

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет

Кафедра

вычислительной техники

«Вычислительная техника»

Специальность: 09.05.01 «Применение и эксплуатация
автоматизированных систем специального
назначения»
Специализация: «Эксплуатация вычислительных машин,
комплексов, систем и сетей специального
назначения»

ДИПЛОМНЫЙ ПРОЕКТ

на тему

БИЗНЕС-ПРОЕКТ «ПРИЛОЖЕНИЕ ДЛЯ АНАЛИЗА И
ОБРАБОТКИ РЕНТГЕНОВСКИХ СНИМКОВ»

Студент М.Т. Нагаев 11.06.25 Нагаев Марат Тимурович
(подпись, дата) (ФИО полностью)

Руководитель Н.С. Карамышева 11.06.25 Карамышева Н.С.
(подпись, дата) (фамилия, инициалы)

Нормоконтролёр А.С. Бычков 11.06.25 Бычков А.С.
(подпись, дата) (фамилия, инициалы)

Рецензент А.Т. Чернышова 12.06.25 Севастьянов
(должность, место работы) (подпись, дата) (фамилия, инициалы)

Работа допущена к защите (протокол зас. кафедры от 23.05.2025 г. № 16)

Заведующий кафедрой М.А. Митрохин
(подпись) (фамилия, инициалы)

Работа защищена с отметкой отлично
(протокол заседания ГЭК от 19.06.2025 г. № 6)

Секретарь ГЭК

Е.И. Гурин
(подпись)

Гурин Е.И.
(фамилия, инициалы)

Пенза 2025

Федеральное государственное бюджетное образовательное учреждение
высшего образования

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

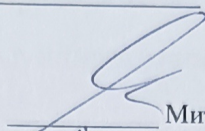
Факультет

Кафедра

Вычислительной техники

Вычислительная техника

«Утверждаю»
Заведующий кафедрой



Митрохин М.А.

«6» мая 2025 г.

ЗАДАНИЕ НА ДИПЛОМНЫЙ ПРОЕКТ

1. Студент Нагаев Марат Тимурович гр. 20ВОЭ1 факультета вычислительной техники.

2. Тема работы: «Бизнес-проект «Приложение для анализа и обработки рентгеновских снимков».

Тема утверждена приказом ПГУ № 601/о от 06.05.2025 г.

3. Руководитель работы к.т.н., доцент Карамышева Н.С.

4. Задание на работу (назначение разработки, исходные данные и т.п.).

Разработать бизнес-проект: приложение для анализа и обработки рентгеновских снимков. Приложение для анализа и обработки рентгеновских снимков должно выполнять следующие функции: добавление, изменение и удаление обследований; добавление, изменение и удаление рентгеновских снимков из обследования; получение снимка; формировать медицинское заключение. Приложение для анализа и обработки рентгеновских снимков должно разрабатываться с помощью среды разработки Qt Creator 14. База данных должна разрабатываться для СУБД «SQLite3».

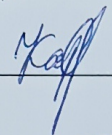
5. Перечень подлежащих разработке вопросов.

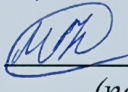
Требуется произвести анализ предметной области, определить функциональный состав программы, выбрать стек технологий для реализации программы, разработать алгоритм работы программы, разработать диаграммы: прецедентов, активностей, последовательностей, объектов, классов, с последующим написанием программы.

6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1.	Анализ предметной области	10	06.05.25- 09.05.25	
2.	Проектирование программного приложения	10	10.05.25 — 13.05.25	
3.	Разработка базы данных	20	14.05.25- 18.05.25	
4.	Отладка и тестирование	10	19.05.25- 22.05.25	
5.	Разработка экономической и бжд части	10	23.05.25- 27.05.25	
6.	Оформление графической части	10	28.05.25	
7.	Оформление выпускной работы	30	29.05.25- 12.06.25	

Дата выдачи задания «06.05» 2025 года.

Руководитель дипломного проекта  06.05.25 Карамышева Н.С.
(подпись, дата)

Задание к исполнению принял студент  06.05.25 Нагаев М.Т.
(подпись, дата)

Работу к защите допустить
Декан факультета  Фионова Л.Р.
18.06.2025 (подпись, дата)

Реферат

Пояснительная записка 168 листов, 12 рисунков, 14 таблиц и 6 приложений

ПРИЛОЖЕНИЕ, МАММОГРАФИЯ, ВРАЧЕБНЫЕ ЗАКЛЮЧЕНИЯ, C++, QT, SQLITE

Объектом исследования является процесс формирования врачебных заключений и анализа рентгеновских снимков.

Цель работы – разработка приложения для анализа и обработки рентгеновских снимков.

В процессе работы необходимо выполнить анализ предметной области, сформулировать техническое задание для разрабатываемого приложения, разработать логическую и физическую модели данных для разработки базы данных.

В результате проделанной работы разработано приложение для анализа и обработки рентгеновских снимков, состоящее из базы данных и приложения для работы с ней.

База данных спроектирована с использованием CASE-средства StarUML. База данных разработана для СУБД SQLite3, приложение реализовано в среде разработки Qt Creator 14.

					ПГУ1.090501.13.001 ПЗ			
Изм.	Лист	№ докум.	Подпись	Дата	Бизнес-проект «Приложение для анализа и обработки рентгеновских снимков» Пояснительная записка	Лит.	Лист	Листов
Разраб.		Нагаев М.Т.		11.06.25				
Провер.		Карамышева Н.С.		11.06.25			2	168
Н. Контр.		Бычков А.С.		11.06.25		ФВТ, каф. ВТ		
Утверд.		Митрохин М.А.		11.06.25		гр.20ВОЭ1		

Содержание

Перечень сокращений и обозначений

Введение.....	6
1. Анализ предметной области.....	9
1.1 Описание предметной области.....	9
1.2 Анализ аналогов.....	11
1.3 Цели и задачи разрабатываемого программного приложения.....	14
1.4 Анализ целевой аудитории.....	15
2. Техническое задание.....	16
2.1 Назначение разработки.....	16
2.2 Требования к программе.....	16
2.2.1 Требования к функциональным характеристикам.....	16
2.2.2 Требования к надежности.....	17
2.2.3 Требования к составу и параметрам технических средств.....	18
2.3 Требования к программной документации.....	19
2.4 Стадии и этапы разработки.....	19
2.5 Порядок контроля и приемки.....	20
3. Обоснование целесообразности выбранных заданных средств разработки..	21
3.1 Язык программирования C++.....	21
3.2 Язык программирования Go.....	21
3.3 CASE-средство StarUML.....	22
3.4 СУБД SQLite3.....	23
3.5 Среда разработки Qt Creator 14.....	23
4. Разработка приложения.....	26
4.1 Логическое проектирование базы данных.....	26
4.2 Физическое проектирование базы данных.....	41
4.3 Архитектура приложения	41
4.4 Описание программы.....	47
4.5 Методика испытаний программы.....	51
5. Оценка затрат на разработку и внедрение приложения для анализа и обработки рентгеновских снимков.....	56

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		3

5.1 Организация производства разработки приложения.....	56
5.2 Расчет себестоимости разработки приложения для анализа и обработки рентгеновских снимков.....	59
5.3 Оценка эффективности приложения для анализа и обработки рентгеновских снимков.....	65
5.4 Анализ рисков при разработке и внедрении приложения для анализа и обработки рентгеновских снимков.....	66
6. Общие принципы обеспечения безопасности на рабочем месте оператора приложения для анализа и обработки рентгеновских снимков.....	68
6.1 Актуальность вопросов безопасности на рабочем месте.....	69
6.2 Требования безопасности при работе с электроприборами.....	69
6.3 Влияние ПЭВМ на физиологию человека.....	69
6.4 Техника безопасности при работе с персональным компьютером.....	70
Заключение.....	72
Список использованных источников.....	73
Приложение А – UML-диаграммы.....	74
Приложение А.1.....	75
Приложение А.2.....	76
Приложение А.3.....	77
Приложение Б – SQL-скрипт.....	78
Приложение В – Результаты испытаний.....	97
Приложение Г – Листинг программы.....	139
Приложение Д – Презентация.....	300

Перечень сокращений и обозначений

В настоящей работе применяют следующие сокращения и обозначения

БД – база данных

ГОСТ – государственный стандарт

ЕГИСЗ – Единая государственная информационная система в сфере здравоохранения

ЕСПД – единая система программной документации

МРОТ – минимальный размер оплаты труда

НФ – нормальная форма

ОС – операционная система

ПО – программное обеспечение

ПЭВМ – персональная электронно-вычислительная машина

РМЖ – рак молочной железы

ЦНС – центральная нервная система

ЭВМ – электронно-вычислительная машина

DICOM – стандарт хранения и передачи медицинских изображений

FK (Foreign Key) – внешний ключ

ID (Identifier) – идентификатор объекта

PK (Primary Key) – первичный ключ

SQL (Structured Query Language) – язык структурированных запросов

UML (Unified Modeling Language) – унифицированный язык моделирования

					ПГУ1.090501.13.001 ПЗ	Лист
						5
Изм.	Лист	№ докум.	Подпись	Дата		

Введение

Современные информационные технологии играют важную роль в самых разных отраслях человеческой жизнедеятельности. В начале 1970х началось массовое внедрение информационных технологий в медицину. Появились стандартизированные протоколы для хранения и передачи медицинской информации: HL7 и DICOM (стандарт хранения и передачи медицинских изображений и отчетности), широко стали применяться технологии телемедицины, экспертные системы [1].

В настоящий момент цифровизация здравоохранения продолжается: в 2023 году российский рынок медицинских технологий вырос до 46,63 миллиардов рублей, продемонстрировав рост на 27% [2]. Согласно исследованию Smart Ranking, пристальное внимание уделяется дистанционному мониторингу здоровья, VR и внедрению нейросетевых технологий.

Нейросети являются частью понятия ИИ (искусственный интеллект). В последние годы с изобретением больших языковых моделей интерес к этой теме сильно вырос не только в России, но и в мире. Другим направлением развития нейросетей является компьютерное зрение - технология, позволяющая извлекать значимую информацию из визуальных данных (изображения, видео), анализировать, принимать решения и осуществлять действия на основе этой информации [3].

В 2020 году в Москве начался эксперимент по использованию технологий компьютерного зрения в медицине. По данным на начало 2023 года, нейросети проанализировали более 8.5 миллионов изображений, помогая врачам с постановкой правильного диагноза [4]. На текущий момент существуют решения для автоматизации диагностики 38 заболеваний, 41 направление эксперимента [5]. В 2022 году в России был принят ГОСТ на применение ИИ в медицине, началось внедрение ИИ по всей стране.

В последние годы по данным Высшей школы экономики в России сложился дефицит врачей. С 2000 года количество врачей в государственных

					ПГУ1.090501.13.001 ПЗ	Лист
						6
Изм.	Лист	№ докум.	Подпись	Дата		

больницах снизилось с 608,7 тыс. до 541,5 тыс. При этом общее количество врачей выросло до 744,1 тыс, что объясняется перетоком кадров в частные клиники. Рентгенологи и УЗИ-специалисты относятся к особо дефицитным кадрам как в стационаре, так и в городских поликлиниках [6]. В этой связи повышение производительности труда врачей с помощью внедрения современных информационных технологий становится особенно актуальным.

Таким образом, разработка приложения для анализа и обработки рентгеновских снимков является актуальной и необходимой по следующим причинам:

- увеличение скорости работы врача (единое ПО для всего: от получения рентгеновского снимка и его обработки до формирования врачебного заключения);
- упрощение постановки диагноза (приложение имеет систему фильтров, которые способны улучшать качество снимков);
- снижение радиационной нагрузки на пациента (приложение работает в составе программно-аппаратного комплекса, который обеспечивает более высокую чувствительность к рентгеновскому излучению чем традиционная пленочная технология, благодаря чему дозу излучения можно снизить);
- улучшение качества постановки диагноза (при формировании врачебного заключения врач не сможет забыть что-то указать).

В целом, разработка приложения для анализа и обработки рентгеновских снимков является важным шагом к повышению эффективности работы врачей, улучшению качества постановки диагнозов и упрощению рутинных процессов.

Приложение для анализа и обработки рентгеновских снимков включает в себя базу данных и клиент-серверное приложение обработки данных. При проектировании базы данных использовалось CASE-средство StarUML для СУБД SQLite3, для реализации приложения была применена среда разработки Qt Creator 14.

1 Анализ предметной области

1.1 Описание предметной области

Приложение для анализа и обработки рентгеновских снимков предназначено для получения рентгеновских снимков, упрощения составления врачебных заключений, учета медицинских обследований и пациентов, упрощения постановки диагноза врачом.

Актуальностью создания приложения для анализа и обработки рентгеновских снимков является их широкое применение в разных отраслях промышленности и медицины. Приложение позволяет работать с любыми рентгеновскими снимками, однако особое внимание уделено маммографическим снимкам, которые используются для диагностики рака молочной железы. В 2022 году от него умерло 670000 человек во всем мире [7]. Для выявления рака обычно используется комбинация двух видов исследований: маммографии и УЗИ [8]. Приложение позволяет формировать врачебное заключение по стандартизированному протоколу BIRADS [9], что упрощает труд врача, позволяет снизить уровень смертности и ускорить выздоровление.

