**Национальный исследовательский ядерный университет «МИФИ»**

|  |  |
| --- | --- |
|  | **Институт интеллектуальных и кибернетических систем**  **Кафедра «Компьютерные системы и технологии»** |

Пояснительная записка

к проекту по курсу «Программирование

сетевых приложений»

на тему «Система аннотирования изображений»

Студент гр. Б20-523 Шпачков Е. А. /\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /

Руководитель Овчаренко Е. С. / \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /

Москва, 2023г.

# **Введение**

Специалистам в области изучения раковых опухолей необходимо проводить анализ клеток опухолей на основе их изображений, маркировки этих изображений и заносить промаркированные изображения в уже реализованную модель ML с последующим вынесением диагноза/результата обследования.

Вследствие этого появилась необходимость в создании веб приложения, которое позволит медикам через интерфейс заносить данные исследований в единое хранилище данных, проводить марикровку изображений и вести историю болезни, что позволит заметно повысить качество анализа опухолей. А так же необходимость в функционале администратора, с помощью которого можно будет управлять всей системой.

# **Теоретическая часть**

* 1. **Краткий анализ существующих решений**

В настоящее время аналогичных проектов не существует, т.к. разработка заточена под автоматизацию работы научных сотрудников в области медицины, поэтому анализ важных функций проводился на основе требований, выдвигаемых заказчиком:

1. возможность просмотра/редактирования информации о пациентах, опухолях, клетках и остальных сущностях, рассматриваемых в рамках исследования;
2. удобство интерфейса пользователя и администратора, интуитивность;
3. безопасность веб-приложения (защита от некоторых атак на инфраструктуру);
4. управление системой через соответствущие параметры, регулируемые администартором
   1. **Постановка задачи**

Исходя из полученных результатов были выделены следующие требования к приложению:

* возможность добавлять изображения клеток и их маркировок на эти изображения;
* возможность получать информацию о пациентах и справочниках, требуемых специалистам (справочник с терминами);
* наличие админ-панели;
* В качестве клиентской части должно выступать веб-приложение

Кроме вышеперечисленных функций в приложении также должна быть реализована аутентификация и авторизация пользователя. Также должно вестись логирование всех действий, совершаемых пользователем.

* 1. **Схема приложения**

Клиент-серверное приложение состоит из нескольких связанных частей. Схема взаимодействия между частями приложения представлена на рисунке 1.



Рисунок 1. Схема взаимодействия между частями приложения

Пользователь через веб-приложение может совершать действия, которые будут обрабатываться сервером и сохраняться в базе данных.

* 1. **Формализация данных**

База данных должна хранить данные о следующих сущностях, представленных в таблице 1.

Таблица 1. Сущности базы данных

|  |  |
| --- | --- |
| **Сущности** | **Сведения** |
| Пользователь | Логин, пароль, имя, информация о пользователе |
| Тип пользователя | Название категории пользователя |
| Пациент | Группа, имя, номер истории болезни |
| История болезни | Номер истории болезни, дата начала ведения истории болезни |
| Клетки | Изображение, масштаб изображения, параметры scd2 |
| Мареры | Название маркера, описание |
| Лог | Название, описание, тип лога, статус выполнения операции |
| Термины | Название термина, определение |
| Данные имунофенотерапии | Дата, описание |
| Результат исследования | Текст результата исследования, пациент |

1. **Практическая часть**
   1. **Используемые технологии**

В качестве языка программирования для разработки серверной части приложения был выбран язык программирования Python, поскольку он поддерживает парадигму объектно-ориентированного программирования, а также поддерживает возможность работы с фреймворками, такими как Django и Django Rest Framework. Данный фреймворк упрощает разработку, используя большое количество готовых пресетов, позволяющих упростить разработку рутинных элементов реализации, а так же ясную не перегруженную архитектуру..

В качестве базы данных используется PostgreSQL, как самая распространённая, надежная и быстрая СУБД с открытым исходным кодом.

