.

СПИСОК ИСПОЛЬЗУЕМЫХ МАТЕРИАЛОВ

3.1. Алекс Бэнкс, Ева Порселло. *React и Redux. Функциональная веб-разработка.* 2-е издание. Питер СПб, 2020, 304с.  
 3.2. Дэн Вандермейр. *TypeScript в действии.* Manning Publications, 2021, 360с.  
 3.3. Борис Черны. *TypeScript. Полное руководство.* Питер СПб, 2020, 408с.  
 3.4. Себастьян Рамирес. *FastAPI для профессионалов.* Официальная документация, 2023, 280с.  
 3.5. Кэролайн де Мартин. *Полный справочник по React.* O'Reilly Media, 2019, 416с.  
 3.6. Михаил Кузнецов. *Создание REST API с FastAPI.* Издательство Питер, 2021, 320с.  
 3.7. Официальный сайт React [Электронный ресурс] URL: <https://reactjs.org/> (Дата обращения 02.11.2024).  
 3.8. Официальный сайт TypeScript [Электронный ресурс] URL: <https://www.typescriptlang.org/> (Дата обращения 06.11.2024).  
 3.9. Официальный сайт FastAPI [Электронный ресурс] URL: https://fastapi.tiangolo.com/ (Дата обращения 02.11.2024).

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

«Российский экономический университет им. Г.В. Плеханова»

Московский приборостроительный техникум

Проверка на уникальность авторского текста

курсового проекта

| На тему: | Разработка программного комплекса «Природа разума» |
| --- | --- |

МЕЛЬНИКОВА СЕРГЕЯ АЛЕКСАНДРОВИЧА

Студента 4 курса группы П50-6-21

по специальности 09.02.07 «Информационные системы и программирование»

квалификация: Программист

Уникальность текста проверялась на сайте www.antiplagiat.ru. Процент оригинальности содержимого пояснительной записки составил 89,91%. Результат представлен на Рисунке 1.

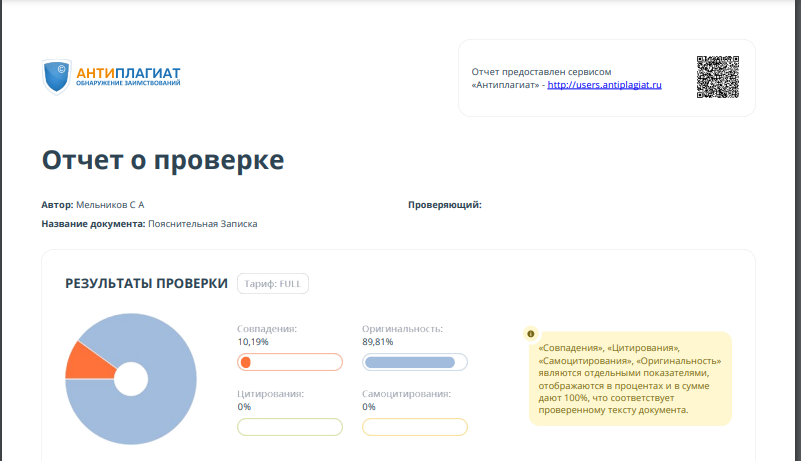


Рисунок 1 - Проверка на уникальность текста пояснительной записки