Современная диагностика рака молочной железы производится при помощи цифровых программно-аппаратных комплексов, состоящих из оборудования для получения маммографических снимков и специализированного ПО для их получения, хранения и обработки. Для получения снимка используются рентгеновский моноблок и плоскопанельный детектор рентгеновского излучения. Более традиционный, аналоговый способ с использованием рентгеновской пленки обладает меньшей чувствительностью, при этом пациент получает более высокую дозу радиации [10].

DICOM – стандарт для хранения и передачи медицинских изображения (в том числе для маммографии). Помимо собственно изображения хранятся

					ПГУ1.090501.13.001 ПЗ	Лист
						8
Изм.	Лист	№ докум.	Подпись	Дата		

DICOM теги – метаинформация, которая содержит данные о пациенте, враче, оборудовании, на котором был получен снимок.

После каждого посещения профильного специалиста пациенту выдается документ – врачебное заключение. Медицинские учреждения почти во всем мире переходят на стандартизированную систему отчетности BIRADS, которая содержит набор требований к врачебным заключениям [11]. Это упрощает коммуникацию между разными врачами, которые обследуют одного пациента и позволяет пациенту в случае необходимости обследоваться в разных больницах. В отчет, написанный по BIRADS входят:

- показания к проведению исследования;
- описание структуры молочных желез по ACR;
- описаний патологических находок;
- заключение с выставлением итоговой категории по шкале BI-RADS;
- дальнейшие рекомендации.

Структура молочных желез в терминологии BIRADS описывается буквой латинского алфавита – A, B, C, D, которые описывают проницаемость молочной железы к рентгеновскому излучению [11].

К патологическим находкам относят объемные образования, нарушение архитектоники, асимметрию, кальцинаты и сопутствующие признаки. Объемное образование – это пространственная структура патологической плотности, занимающая определённый объём молочной железы, обязательно визуализирующаяся на изображениях в двух проекциях (CC и MLO). Характеризуется формой, контуром и плотностью. Если потенциальное образование определяется только на изображении в одной проекции, используется термин "асимметрия" до тех пор, пока наличие образования не будет подтверждено или опровергнуто в ходе дообследования.

Нарушение архитектоники – «локальный» участок ткани молочной железы с нарушением привычной анатомической структуры, в виде тонких прямых линий или спикул, расходящихся из одной точки; локального втяжения или деформации ткани железы, без визуализации объемного образования.

					ПГУ1.090501.13.001 ПЗ	Лист
						9
Изм.	Лист	№ докум.	Подпись	Дата		

Кальцинаты – новообразование, представляющее скопление солей кальция. Характеризуются видом (доброкачественные или злокачественные) и распределением (диффузное, региональное).

Итоговая оценка BIRADS представляет число, характеризующее вероятность рака молочной железы:

- пациент недостаточно обследован, вероятность рака установить нельзя (категория 0);
- маммограммы без патологических образований (категория 1);
- доброкачественные изменения (категория 2);
- вероятнее всего доброкачественные изменения (категория 3);
- подозрительные изменения (категория подразделяется на 4А, 4В, 4С);
- достоверно злокачественные изменения (категория 5);
- гистологически подтвержденный РМЖ (категория 6).

Целью данной работы является разработка приложения со встроенной базой данных. Приложение будет обеспечивать получение рентгеновских снимков, формирование врачебных заключений системы BIRADS, обработку рентгеновских снимков и их анализ. База данных будет хранить информацию о пациентах и обследованиях.

1.2 Анализ аналогов

Для того чтобы определиться с функционалом будущего приложения необходимо провести анализ аналогов. Рассмотрим каждый из них.

MicroDicom – приложение для просмотра медицинских снимков и отчетов в формате DICOM. Позволяет проводить измерения, делать пометки, имеет предустановленный набор для удобного просмотра различных частей тела на рентгеновских снимках (кости, печень, череп и другие). Для медицины есть базовый инструментарий: угол Кобба, возможность просмотра МРТ снимков. Не имеет функционала по формированию медицинских заключений, врачу требуется писать их вручную. Никак не адаптирован под российское

					ПГУ1.090501.13.001 ПЗ	Лист
						10
Изм.	Лист	№ докум.	Подпись	Дата		

законодательство, не имеет сертификации, интерфейс на английском языке. Стоимость — 350 американских долларов за 1 рабочее место.

Iray Idemo – ПО для работы с плоскопанельными детекторами Iray. Позволяет получать рентгеновские снимки. К недостаткам можно отнести отсутствие возможности редактирования снимков, невозможность формирования врачебных заключений и англоязычный интерфейс. Поставляется вместе с оборудованием, включен в стоимость детектора.

Квант.МИС Лайт – ПО для работы с электронными медицинскими документами. Позволяет составлять протокол консультации, медицинское заключение по результатам медицинского осмотра, протокол медицинской манипуляции и другое. Создается российским разработчиком и адаптирован под российское законодательство, имеет интеграцию с ЕГИСЗ. Не имеет функций для работы с рентгеновскими снимками. Стоимость – 80400 рублей в год.

Osirix Dicom Viewer – ПО для просмотра рентгеновских снимков. Имеет поддержку модулей, в том числе имеется модуль для генерации отчетов по BIRADS. Интерфейс на английском языке. К недостаткам так же можно отнести работу только в macOS (нет поддержки российских операционных систем и Windows). Не имеет функций для получения рентгеновских снимков. Стоимость – от 70 долларов США в месяц.

После проведения анализа аналогов разрабатываемого приложения для анализа и обработки рентгеновских снимков можно сделать вывод, что данное приложение будет иметь следующие преимущества перед конкурентами: комплексное решение (получение снимка, анализ снимка, генерация врачебного заключения в одной программе) и адаптация на использование в российских реалиях: русскоязычный интерфейс, на российском оборудовании, с учетом требований российского законодательства и на российских операционных системах.

					ПГУ1.090501.13.001 ПЗ	Лист
						11
Изм.	Лист	№ докум.	Подпись	Дата		

1.3 Цели и задачи разрабатываемого программного приложения

Цели создания приложения:

- увеличение скорости работы врачей;
- упрощение постановки диагноза;
- улучшение качества постановки диагнозов.

В результате внедрения приложения планируется достичь ряда задач, а именно:

- повысить эффективность работы врачей;
- снизить лишнюю нагрузку на врачей;
- повысить качество постановки диагнозов.

1.4 Анализ целевой аудитории

Целевая аудитория приложения включает в себя медицинских специалистов и медицинские учреждения. К медицинским специалистам можно отнести онкологов, радиологов и рентгенологов. Дополнительно приложение может быть использовано IT интеграторами, которые внедряют PACS (Picture Archiving and Communication System). Они могут интегрировать его со своими системами.

Небольшие клиники часто выбирают бюджетное рентгеновское оборудование: например, плоскопанельный детектор от фирмы Iray. Оборудование более высокого класса стоит ощутимо дороже, но предоставляет более качественное ПО. Программное обеспечение Iray не позволяет редактировать полученные снимки. Внедрение приложения позволит таким клиникам сэкономить на закупке (не покупать отдельно ПО для редактирования снимков, оформления врачебных заключений), при этом используя уже имеющиеся технические средства.

					ПГУ1.090501.13.001 ПЗ	Лист
						12
Изм.	Лист	№ докум.	Подпись	Дата		

2 Техническое задание

2.1 Назначение разработки

Приложение для анализа и обработки рентгеновских снимков должно быть предназначено для автоматизации формирования врачебных заключений, получения и обработки снимков.

2.2 Требования к программе

2.2.1 Требования к функциональным характеристикам

Программное обеспечение должно выполнять следующие функции:

- формирование медицинских заключений;
- получение рентгеновских снимков;
- обработка снимков (поворот, инверсия цвета, фильтры CLAHE и Unsharp Mask);
- добавление аннотаций (текстовая надпись, стрелка)
- измерения (линейка, угол)
- хранение справочных данных:
 - 1) данные о пациентах;
 - 2) данные об обследованиях;
 - 3) данные о снимках.

Входными данными приложения должны быть:

- данные о пациенте;
- параметры съемки рентгеновского снимка.

Выходными данными приложения должны быть:

- врачебное заключение;
- обработанные рентгеновские снимки.

2.2.2 Требования к надежности

Приложение для анализа и обработки рентгеновских снимков должно работать без перебоев, контролировать вводимые данные и в случае некорректного ввода данных должна сообщать об этом пользователю.

При вводе входной информации должен осуществляться ее контроль на

					ПГУ1.090501.13.001 ПЗ	Лист
						13
Изм.	Лист	№ докум.	Подпись	Дата		

допустимость вводимых значений по следующим правилам:

- вводимые данные не должны противоречить заявленному типу данных;
- данные, передаваемые на рентгеновский аппарат должны находиться в пределах паспортных значений изделия;
- вводимые данные не должны противоречить законам логики.

При выполнении операции по изменению и удалению данных должна быть обеспечена целостность базы данных.

Конечный пользователь не должен иметь доступ к исходным данным приложения.

2.2.3 Требования к составу и параметрам технических средств

Приложение должно быть предназначено для использования на локальном компьютере. Рабочие станции должны иметь следующие минимальные технические характеристики:

- операционная система Windows 7 и выше;
- процессор Intel Core 3,9 ГГц и выше;
- свободное место на жестком диске 2 ГБ и выше;
- оперативную память 8 ГБ и выше;
- монитор с диагональю не менее 16 дюймов;
- клавиатура;
- мышь.

2.3 Требования к программной документации

Приложение для анализа и обработки рентгеновских снимков должно сопровождаться документацией, содержащей:

- техническое задание;
- описание логического проектирования БД;
- описание физического проектирования БД;
- описание программы;
- программу и методику испытаний;
- описание применения.

					ПГУ1.090501.13.001 ПЗ	Лист
						14
Изм.	Лист	№ докум.	Подпись	Дата		

Структура и содержание программной документации должны соответствовать требованиям стандартов Единой системы программной документации (ЕСПД).

2.4 Стадии и этапы разработки

Этапы разработки приложения для анализа и обработки рентгеновских снимков представлены в таблице 1.

Таблица 1 — Стадии и этапы разработки

Стадии	Этапы	Объем работы в процентах	Срок выполнения	Исполнитель
1 Анализ предметной области	Анализ предметной области	10		Нагаев М.Т.
2 Проектирование программного приложения	- проектирование БД - проектирование приложения - проектирование модуля для составления врачебных заключений	40		Нагаев М.Т.
3 Разработка базы данных	Разработка базы данных	10		Нагаев М.Т.
4 Отладка и тестирование	Отладка и тестирование	10		Нагаев М.Т.
5 Разработка экономической части	Оценка затрат на разработку и внедрение программного приложения контроля выполнения задач	5		Нагаев М.Т.
6 Разработка раздела БЖД	Оценка тяжести труда эксплуататора программного приложения	5		Нагаев М.Т.
7 Оформление графической части	Оформление графической части	10		Нагаев М.Т.

Стадии	Этапы	Объем работы в процентах	Срок выполнения	Исполнитель
8 Оформление Выпускной работы	Оформление пояснительной записки	10		Нагаев М.Т.

2.5 Порядок контроля и приемки

Для проверки выполнения системой всех ее функций в соответствии с требованием технического задания необходимо разработать тестовые наборы данных. Протестировать действия системы на возможные действия пользователя: запуск программы, заполнение полей данными, выбор пунктов меню, нажатие кнопок, работа с базой данных. Система соответствует требованиям технического задания, если она выполняет все заданные функции при различных входных данных из тестовых наборов данных.

Проверка и приемка системы должна осуществляться при ее работоспособности и в присутствии заказчика, выполнения всех возложенных на нее функциональных возможностей, а также при наличии полной документации на программу (пояснительная записка с приложениями: текстом программы и результатом текстовых примеров).

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		16

3 Обоснование целесообразности использования выбранных средств разработки

На основании анализа требований к функциональным характеристикам приложения для анализа и обработки рентгеновских снимков были выбраны наиболее актуальные средства разработки.

3.1 Язык программирования C++

Язык программирования – это набор правил, предназначенный для записи компьютерных программ. Далее из полученного текста специализированные программы (трансляторы, компоновщики и др.) практически без участия человека формируют код, предназначенный для процессора. В объектно-ориентированных языках центральное место отведено данным, а выполнение функций так или иначе организовано вокруг этих данных.

C++ — это язык программирования, который был разработан Бьярне Страуструпом в 1980-х годах как расширение языка C, добавляющее поддержку объектно-ориентированного программирования.[12] Он предназначен для создания высокопроизводительных и масштабируемых приложений, особенно в областях, требующих точного управления ресурсами, таких как системы реального времени, высоконагруженные серверные приложения и медицинские системы.