Для разработки пользовательского интерфейса было принято решение использовать стандартный стек HTML, CSS, JS. Была использована бибилотека готовых стилей Bootstrap.

* 1. **Описание таблиц базы данных**

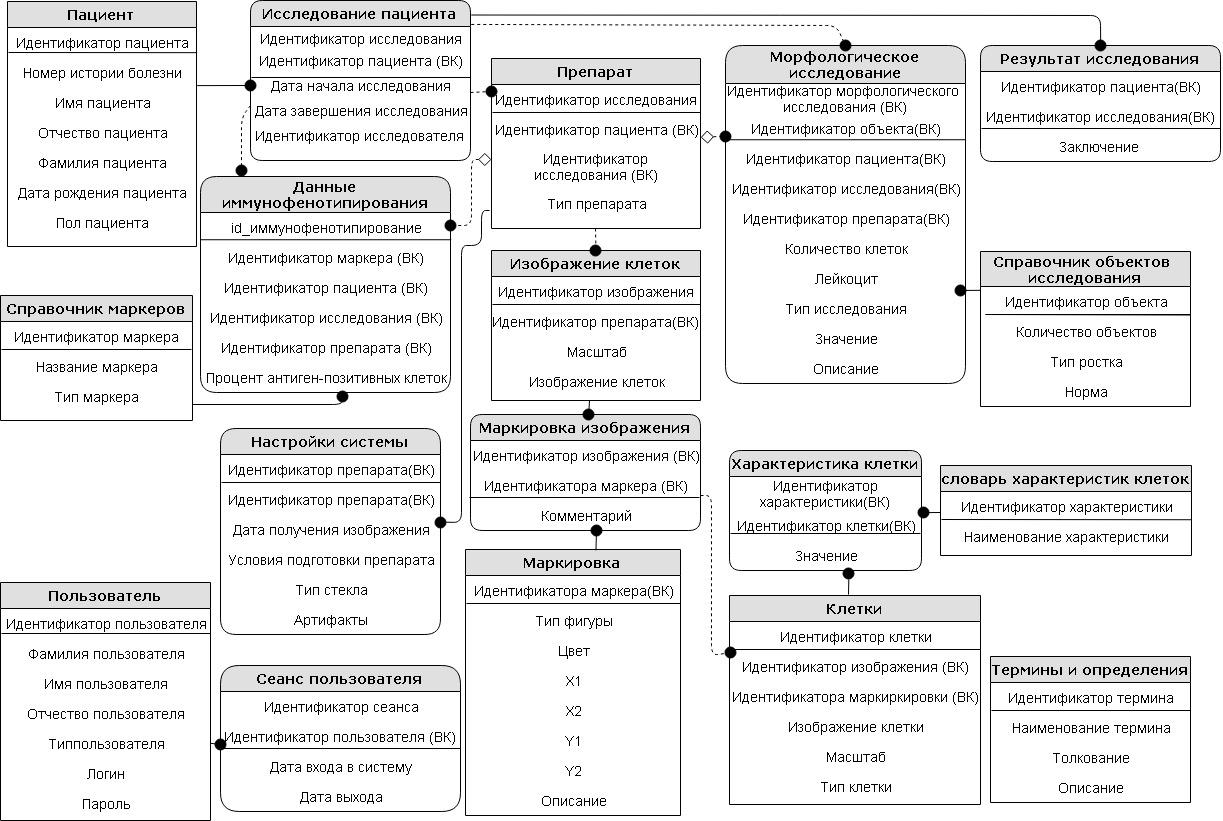
Чтобы организовать хранение информации о пользователях, группах и питомцах используется база данных PostgreSQL. Структура базы данных представлена ниже. 

Рисунок 2. Структура базы данных

* 1. **Уровень бизнес-логики**

После того, как появилась возможность взаимодействовать с базой данных посредствам Django ORM, можно приступать к уровню бизнес-логики. На данном этапе будут проводится все вычисления, которые нужны для функционирования самого сервиса. От создания пользователя до обновления информации о новой версии изображенияь клетки.