Язык C++ разработан прежде всего для системного программирования, однако со временем сообщество создало инструменты, упрощающие разработку приложений с графическим интерфейсом на C++. Современные версии C++ сконцентрированы на предоставлении инструментов для разработки безопасных программ такие как умные указатели, профили и специальные шаблонные классы. По данным различных рейтингов языков программирования, таких как TIOBE [13], IEEE Spectrum [14] C++ занимает ведущие позиции по использованию, что упрощает найм квалифицированных

специалистов и упрощает поддержку проекта по сравнению с менее популярными аналогами.

3.2 Язык программирования Go

Для разработки вспомогательного модуля генерации отчетности был выбран язык программирования Go.

Go – компилируемый язык программирования, созданный компанией Google в 2009 году. В отличие от C++, который больше нацелен на системное программирование и разработку приложений с графическим интерфейсом, Go имеет широкую коллекцию модулей, создаваемых в том числе сторонними разработчиками для самых разных целей. [15] Одним из таких модулей является docx, решение для создания и редактирования файлов формата Office Open XML.

Для Go характерны:

- простой синтаксис;
- быстрая компиляция;
- встроенный пакетный менеджер;
- сборщик мусора.

3.3 CASE-средство StarUML

Для разработки базы данных используются специальные CASE-средства для разработки концептуальных, логических и реляционных моделей.

Популярным CASE-средством является StarUML — бесплатный инструмент моделирования [16].

Преимущества StarUML:

- поддержка UML 2.0;
- генерация кода на различных языках программирования
- система плагинов, что позволяет интегрировать дополнительные возможности;
- экспорт в различные форматы (png, svg, jpg);
- бесплатное ПО.

					ПГУ1.090501.13.001 ПЗ	Лист
						18
Изм.	Лист	№ докум.	Подпись	Дата		

Недостатками StarUML являются:

- ограниченная поддержка командной работы;
- отсутствие встроенной системы контроля версий.

3.4 СУБД SQLite3

Для функционирования базы данных необходимо выбрать систему управления базами данных.

В приложении для анализа и обработки рентгеновских снимков СУБД используется для хранения обследований, пациентов и настроек приложения.

СУБД SQLite3 – это компактная встраиваемая система управления базами данных.

Выделяется ряд особенностей SQLite3:

- кроссплатформенность;
- бессерверный режим упрощает разворачивание и настройку;
- встраиваемость: в отличие от большинства СУБД не зависит от сторонних программ и библиотек;
- надежность – 100% исходного кода покрыто тестами;
- компактность [17].

3.5 Среда разработки Qt Creator 14

Qt Creator 14 — это современная интегрированная среда разработки (Integrated Development Environment, IDE), она предлагает набор инструментов для разработки ПО для Windows, Android, iOS, а также веб-приложений и облачных сервисов.

К главным возможностям Qt Creator 14 можно отнести:

- многоязычная поддержка;
- редактирование и отладка кода;
- интеграция с Qt;
- совместная работа и управление версиями;
- создание пользовательского интерфейса;

					ПГУ1.090501.13.001 ПЗ	Лист
						19
Изм.	Лист	№ докум.	Подпись	Дата		

– расширяемость [17].

Qt Creator создана специально для разработки программ на языке C++ с использованием фреймворка Qt и предлагает богатый набор инструментов для поиска утечек памяти (Heob, Valgrind), форматирования кода (Clang-Tidy), анализа производительности.

Новая версия поддерживает Qt 6 и CMake [17].

Qt 6 – новая версия популярного фреймворка для разработки приложений на C++, получившая многочисленные оптимизации отрисовки интерфейса, который благодаря технологии RHI может быть аппаратно ускорен на самых разных платформах с помощью технологий Vulkan, Metal и DirectX. За счет разбиения кода фреймворка на более мелкие части приложение стало занимать меньший объем дискового пространства.

CMake – система метасборки, получившая широкое распространение среди C++ разработчиков, ставшая негласным индустриальным стандартом. В новой версии Qt Creator и Qt6 она используется по умолчанию вместо устаревшего qmake. В отличие от родных систем сборки, таких как make и MSBuild, CMake не привязан к какой-то конкретной ОС, будь то Windows или Unix и позволяет разрабатывать кроссплатформенные программные продукты, упрощая непрерывную интеграцию и кросс-компиляцию [18].

					ПГУ1.090501.13.001 ПЗ	Лист
						20
Изм.	Лист	№ докум.	Подпись	Дата		

4 Разработка приложения

Диаграмма развертывания приложения для анализа и обработки рентгеновских снимков представлена на рисунке 1.

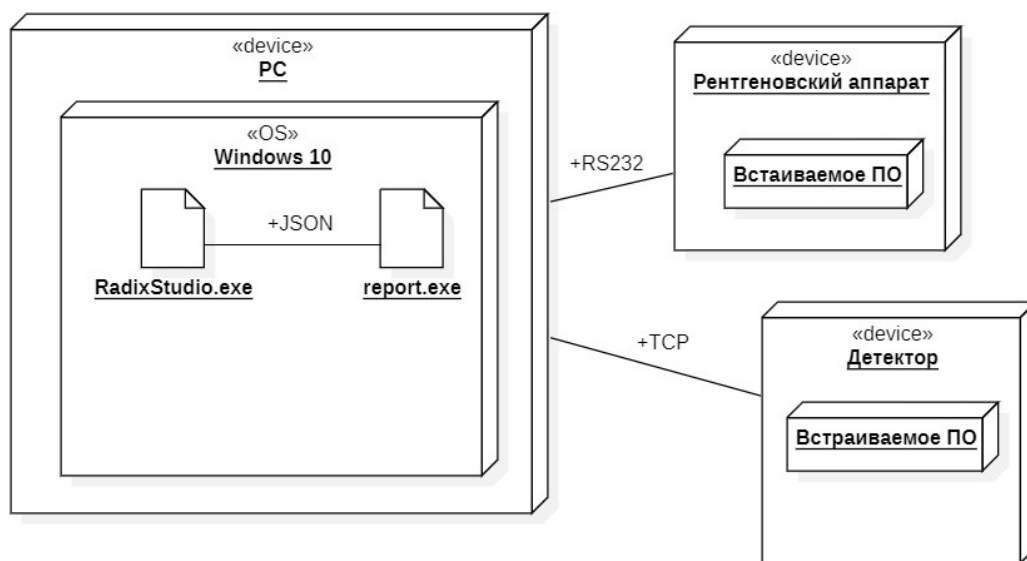


Рисунок 1 – Диаграмма развертывания

4.1 Логическое проектирование базы данных

Логическая модель базы данных приведена на рисунке 2 и содержит три сущности.

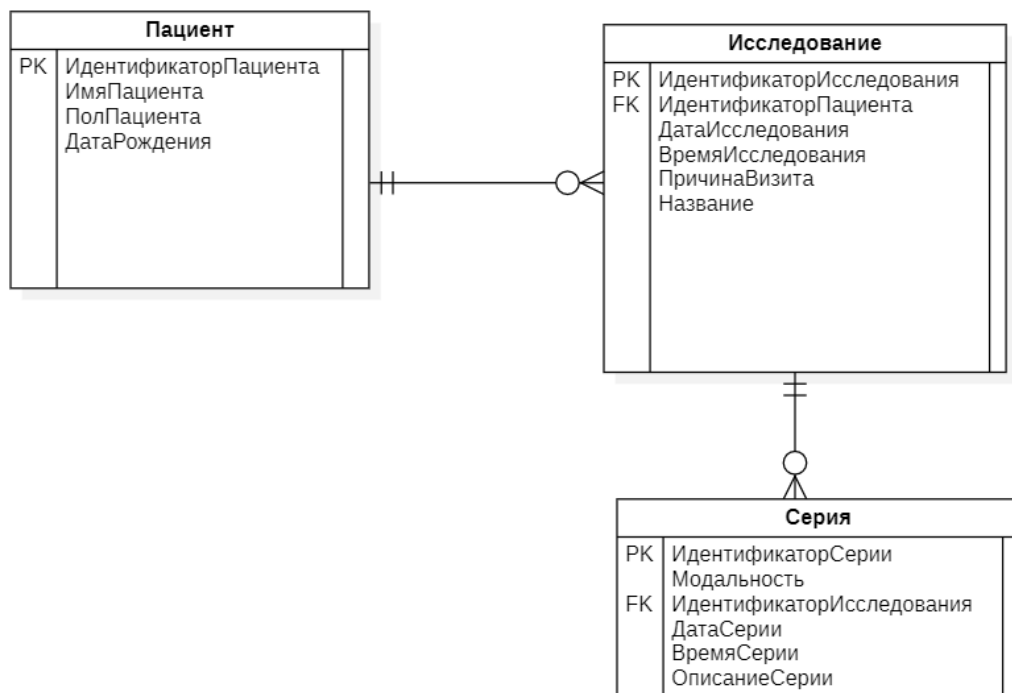


Рисунок 2 – Логическая модель данных

Описание атрибутов сущностей логической модели приведено в таблицах 2 – 4.

Таблица 2 – Описание атрибутов сущности «Пациент»

Имя атрибута	Назначение	Первичный ключ	Внешний ключ
ИдентификаторПациента	ID пациента	РК	
ИмяПациента	ФИО пациента		
ПолПациента	Пол пациента		
ДатаРождения	Дата рождения пациента		

Таблица 3 – Описание атрибутов сущности «Исследование»

Имя атрибута	Назначение	Первичный ключ	Внешний ключ
ИдентификаторИсследования	ID исследования	РК	
ИдентификаторПациента	ID пациента		FK
ДатаИсследования	Дата исследования		
ВремяИсследования	Время исследования		
ПричинаВизита	Описание исследования		

Таблица 4 – Описание атрибутов сущности «Series»

Имя атрибута	Назначение	Первичный ключ	Внешний ключ
ИдентификаторСерии	ID серии снимков	РК	
Модальность	Модальность серии		

Имя атрибута	Назначение	Первичный ключ	Внешний ключ
ИдентификаторИсследования	ID исследования		FK
ДатаСерии	Дата серии		
ВремяСерии	Время серии		
ОписаниеСерии	Описание серии		

Процесс нормализации – это последовательный процесс приведения базы данных к эталонному виду, то есть переход от одной нормальной формы к следующей.

Отношение находится в 1 НФ, если все строки таблицы различные и все атрибуты являются атомарными.

Отношение находится во 2 НФ, если оно находится в 1 НФ и каждый не ключевой атрибут таблицы функционально полно зависит от первичного ключа.

Отношение находится в 3 НФ, если оно находится во 2 НФ и между не ключевыми атрибутами не существует транзитивных зависимостей.

Все атрибуты трех сущностей атомарны и находятся в 3 НФ. Для подтверждения данного фактора необходимо рассмотреть каждую сущность:

– сущность «Пациент» имеет ключевой атрибут «ИдентификаторПациента». Атрибуты «ИмяПациента», «ПолПациента», «ДатаРождения» данной сущности являются простыми и неделимыми;

– сущность «Серия» имеет ключевой атрибут «ИдентификаторСерии». Атрибут «Модалность», «ДатаСерии», «ВремяСерии», «ОписаниеСерии», «ИдентификаторИсследования» данной сущности является простым и неделимым;

– сущность «Исследование» имеет ключевой атрибут «ИдентификаторИсследования». Атрибуты «ДатаИсследования», «ВремяИсследования», «ПричинаВизита», «Название»,

«ИдентификаторПациента» данной сущности являются простыми и неделимыми.

Таким образом, была разработана логическая модель данных.

4.2 Физическое проектирование

На основании разработанной логической модели данных была построена физическая модель данных. Физическая модель данных разработана для СУБД Sqlite и представлена на рисунке 3.

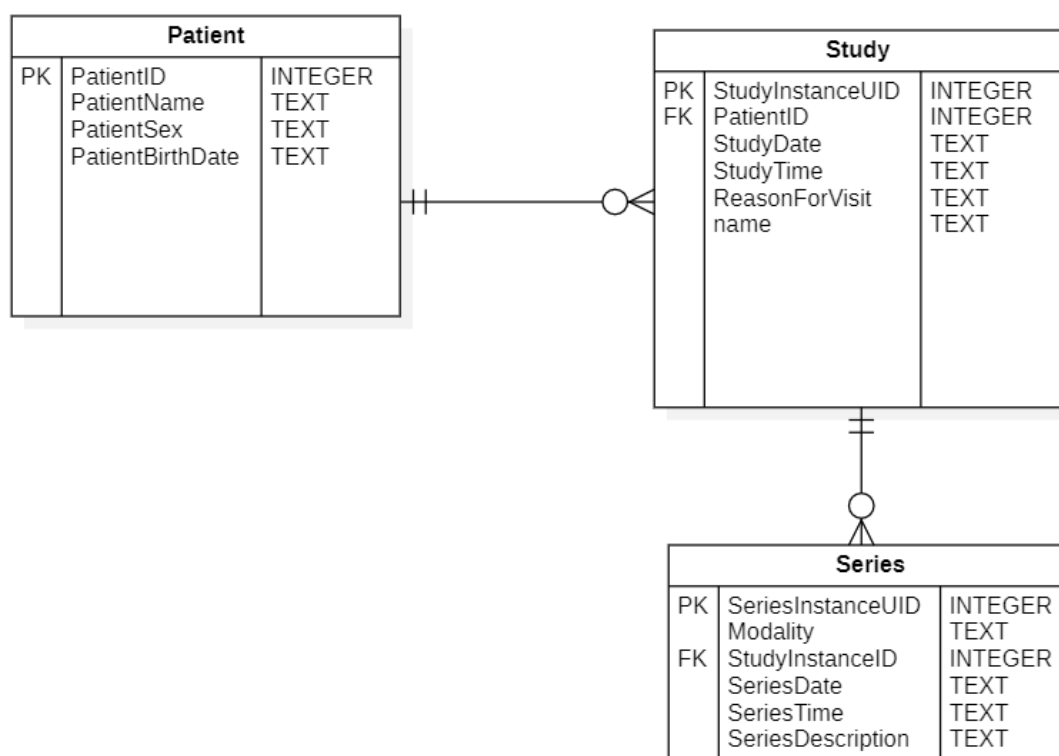


Рисунок 3 – Физическая модель данных

Физическая модель данных содержит три таблицы. Описание таблиц физической модели приведено в таблицах 5 — 7.