* 1. **Уровень взаимодействия**

Чтобы пользователи сети посылали запрос на сервис, нужны конечные точки, по запросу на которые сервис будет отдавать ответ. Для этой цели были созданы контроллеры, которые управляют запросами приходящие на сервер.

На рисунке ниже представлены контроллеры, обрабатывающие все входящие запросы на сервер:

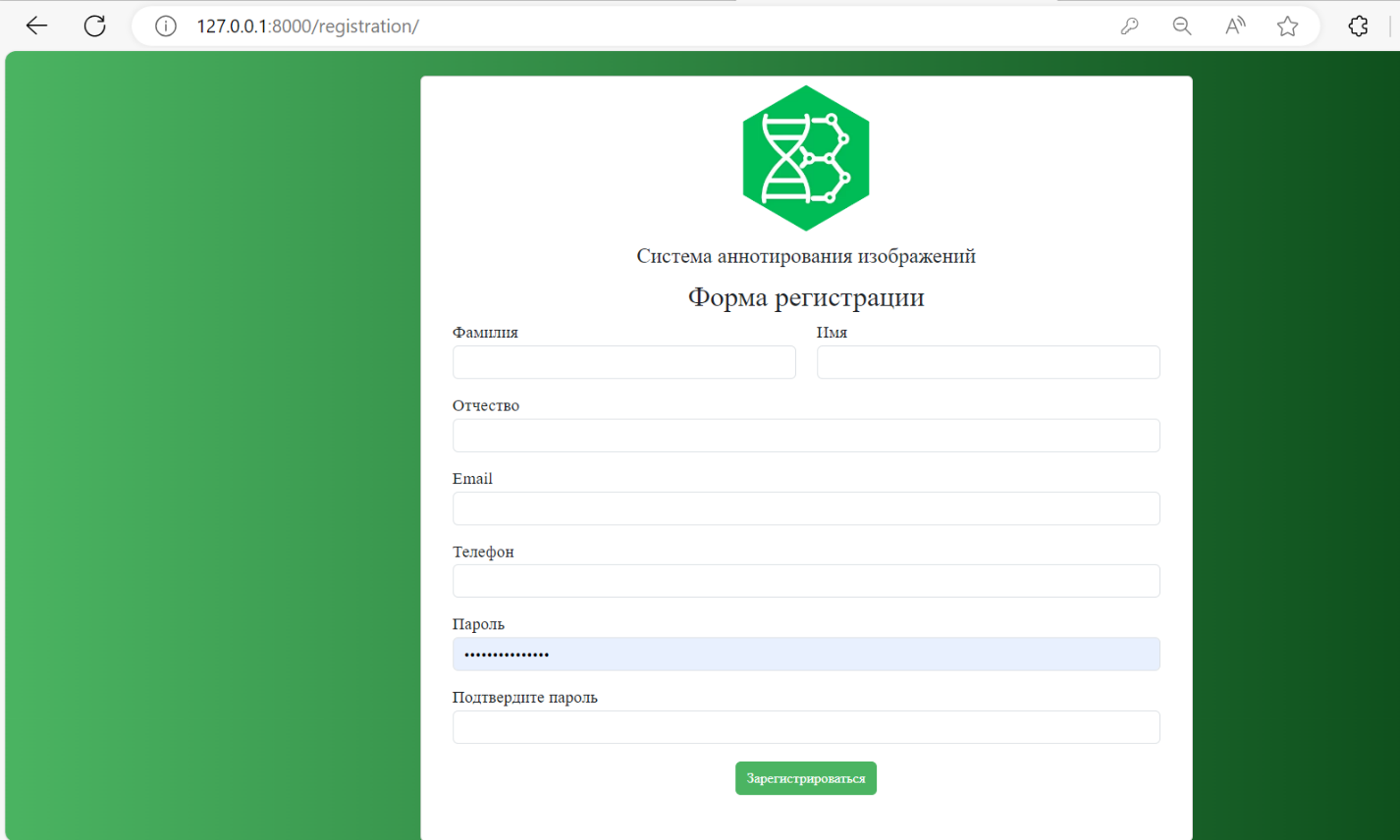


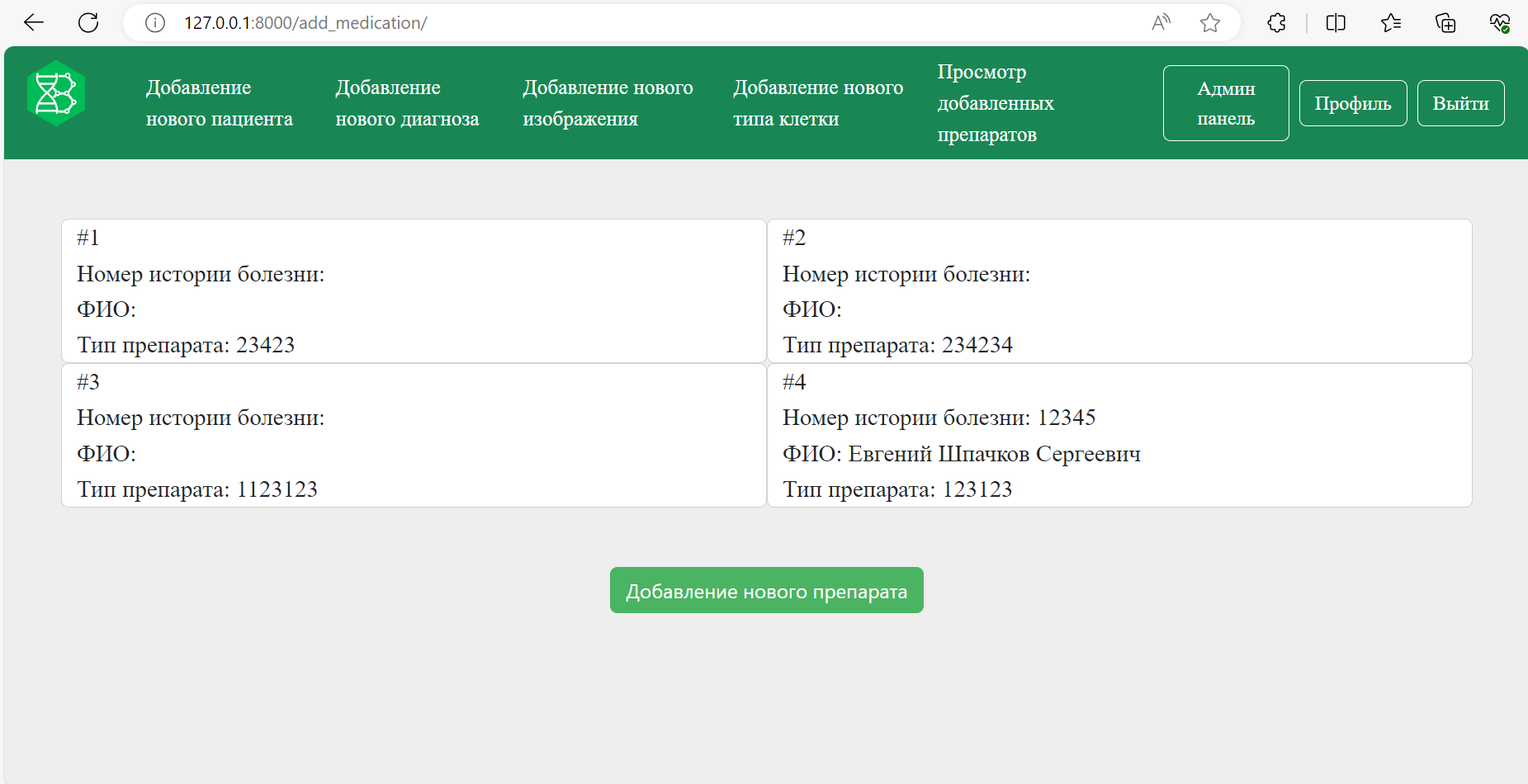
* 1. **Аутентификация пользователей**

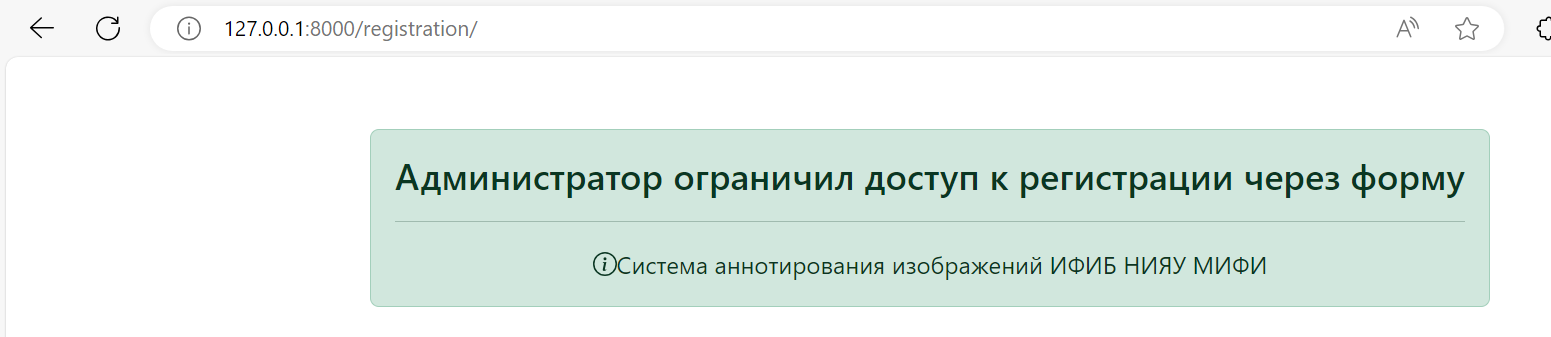
В сервисе предусмотрена аутентификация пользователей и последующее взаимодействие используя сессии. Для этого был создан отдельный контроллер SignInView, который позволяет пользователю выполнять вход в аккаунт и SignOutView, чтобы выходить из него. Если пользователь захотел прекратить работу, он может выйти из аккаунта, послав POST запрос logout/. После этого сессия будет считаться завершенной. Получить доступ к фунционалу веб-приложенпя без аутентификации также не получится.