Таблица 5 – Описание полей таблицы «Patient»

Имя атрибута	Физическое имя	Нулевое значение	Ключевое поле	Тип данных
Идентификатор Пациента	PatientID	Не допустимо	PK	INTEGER
ИмяПациента	PatientName	Не допустимо		TEXT
ПолПациента	PatientSex	Не допустимо		TEXT
ДатаРождения	PatientBirthDate	Не допустимо		TEXT

Таблица 6 – Описание полей таблицы «Study»

Имя атрибута	Физическое имя	Нулевое значение	Ключевое поле	Тип данных
Идентификатор Исследования	StudyInstanceUID	Не допустимо	PK	INTEGER
Идентификатор Пациента	PatientID	Не допустимо	FK	INTEGER
ДатаИсследования	StudyDate	Не допустимо		TEXT
ВремяИсследования	StudyTime	Не допустимо		TEXT
ПричинаВизита	ReasonForVisit	Не допустимо		TEXT
Название	name	Не допустимо		TEXT

Таблица 7 – Описание полей таблицы «Series»

Имя атрибута	Физическое имя	Нулевое значение	Ключевое поле	Тип данных
Идентификатор Серии	SeriesInstanceUID	Не допустимо	PK	INTEGER
Модальность	Modality	Не допустимо		TEXT
Идентификатор Исследования	StudyInstanceUID	Не допустимо	FK	INTEGER
ДатаСерии	SeriesDate	Не допустимо		TEXT
ВремяСерии	SeriesTime	Не допустимо		TEXT
ОписанииСерии	SeriesDescription	Не допустимо		TEXT

Таким образом, была разработана физическая модель данных для СУБД Sqlite3.

4.3 Описание архитектуры

Программа реализована с помощью языка программирования C++ в среде разработки Qt Creator 14 и имеет модульную структуру, представленную на рисунке.

Название класса	Описание класса
RExaminationImgsVerticalPreview	Модуль для отображения списка радиологических изображений.
RExaminationTools	Модуль для отображения инструментов аннотации и измерения.
RStatusBar	Модуль для отображения статус-бара, отображающего информацию о подключении оборудования.
RExplorer	Модуль для отображения списка обследований. Предназначен для обеспечения работы функций добавления, изменения и удаления данных об обследованиях.
RImageInfo	Модуль для отображения информации о снимке.
RImagePreviewWidget	Модуль обеспечивающий вывод превью снимка.
RExaminationDevicePanel	Модуль для отображения элементов управления оборудованием: рентгеновским моноблоком и плоскопанельным детектором рентгеновского излучения.
NoiseEffect	Модуль для поддержки фильтра изображения — размытия Гаусса.
DicomManager	Модуль для поддержки стандарта DICOM.
RadixImage	Модуль, обеспечивающий поддержку операций над изображениями (поворот, инверсия, отражение, обрезка).
RadixController	Модуль для поддержки рентгеновского аппарата. Реализует команды начало излучения, остановку, передачу параметров на устройство (напряжение, ток, время экспозиции).
Venu1417	Модуль для поддержки детектора. Реализует чтение телеметрии, начало приема излучения и получение изображения.
Record	Модуль для хранения данных об обследовании.

Название класса	Описание класса
RUndoRedo	Модуль для хранения истории операций
ClaheEffect	Модуль для поддержки фильтра изображения — CLAHE.
SharpnessEffect	Модуль для поддержки фильтра изображения — нечеткой маски.
RGraphicsToolsLine	Модуль, реализующий поддержку графического инструмента — линейки
RGraphicsToolsRectangle	Модуль, реализующий поддержку графического инструмента — измерительного прямоугольника
RGraphicsToolsAngle	Модуль, реализующий поддержку графического инструмента — угла
RExaminationView	Модуль, реализующий собственную графическую сцену
MainWindow	Модуль главного окна

Диаграмма вариантов использования представлена на рисунке 5.



Рисунок 5 — Диаграмма вариантов использования

4.4 Описание программы

Папка assets хранит ресурсы приложения (шрифты, иконки, SQL скрипты).

Код приложения хранится в папке src.

Папка src/graphicEffects описывается графические эффекты (фильтры) изображения. Всего эффектов 3: CLAHE, нечеткая маска, размытие Гаусса. Эффекты реализованы при помощи библиотеки OPENCV. Пример реализации фильтра CLAHE (остальные реализованы аналогично):

```
#include "ClaheEffect.h"
#include <opencv2/core/ocl.hpp>
#include <opencv2/imgcodecs.hpp>
#include <opencv2/imgproc.hpp>
#include <opencv2/opencv.hpp>

ClaheEffect::ClaheEffect() : IGraphicsEffect() {
    m_value = 0;
}

void ClaheEffect::applyEffect(cv::UMat& image) {
    if (0 != m_value) {
        int k = 5;
        int clipLimit = m_value * k;
        cv::Ptr<cv::CLAHE> clahe = cv::createCLAHE();
        clahe->setClipLimit(clipLimit);
        clahe->setTilesGridSize(cv::Size(8, 8));

        cv::UMat clahed;
        if (image.channels() != 1) {
            cv::cvtColor(image, image, cv::COLOR_RGB2GRAY);
        }
        clahe->apply(image, clahed);
        image = clahed;
    }
}
```

					ПГУ1.090501.13.001 ПЗ	Лист
						30
Изм.	Лист	№ докум.	Подпись	Дата		

}

Папка `src/components` содержит графический интерфейс программы. Интерфейс описывается с помощью `ui` файла фреймворка Qt. В `cpp` и `h` файлах описывается логика работы графических компонентов.

Файл `src/components/RMessageItem.cpp` описывает сообщение, показываемое пользователю.

Файл `src/components/RExaminationView.cpp` описывает графическую сцену `RExaminationView`.

Метод `fit` масштабирует изображение.

Методы `enableSelectionMode`, `disableSelectionMode` включают/выключают режим выделения.

Метод `selectedArea` возвращает прямоугольную область выделения.

Папка `src/components/windows` содержит окна, показываемые пользователю. Интерфейс описывается с помощью `ui` файла фреймворка Qt. В `cpp` и `h` файлах описывается логика работы. Все окна — наследники класса `QDialog` и содержат кнопки, испускающие сигналы (используя механизм сигналов-слотов Qt). Таким образом реализуется отделение кода, описывающего интерфейс от бизнес-логики.

В папке `src/image` содержатся файлы, относящиеся к работе над изображениями.

Файл `src/image/DicomImage.cpp` содержит класс `DicomImage`, реализующий поддержку стандарта DICOM.

Метод `getDistance` возвращает расстояние между точками в сантиметрах, что позволяет реализовать инструмент линейка.

Метод `getImageFormat` возвращает формат картинки для библиотеки Qt, основываясь на DICOM тегах изображения.

Метод `getPixelData` возвращает указатель на графическую информацию, используя для этого библиотеку DCMTK.

Методы `getStudyDate`, `getStudyTime`, `getWidth`, `getHeight` возвращает значения тегов DICOM, используя для этого библиотеку DCMTK.

					ПГУ1.090501.13.001 ПЗ	Лист
						31
Изм.	Лист	№ докум.	Подпись	Дата		

Метод `getGeneratorParams` заполняет структуру рентгеновского моноблока (ток, напряжение, время экспозиции) тегов DICOM, используя для этого библиотеку DCMТK.

Метод `rotate` поворачивает изображение на заданный угол, используя метод `rotateImage` класса `DicomImage` библиотеки DCMТK.

Метод `cropImage` обрезает изображение, используя метод `createClippedImage` библиотеки DCMТK, предварительно сохраняя оригинал изображения.

Метод `save` сохраняет изображение, предварительно записав в теги DICOM актуальную информацию. Само сохранение работает с помощью метода `saveFile` библиотеки DCMТK.

Метод `readDicomFile` выполняет чтение: создает объект класса `DicomImage`, вычитывая значение тегов.

Файл `src/image/RUndoRedo.cpp` содержит класс `RUndoRedo`, реализующий поддержку операций отмены и повторения действия.

Метод `undo` отменяет последнюю операцию: удаляет элемент из стека `undo` и добавляет в стек `redo`. Возвращает объект класса `Operation`.

Метод `redo` повторяет последнюю операцию: удаляет элемент из стека `redo` и добавляет в стек `undo`. Возвращает объект класса `Operation`.

Класс `Operation` содержит информацию о операции (например угол поворота, который необходимо отменить), вид операции и реализует методы `undo/redo`.

Файл `src/image/RadixImage.cpp` содержит класс `RadixImage`, реализующий поддержку изображения.

Метод `addGraphicsItem` добавляет инструмент (линейку или угол) на графическую сцену Qt. После добавления создается объект `Operation`, который добавляется в стек `undoRedo`.

Метод `attach` добавляет изображение на сцену (`RExaminationView`).

Папка `src/hardware` содержит код для работы с оборудованием (получение рентгеновского снимка и телеметрии).

					ПГУ1.090501.13.001 ПЗ	Лист
						32
Изм.	Лист	№ докум.	Подпись	Дата		

Файл `src/hardware/fpd/Venu1417.cpp` содержит код для работы с детектором.

Метод `getFpdTelemetry` заполняет структуру, описывающую телеметрию детектора: его температуру и заряд. Для получения этих данных на детектор посылается команда. При этом используется библиотека от производителя оборудования.

Метод `startAcquisition` используется для получения снимка, посылая команду на детектор. При этом используется библиотека от производителя оборудования.

Метод `connect` используется для подключения к детектору по WI-FI. При этом используется библиотека от производителя оборудования.

Файл `src/hardware/xray/RadixController.cpp` содержит код для работы с рентгеновским аппаратом.

Метод `StartXray` используется для запуска рентгена. При этом используется библиотека `Qt Serial Port`, которая посылает команду на аппарат используя COM порт.

Метод `StopXray` реализован аналогично и используется для остановки излучения.

Методы `SetExpoTime`, `SetVoltage`, `SetCurrent` передают данные параметров генерации (время экспозиции, напряжение и ток) на рентгеновский аппарат.

В папке `src/components/graphic_tools` содержатся файлы, относящиеся к работе над изображениями.

Файл `src/components/graphic_tools` содержит класс `RGraphicsToolsLine`, реализующий поддержку инструмента линейка.

Метод `rotate` поворачивает линейку вместе с поворотом изображения.

Метод `flip` вызывается при вертикальном или горизонтальном отражении.

Остальные графические инструменты реализованы аналогично.

В папке `screens` содержится код экранов приложения — экран обследования и экран БД.

					ПГУ1.090501.13.001 ПЗ	Лист
						33
Изм.	Лист	№ докум.	Подпись	Дата		

Файл `src/screens/RExaminationScreen.cpp` описывает контроллер для экрана обследования и содержит 1 класс — `RExaminationScreen`.

Метод `onMakeSnapshot` вызывается при получении снимка.

Метод `onGotSnapshot` вызывается когда снимок получен.

Метод `onShowAboutDialog` вызывается при открытии диалога о программе.

Метод `voltageChanged` вызывается при изменении напряжения рентгеновского аппарата на панели устройств и передает актуальное значение на рентгеновский аппарат с помощью метода `SetVoltage` объекта `RadixController`.

Метод `currentChanged` вызывается при изменении тока рентгеновского аппарата на панели устройств и передает актуальное значение на рентгеновский аппарат с помощью метода `SetCurrent` объекта `RadixController`.

Метод `exptimeChanged` вызывается при изменении времени экспозиции рентгеновского аппарата на панели устройств и передает актуальное значение на рентгеновский аппарат с помощью метода `SetExptime` объекта `RadixController`.

Файл `src/screens/RExplorerScreen.cpp` описывает контроллер для экрана проводника и содержит 1 класс — `RExplorerScreen`.

Метод `init` вызывается при создании объекта и используется для создания графического интерфейса пользователя.

Вспомогательное приложение `report.exe` нужно для формирования медицинского заключения.

Код приложения состоит из одного файла `main.go`, внутри которого объявлена функции `main`. Приложение формирует отчет формата `DOCX`, ожидая на вход `JSON` файл, оформленный по схеме. Для этого используется модуль `go-docx` и его функции `AddParagraph`, `AddText`, `Justification` и `Size`.

4.5 Руководство пользователя

При запуске программы пользователь попадает на экран со списком обследований — проводник (рисунок 6).

Проводник предназначен для создания, поиска и доступа к сохраненным обследованиям, а также к файлам DICOM. Его интерфейс включает в себя:

- панель инструментов;
- список обследований.

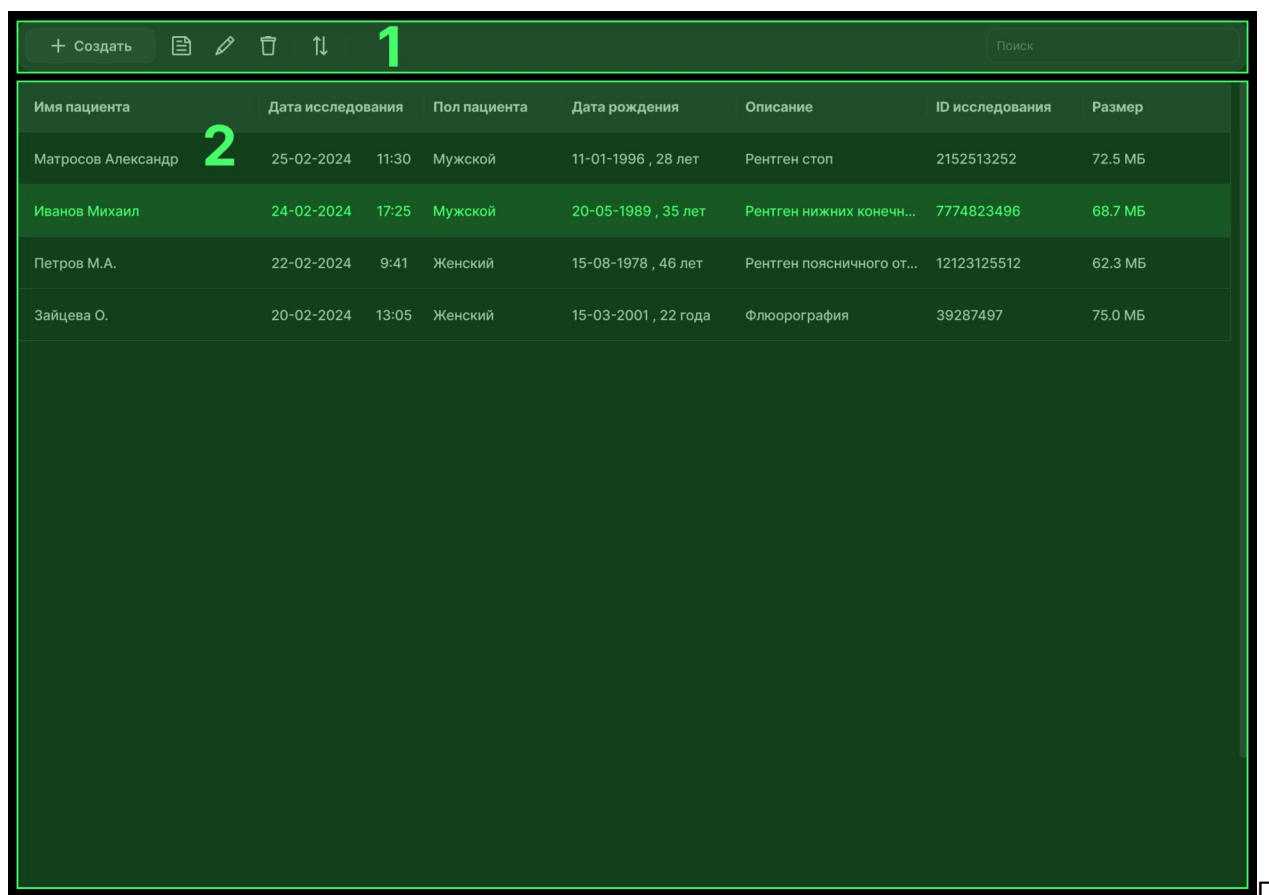


Рисунок 6 — Интерфейс проводника

Панель инструментов проводника (рисунок 7) расположена над списком обследований, предоставляя все необходимые функции для работы с обследованиями и открытия файлов.

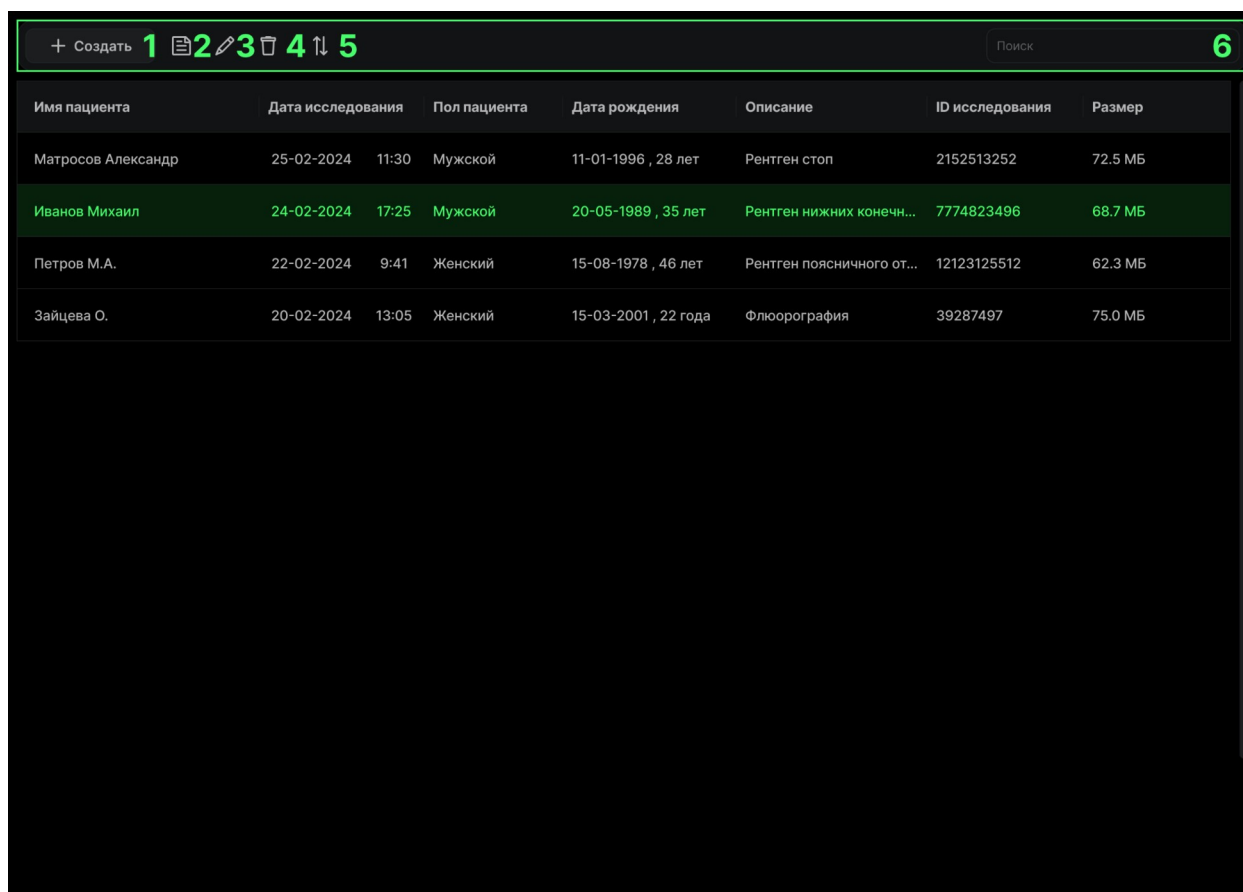


Рисунок 7 — Панель инструментов проводника

Кнопка «Создать» инициирует новое обследование и позволяет добавить в него информацию о пациенте.

Кнопка «Файл» открывает контекстное меню, в котором возможен выбор следующих операций:

- открыть файл DICOM;
- открыть папку DICOM;
- экспорт обследования.

Первая кнопка контекстного меню открывает окно в файловой системе рабочей станции для просмотра и редактирования файла DICOM.

Вторая кнопка контекстного меню открывает папку с файлами DICOM в файловой системе рабочей станции для просмотра и редактирования.

Последняя кнопка контекстного меню сохраняет выбранное обследование в файловую систему рабочей станции как папку с файлами в одном из форматов: DICOM, PNG, JPEG, BMP, TIFF.

Кнопка «Редактирование обследования» открывает ответственное окно для выбранного обследования.

Кнопка «Удаление» позволяет удалять одно или несколько выбранных обследований. Для выбора нескольких обследований следует зажать клавишу CTRL и выбрать в списке те, которые нужно удалить.

Кнопка «Сортировка» упорядочивает список обследований по имени, по дате и по идентификатору обследования. Повторное нажатие на выбранный пункт в меню сортирует список обследований по возрастанию или убыванию.

Поиск — строка ввода текста для поиска по сохраненным данным пациентов и атрибутам обследований.

Под панелью инструментов проводника расположен список обследований, организованный в виде таблицы (рисунок 8). В столбцах содержится информация об обследованиях:

- имя пациента;
- дата проведения обследования.
- пол пациента;
- дата рождения и возраст пациента;
- описание обследования;
- id обследования (идентификатор);
- размер сохраненных данных обследования в мегабайтах.

+ Создать						Поиск	
Имя пациента 1	Дата исследования 2		Пол пациента 3	Дата рождения 4	Описание 5	ID исследования 6	Размер 7
Матросов Александр	25-02-2024	11:30	Мужской	11-01-1996 , 28 лет	Рентген стоп	2152513252	72.5 МБ
Иванов Михаил	24-02-2024	17:25	Мужской	20-05-1989 , 35 лет	Рентген нижних конечн...	7774823496	68.7 МБ
Петров М.А.	22-02-2024	9:41	Женский	15-08-1978 , 46 лет	Рентген поясничного от...	12123125512	62.3 МБ
Зайцева О.	20-02-2024	13:05	Женский	15-03-2001 , 22 года	Флюорография	39287497	75.0 МБ

Рисунок 8 — Интерфейс инструментов проводника

Главный экран (рисунок 9) предназначен для работы с изображениями и управления рентгеновским оборудованием. На главном экране расположены:

- панель инструментов;
- панель миниатюр изображений;
- панель состояния;
- панель управления;
- рабочая область.

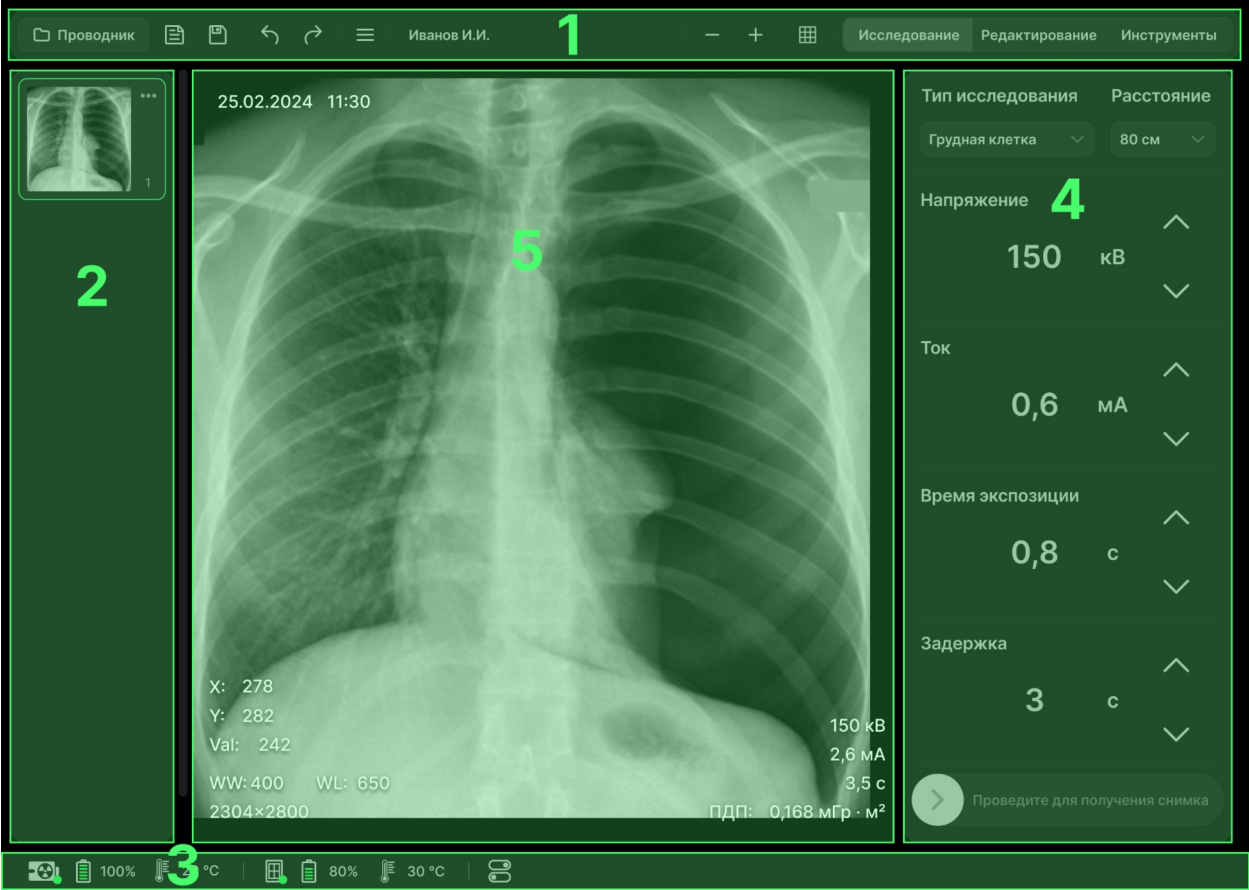


Рисунок 9 — Главный экран

Панель инструментов (рисунок 10) предназначена для работы с файлами и вызовом дополнительных функций.