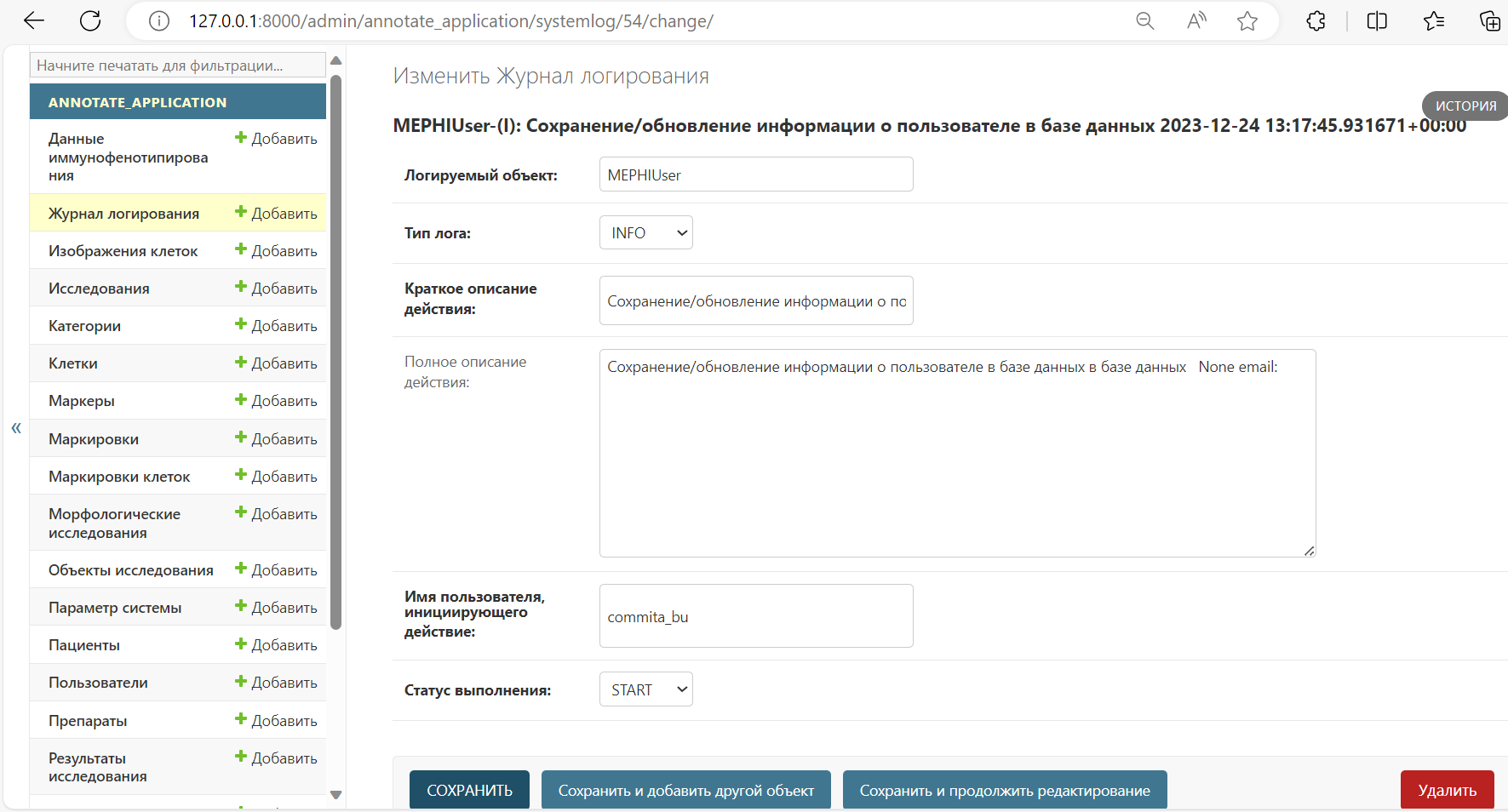
* 1. **Веб приложение**
     1. **Интерфейс веб приложения**

Веб приложение состоит различных вкладок, к каждой из которых подключены кастомные стили.









1. **Тестирование**
   1. **Тестирование серверной части приложения**

При разработке серверной части приложения были проведены ручное, модульное и интеграционные тестирования.

* + 1. **Ручное тестирование**

Ручное тестирование серверной части приложения проводилось с помощью инструмента Postman и через веб-интерфейс.

Postman это программа для отправки запросов на сервер и получения ответов. С помощью неё удобно проводить ручное тестирование, получать ответы от сервера, добавлять нужные поля в заголовки и выбирать тип авторизации.

# **Заключение**

В соответствии с техническим заданием было разработано приложение, которое состоит из серверной части, мобильного приложения и базы данных.

В приложении представлены такие функции как:

* регистрация и вход в аккаунт пользователя;
* создание и просмотр групп пользователя;
* возможность добавлять изображения клеток и их маркировок на эти изображения;
* возможность получать информацию о пациентах и справочниках, требуемых специалистам (справочник с терминами);
* наличие админ-панели;
* логирование информации о пользователе
* scd2 версионирование данных, загружаемых пользователем

Была спроектирована база данных, которая позволяет обращаться с информацией, которая нужна для функционирования приложения. Серверное приложение, которое позволяет производить действия по обработке данных и мобильное приложение, которое является клиентской частью. С помощью него пользователь может получать информацию и вносить изменения в удобном виде.

Произведено тестирование серверной части приложения и ручное тестирования серверной и клиентской части.

Возможное дальнейшее развитие сервиса связано с улучшением интерфейса веб приложения, добавления новых функций для аналитики опухолей.

# 