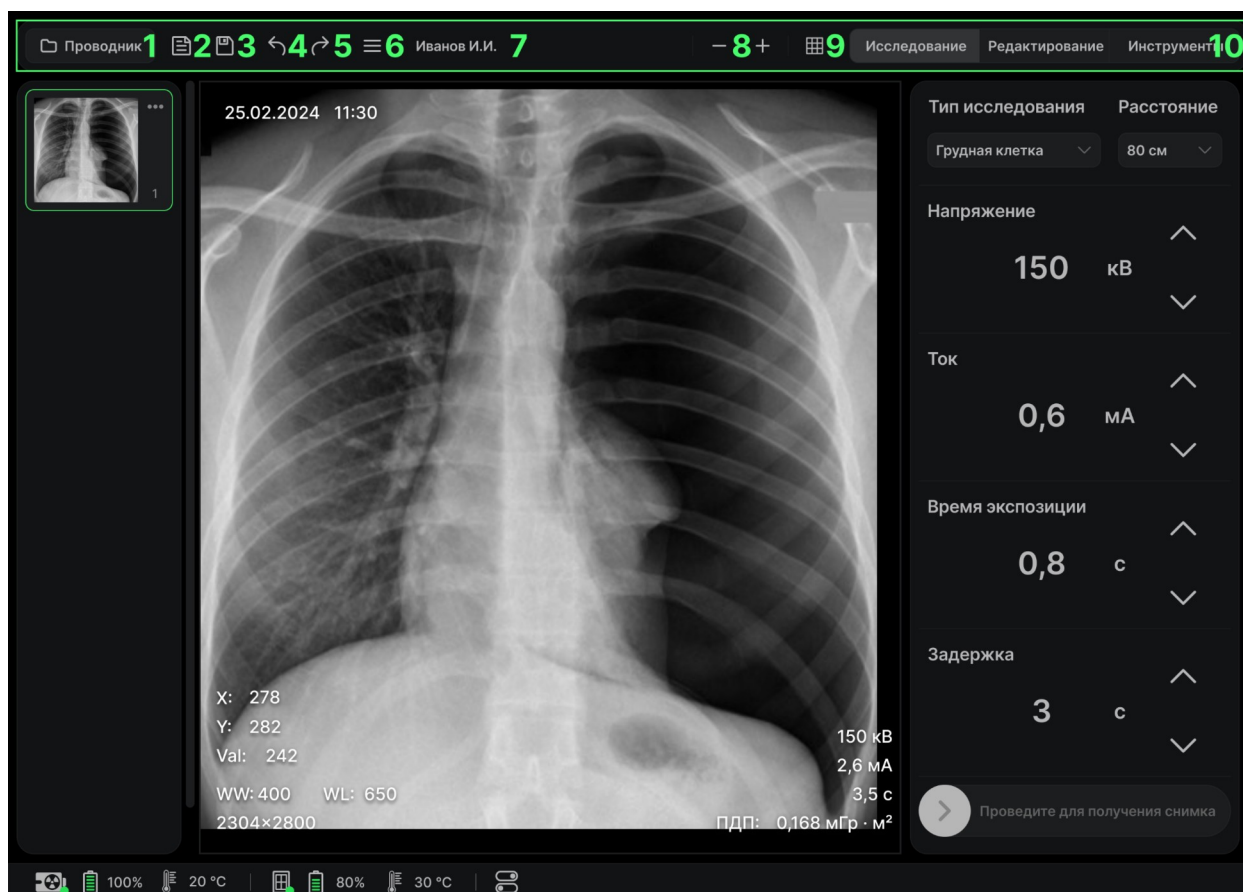


Рисунок 10 — Панель инструментов

Кнопка «Проводник» закрывает текущее обследование и открывает список обследований.

Кнопка «Файл» открывает контекстное меню с двумя пунктами:

- открыть файл DICOM;
- экспорт изображения.

Первый пункт контекстного меню открывает окно файловой системы для выбора и просмотра файла DICOM в открытом обследовании.

Второй пункт контекстного меню открывает окно файловой системы, чтобы в нем выбрать место для экспорта выбранного изображения в одном из форматов: DICOM, PNG, JPEG, BMP, TIFF.

Кнопка «Сохранить» сохраняет все текущие изменения в открытом обследовании.

Кнопка «Отмена операции» Отменяет последнее выполненное действие на шаг назад. Например, если на изображение случайно была добавлена аннотация, то «Отмена действия» приведет к ее удалению.

Кнопка «Возврат действия» возвращает последнее отмененное действие. Например, если на изображении случайно удалить аннотацию, то «Возврат действия» приведет к ее возврату.

Кнопка «Главное меню» открывает доступ к дополнительным функциям и настройке вида интерфейса.

Имя пациента — строка для отображения имени пациента, сохраненного при создании обследования.

Кнопки масштабирования увеличивают или уменьшают масштаб просматриваемого изображения.

Кнопка «Сетка» позволяет выбирать и настраивать варианты разбивки рабочей области для одновременного просмотра нескольких изображений.

Панель управления содержит следующие вкладки:

- управление рентгеновским оборудованием;
- редактирование изображений;
- инструменты.

Панель миниатюр изображений (рисунок 11) позволяет переключаться между изображениями в обследовании или папке с файлами DICOM. Из панели миниатюр можно удалить, экспортировать или дублировать выбранное изображение.

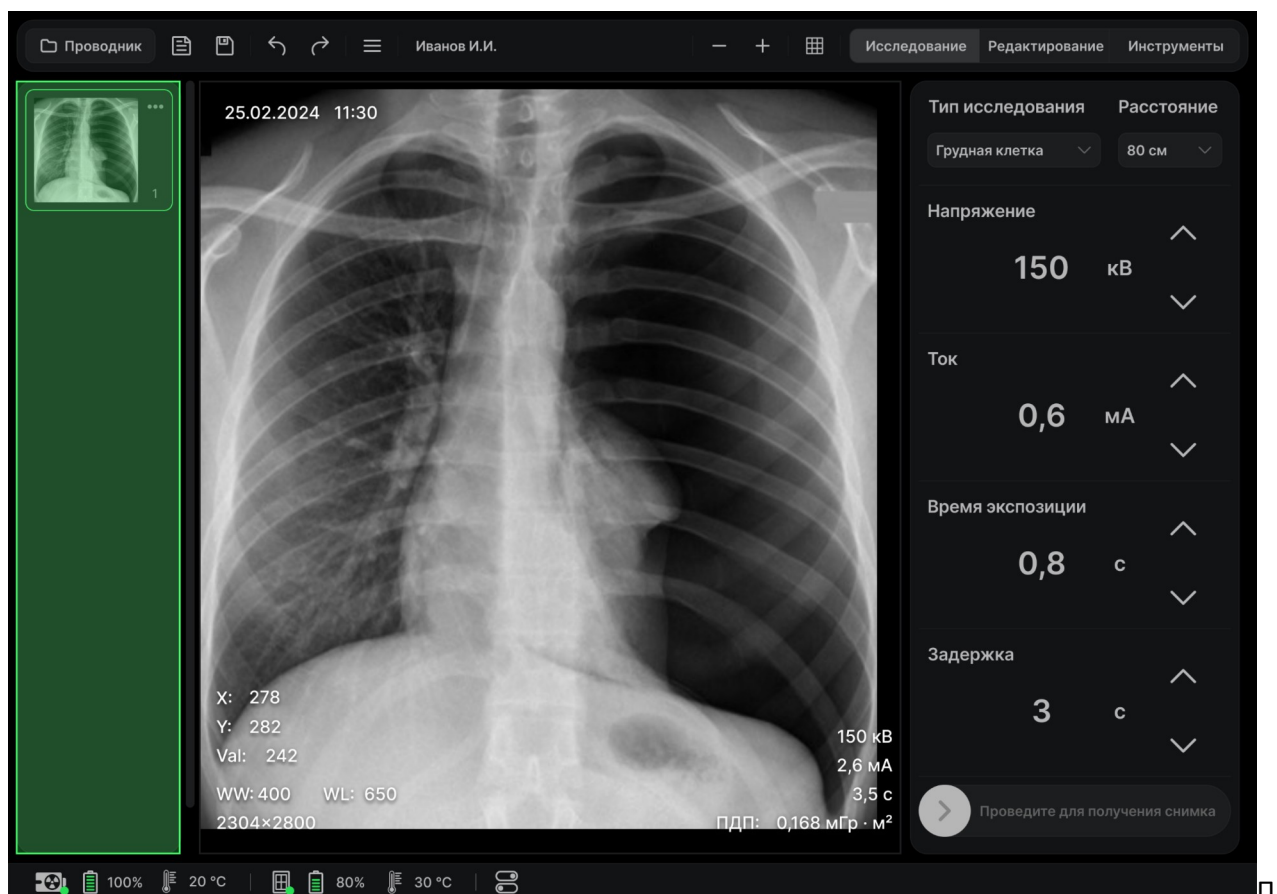


Рисунок 11 — Панель миниатюр

На панели состояния (рисунок 12) отображается важная информация о рентгеновском оборудовании, включая статус его подключения, температуру и уровень заряда аккумуляторов. Панель разделена на 3 блока:

- состояние подключения, уровень заряда аккумулятора и температура рентгеновского моноблока;
- состояние подключения, уровень заряда аккумулятора и температура плоскпанельного детектора;
- выбор рентгеновского оборудования для управления в программе.

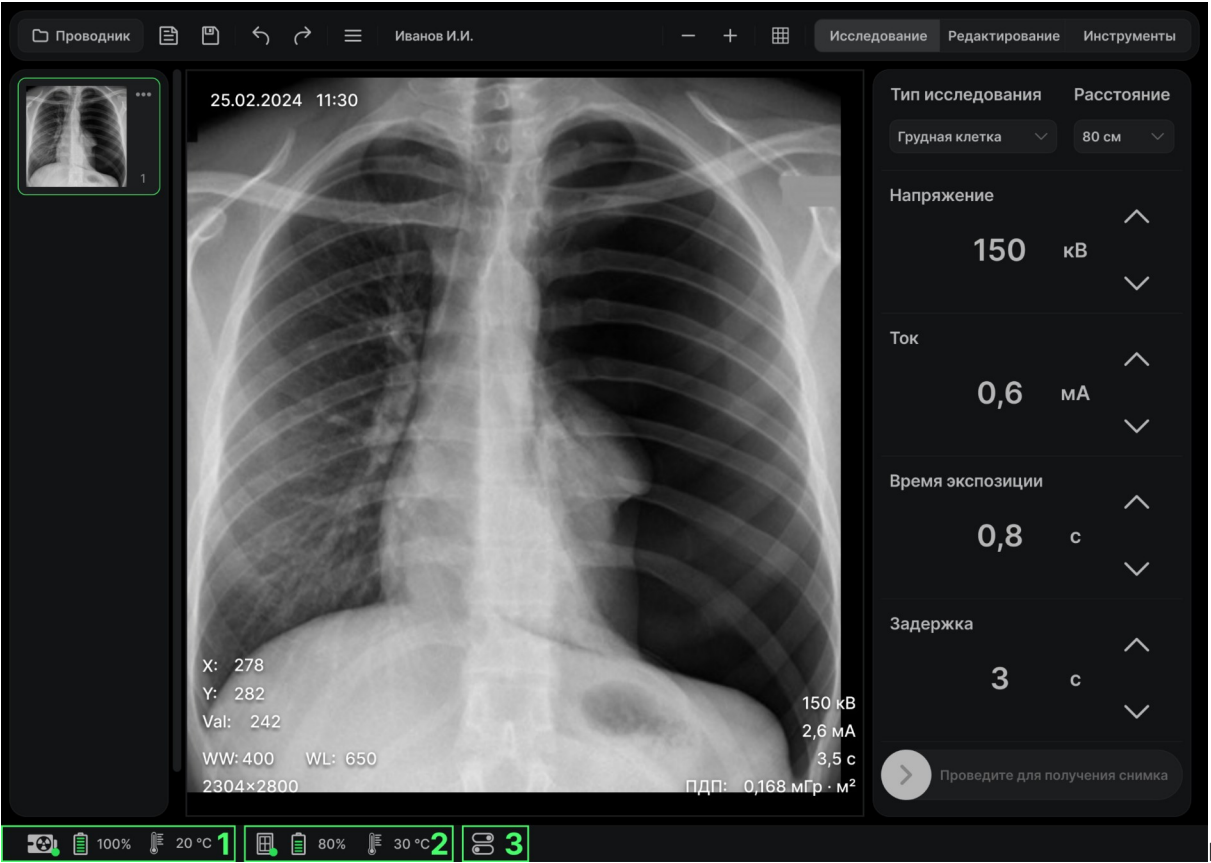


Рисунок 12 — Панель состояния

Панель управления (рисунок 13) используется для работы с изображениями: их получением и редактированием.

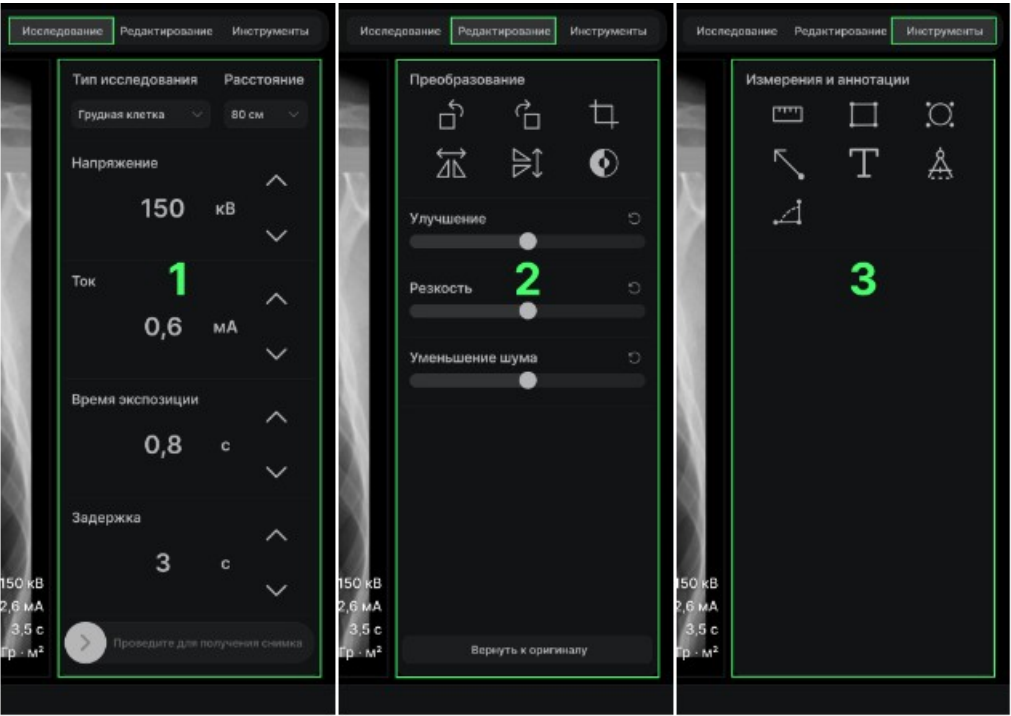


Рисунок 13 — Панель управления

Вкладка «Исследование» предназначена для настройки параметров рентгеновского оборудования, выбора профилей параметров экспозиции рентгеновского моноблока (тип обследования), установки расстояния до пациента и запуска экспозиции для получения изображения.

Вкладка «Редактирование» содержит инструменты преобразования и редактирования изображений.

Вкладка «Инструменты» предоставляет набор инструментов для проведения измерений на изображениях и добавления аннотаций.

Рабочая область (рисунок 14) предназначена для работы с изображениями. Рабочая область может выглядеть по-разному в зависимости от режима просмотра. В рабочей области наиболее важные данные и теги DICOM просматриваемого изображения, отображаются поверх него и группируются по углам.

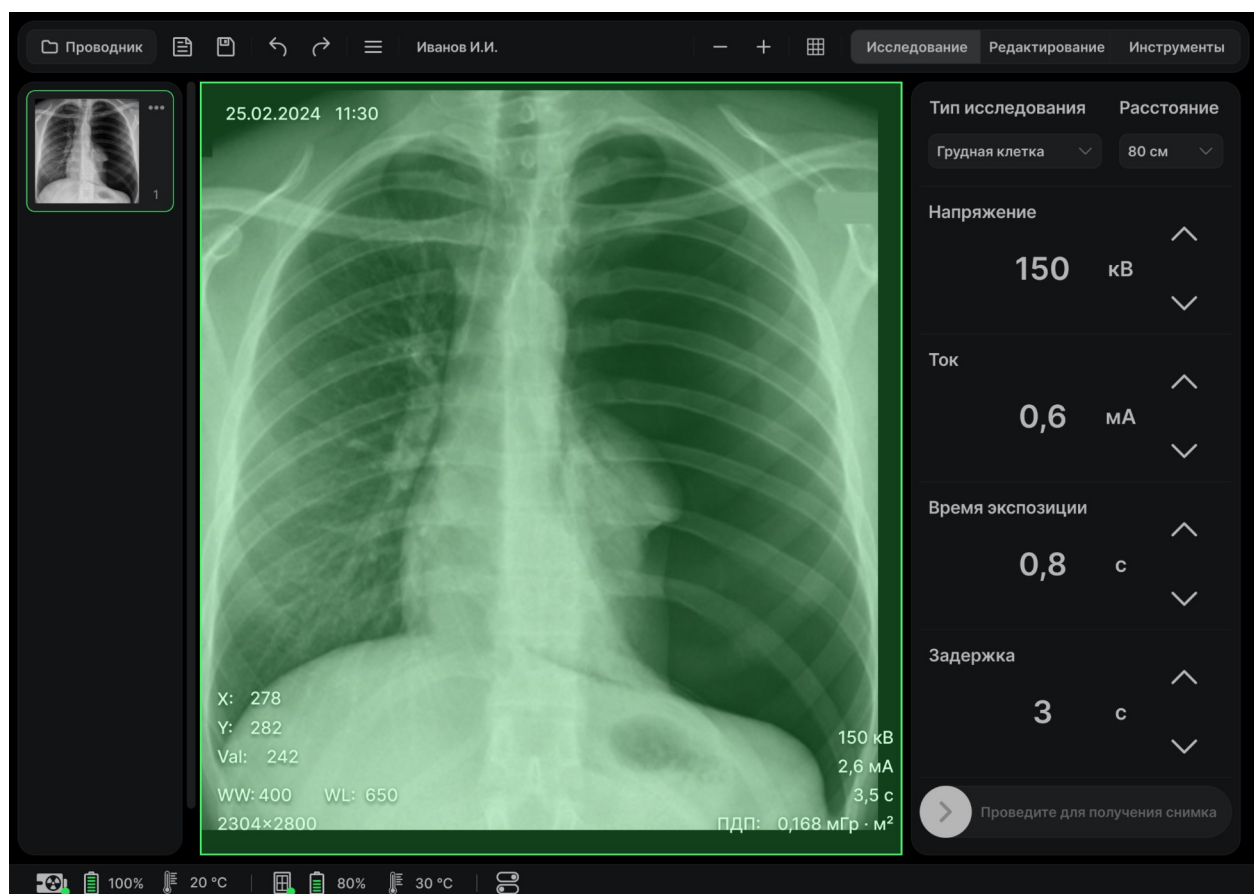


Рисунок 14 — Рабочая область

Обследования предназначены для хранения информации и рентгеновских изображений пациентов. При создании обследования оно добавляется в локальную базу данных на рабочей станции.

Для создания нового обследования (рисунок 15) необходимо в панели инструментов проводника нажать кнопку «Создать». В открывшемся окне создания исследования заполнить поля с информацией о пациенте:

- фамилия (обязательное поле);
- имя;
- отчество;
- дата рождения;
- пол;
- описание.

После добавления информации о пациенте нажать на кнопку «Создать», после чего обследование будет доступно в проводнике.

Рисунок 15 — Создание обследования

Редактирование данных обследования (рисунок 16) дает возможность изменять ранее введенную информацию о пациенте. Чтобы отредактировать данные обследования необходимо в списке обследований выбрать

					ПГУ1.090501.13.001 ПЗ	Лист
						45
Изм.	Лист	№ докум.	Подпись	Дата		

обследование, далее в панели инструментов нажать кнопку «Редактирование обследования». В открывшемся окне изменить необходимую информацию о пациенте редактирования обследования.

Редактирование исследования

Фамилия: Иванов

Имя: Виктор

Отчество: Петрович

Дата рождения: 10 12 1980

Пол: ☒ Мужской ☐ Женский

Описание: Подозрение на перелом

Пара ваших личных пояснений, чтобы было проще вспомнить детали или другой врач смог быстро войти в курс дела.

Сохранить **Отменить**

Рисунок 16 — Редактирование обследования

После редактирования нажать на кнопку «Сохранить», после чего данные исследования будут обновлены. Для открытия обследования необходимо дважды кликнуть левой кнопкой мыши по выбранному исследованию из списка в проводнике.

Во время работы с исследованием может потребоваться сохранить текущие изменения. Есть два варианта сохранения изменений:

- ручной;
- при закрытии обследования.

Для ручного сохранения изменений необходимо нажать кнопку «Сохранить» на панели инструментов или воспользоваться сочетанием клавиш CTRL+S.

Для сохранения изменений при закрытии обследования необходимо после завершения работы с обследованием и нажатия кнопки «Проводник» нажать кнопку "Сохранить" в появившемся диалоговом окне.

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		46

Сохраненные исследования легко найти. При вводе текста в поле поиска результаты сразу появятся в списке (рисунок 16). Для поиска нужного исследования нужно в проводнике в поле поиска ввести текстовый запрос. В списке отобразятся исследования, удовлетворяющие условиям запроса.

+ Создать							Петр
Имя пациента	Дата исследования	Пол пациента	Дата рождения	Описание	ID исследования	Размер	
Петров М.А.	22-02-2024	9:41 Женский	15-08-1978 , 46 лет	Рентген поясничного от...	12123125512	62.3 МБ	

Рисунок 16 — Поиск обследований

Сортировка обследований (рисунок 17) позволяет упорядочить список исследований по нескольким критериям: по имени пациента, дате проведения и идентификатору исследования (ID). Выбрав нужный параметр в меню сортировки, можно упорядочить список в порядке возрастания или убывания.

+ Создать							
Имя пациента	Дат	По имени			Дата рождения		
Матросов Александр	25-	✓ По дате исследования ↓			11-01-1996 , 28 лет		
Иванов Михаил	24-02-2024	17:25	Мужской		20-05-1989 , 35 лет		
Петров М.А.	22-02-2024	9:41	Женский		15-08-1978 , 46 лет		

Рисунок 17 — Сортировка обследований

Удаление обследований позволяет удалить одно или несколько выбранных обследований. Для этого необходимо выбрать обследование, кликнув по нему левой кнопкой мыши. Для удаления нескольких обследований следует зажать клавишу CTRL и выбрать те, которые нужно удалить. Далее нажать кнопку удаления в панели инструментов проводника. В появившемся диалоговом окне нажать кнопку «Удалить» для подтверждения удаления.

4.6 Методика испытаний программы

Ручное тестирование предполагает проверку всех реализованных функций на предмет соответствия их техническому заданию.

После запуска приложения будет открыта главная форма — список обследований (рисунок В.1). Необходимо создать исследование (рисунок В.2). Для этого надо заполнить данные о пациенте и описание исследования. После этого нажать на кнопку Создать. Затем необходимо открыть исследование, нажав на имя пациента. Откроется окно исследования (рисунок В.3). В пустое исследование необходимо добавить снимки: для этого можно воспользоваться меню Файл, пункт «Открыть файл DICOM». Для тестирования необходимо выбрать файл sample.dcm, который распространяется вместе с программой. После выбора файла и нажатия кнопки открыть он будет добавлен в обследование (рисунок В.4). Затем необходимо протестировать инструменты с вкладки Редактирование: необходимо последовательно нажимать на кнопки первого ряда (кроме крайней правой) и второго. Далее провести ползунки Улучшение контраста, Улучшение краев и Уменьшение шума в крайнее правое положение. В результате на экране должно быть представлено изображение, как показано на рисунке В.5. Далее необходимо нажать на кнопку «Вернуть к оригиналу» и протестировать графические инструменты: линейку, угол, измерительный прямоугольник. Необходимо расположить инструменты как показано на рисунке В.6. Если в результате измерений получились такие же данные, инструменты работают корректно. Затем необходимо последователь удалить все инструменты с помощью кнопки Delete, далее нажать на крайнюю правую кнопку в первом ряду. Будет показан диалог обрезки. Необходимо выбрать нужную область изображения и нажать на кнопку «Обрезать» (рисунок В.7). После нажать на кнопку «Продолжить» (рисунок В.8) и убедиться, что изображение было обработано успешно. Далее необходимо протестировать кнопки «+» и «-»: при нажатии на них изображение должно увеличиваться и уменьшаться. На рисунке В.9 изображение было увеличено. Затем необходимо протестировать контекстное

					ПГУ1.090501.13.001 ПЗ	Лист
						48
Изм.	Лист	№ докум.	Подпись	Дата		

меню снимка. Для этого необходимо нажать на кнопку с тремя точками рядом с миниатюрой снимка. При нажатии откроется контекстное меню (рисунок В.10). Необходимо нажать на кнопку «Дублировать». Снимок будет продублирован (рисунок В.11). После этого нажать на кнопку «Экспорт изображения», предварительно вызвав контекстное меню. Откроется диалог, в котором необходимо ввести имя файла и выбрать формат файла (рисунок В.12). После необходимо нажать на кнопку «Сохранить». Далее необходимо удалить произвольный снимок в обследовании, нажав на кнопку «Удалить» контекстного меню. В появившемся окне необходимо подтвердить свое действие (рисунок В.13). Затем необходимо протестировать кнопку главного меню (рисунок В.14). В открывшемся меню (рисунок В.15) необходимо выбрать пункт «О программе» и убедиться, что откроется окно, показанное на рисунке В.16. Далее в подменю «Вид» необходимо отключить отображение панели миниатюр, панели состояния и теги DICOM. Для этого надо нажать на каждый из пунктов подменю. В результате окно исследование должно принять вид, показанный на рисунке В.17. Для получения снимка необходимо перейти на вкладку исследование (рисунок В.18), предварительно подключив рентгеновский моноблок и плоскопанельный детектор. Получение снимка необходимо осуществлять строго в специально отведенном помещении под контролем сертифицированного специалиста (радиолога). Ввести параметры съемки: напряжение рентгеновской трубки, ток и время экспозиции и провести ползунок «Получить снимок». Через некоторое время снимок будет получен и должен появиться на панели миниатюр. Для формирования врачебного заключения необходимо нажать на кнопку (рисунок В.19). В появившемся окне заполнить информацию о патологиях, BIRADS, заключении, после чего нажать кнопку «ОК». Откроется сформированный документ (рисунок В.20). После этого необходимо вернуться на экран выбора обследования, нажав кнопку «Проводник» и затем кнопку редактирования обследования. Изменить поля на свое усмотрение и выйти, нажав кнопку «Сохранить» (рисунок В.21). Затем необходимо протестировать смену типа сортировки. Для этого необходимо нажать на соответствующую кнопку и

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		49

после этого выбрать порядок сортировки (рисунок В.22). Для удаления обследования необходимо нажать кнопку «Удалить» и подтвердить свой выбор в диалоговом окне (рисунок В.23).

					ПГУ1.090501.13.001 ПЗ	Лист
						50
Изм.	Лист	№ докум.	Подпись	Дата		

5 Оценка затрат на разработку и внедрение приложения для анализа и обработки рентгеновских снимков

5.1 Организация разработки приложения

Разработка приложения осуществлена в дистанционном формате и с использованием оборудования сотрудников, что позволило сократить затраты на аренду помещения, закупку и обслуживание техники.

По соглашению с работниками каждый месяц им выплачивалась компенсация в 2000 рублей за использование своего оборудования в рабочих целях. Компенсация выплачивалась только сотрудникам в штате.

Разработка приложения для анализа и обработки рентгеновских снимков осуществлялась командой разработчиков, представленных в таблице 9.

Таблица 9 – Состав коллектива разработчиков

Специалист	Количество, человек	Описание занятости
Руководитель проекта	1	– управление проектом; – набор команды исполнителей; – управление ресурсами проекта; – контроль соблюдения требований заказчика; – приемка и внедрение проекта; – анализ предметной области.

Специалист	Количество, человек	Описание занятости
Программист	2	<ul style="list-style-type: none"> – создание базы данных; – написание, отладка и тестирование; программного кода – разработка и утверждение программной документации; – приемка и внедрение программного кода.
Дизайнер	1	<ul style="list-style-type: none"> – разработка интерфейса приложения; – разработка оформления пользовательской документации.

Данные по основной заработной плате разработчиков приложения и их количественному составу приведены в таблице 10. Важно отметить, что в современной практике разработки приложений дизайн часто передается на аутсорсинг. В этом случае дизайнер работает по гражданско-правовому договору (ГПХ). Налогообложение в этом случае аналогично налогообложению заработной платы. Данные по выплатам по ГПХ представлены в таблице 11.

Таблица 10 – Сведения о заработной плате

Специалист	Количество, человек	Сумма заработной платы, рублей
Руководитель проекта	1	100000
Программист	2	80000

Таблица 11 – Сведения о выплатах по ГПХ

Специалист	Количество, человек	Сумма выплаты, рублей
Дизайнер	1	100000

5.2 Расчет себестоимости разработки приложения для анализа и обработки рентгеновских снимков

Себестоимость разработки приложения включает следующие расходы:

- на заработную плату;
- на выплаты по договорам ГПХ;
- на компенсации сотрудникам;
- на обслуживание программы.

Затраты на заработную плату включают затраты на основную заработную плату разработчиков программного приложения и отчисления на страховые взносы. Аналогично с выплатами по договорам ГПХ.

С учетом выходных дней при 5-дневной рабочей неделе количество рабочих дней в месяце в среднем составляет 21 день. На разработку всего проекта необходимо 6 месяцев.

Таким образом, затраты на заработную плату разработчиков составят 260000 рублей в месяц, 1560000 рублей за весь период создания проекта.

Отчисления на страховые взносы:

- обязательные отчисления по нормам, установленным законодательством государственного социального страхования;
- в Фонд пенсионного и социального страхования РФ.

От затрат на оплату труда работников, включаемых в себестоимость продукции (программного обеспечения), по элементу «Затраты на оплату труда» (кроме тех видов оплаты, на которые страховые взносы не начисляются).

Отчисления на страховые взносы производится согласно «Налоговому кодексу Российской Федерации» от 14.07.2022 № 239-ФЗ. Действующие тарифы на страховые взносы работодателей установлены статьей 425 НК РФ и составляют 30%.

Так как компания осуществляет деятельность в области IT, то она может претендовать на льготную ставку (подп. 3 п. 1 ст. 427 НК РФ) равную 7,6%.

С 1 января 2025 года с выплат до предельной базы (2,759 миллиона рублей) уплачиваются взносы – 7,6%. Зарплаты сотрудников позволяют применять льготный тариф на всю сумму выплат.

Льготный тариф так же распространяется на сотрудников, работающих по гражданско-правовому договору, что подтверждено письмом Минфина от 23.12.2016 № 03-04-12/77644.

Затраты на заработную плату и отчисления на страховые взносы представлены в таблице 12.

Таблица 12 – Затраты на заработную плату

Сотрудник	Заработная плата за месяц, рублей	Общая сумма отчислений за месяц, рублей	Всего за 1 месяц, рублей	Всего за 6 месяцев, рублей
Руководитель проекта	100000	7600	107600	645600
Программист	80000	6080	86080	516480
Программист	80000	6080	86080	516480

Затраты на выплаты по договору ГПХ и отчисления на страховые взносы по этому договору представлены в таблице 13.

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		54

Таблица 13 – Затраты на выплаты по договорам ГПХ

Сотрудник	Общая сумма выплаты, рублей	Общая сумма отчислений, рублей
Дизайнер	100000	7600

Затраты на заработную плату (с учетом отчислений) команды проекта по созданию приложения составят 1678560 рублей из которых общая сумма отчислений на страховые взносы разработчиков составит 118560 рублей, а заработная плата составит 1560000 рублей.

Затраты на выплату компенсаций сотрудникам:

$$2000 * 3 * 6 = 36000 \text{ руб.}$$

Себестоимость разработки определяется как сумма затрат, включающая все статьи затрат на реализацию проекта, представленное в таблице 14.

Таблица 14 – Общая сумма затрат

Статья затрат	Сумма отчислений, рублей
Затраты на заработную плату (включая взносы)	1678560
Затраты на выплаты по договорам ГПХ (включая взносы)	107600
Затраты на компенсации сотрудникам	36000
Итого:	1822160

Таким образом, сумма затрат на разработку проекта составит 1822160 рублей и будет соответствовать себестоимости приложения.

5.3 Оценка эффективности приложения

Необходимость и целесообразность реализации инвестиционных проектов осуществляется путем оценки их экономической эффективности.

Экономическая эффективность инвестиционных проектов характеризуется системой показателей, отражающих соотношение результатов (притока, поступлений, дохода) и затрат (оттока, выплат) от операционной, инвестиционной и финансовой деятельности.

За счет внедрения и использования приложения для анализа и обработки рентгеновских снимков также ожидаются следующие неисчисляемые эффекты:

- увеличение скорости работы врачей;
- упрощение постановки диагноза;
- улучшение качества постановки диагноза;
- снижение радиационной нагрузки на пациентов.

5.4 Анализ рисков при разработке и внедрении приложения

Риски взаимодействия между заказчиком и исполнителем – это риски, связанные с отсутствием коммуникации между исполнителем и заказчиком или их представителями.

Риски планирования – это риски, связанные с отсутствием навыков планирования сроков выполняемых работ, как менеджером, так и исполнителями. Часто слишком мало времени закладывается на этап тестирования и исправления ошибок в исходном коде, предпроектного обследования. Для минимизации риска, проектный руководитель следил за четким выполнением сроков на каждом этапе проекта.

Недостаточное обсуждение задач или архитектуры может негативно сказаться на разрабатываемом приложении. Для уменьшения последствий от таких рисков на начальном этапе проекта разработано соглашение и

обозначена в договоре частота проведения встреч по проекту, куратора проекта со стороны заказчика и исполнителя, а также состав рабочей группы.

Функциональные риски – это риски, связанные с неверным выполнением функций, возложенных на систему. Для минимизации данных рисков необходим тщательный анализ требований заказчика, разработка моделей и привлечение к работе грамотных специалистов.

Риск смены сотрудников возникает, когда проект покидают ключевые сотрудники, которые владеют максимально полной информацией о проекте. В некоторых случаях уход данных сотрудников может полностью остановить разработку проекта. Для того чтобы минимизировать последствия от ухода ключевых сотрудников проекта, руководитель должен поддерживать высокий уровень сотрудничества и обмена информацией в коллективе, планировать работу сотрудников так, чтобы они могли заменить друг друга, а также выстраивает систему внутреннего обучения.

Риски хищения исходного кода возникают, если при уходе разработчики забирают созданный ими проект.

Дополнительной мотивацией для всех сотрудников оставаться на своем рабочем месте является премирование, которое может быть материальным и нематериальным.

Риск привлечения к ответственности, в случае использования компанией нелицензионного ПО, это может удешевить себестоимость проекта, но и может понести последствия. Для минимизации данного риска, каждая компания должна закупать только лицензионное программное обеспечение.

При предпроектном анализе в обязательном порядке должны быть раскрыты все риски, а также пути их минимизации. В конце должен быть составлен обобщающий отчет по всем рискам.

					ПГУ1.090501.13.001 ПЗ	Лист
						57
Изм.	Лист	№ докум.	Подпись	Дата		

6 Общие принципы обеспечения безопасности на рабочем месте оператора приложения для анализа и обработки рентгеновских снимков

6.1 Актуальность вопросов безопасности человека на рабочем месте

С каждым годом человек все больше исследует и использует IT технологии в своей жизни. Это стало возможно благодаря распространению ПЭВМ (персональных электронных вычислительных машин). Они стали употребляться почти на каждом рабочем месте, именно поэтому необходимо помнить про безопасность оператора ПЭВМ, сохранении его здоровья и работоспособности.

Использование ЭВМ связано с распознаванием печатного и рукописного текста, анализ изображений на экране, долгим нахождением в одной позе, и др., что может пагубно сказаться на здоровье человека, если не придерживаться нормами использования ЭВМ.

На основании ГОСТ 12.0.003-74 [20] вредные производственные факторы подразделяются на 4 категории:

- физические, к ним относятся недостаточная освещенность рабочего места, повышенная яркость света, отклонения от нормы температуры воздуха, высокая влажность воздуха, повышенный уровень шума, и др.;
- психофизиологические, к ним относятся нервно-психические и физические перегрузки, перенапряжение ушей и глаз;
- биологические, к ним относятся микроорганизмы и макроорганизмы, которые могут вызвать различные заболевания;
- химические, к ним относятся превышения в рабочей зоне аммиака, фенола, озона, оксида углерода и др.

6.2 Техника безопасности при работе с электроприборами

Персональная электронно-вычислительная машина – это электроприбор, следовательно, работа с ним требует знаний правил техники безопасности:

- при работе с электрооборудованием следует использовать заземленные розетки;
- после выключения электроприборов из сети, нельзя тянуть их за шнур и находиться рядом с мокрыми руками;
- неисправность у прибора нельзя чинить до момента отключения от сети;
- до включения прибора в сеть необходимо проверить сохранность шнура (наличие заломов и повреждения изоляции);
- не оставлять включенные приборы без присмотра.

В состав ПЭВМ входят монитор, блок питания, процессор – все они создают электромагнитные поля. Такие поля создают еще распределительные щиты, электропроводка, линии электропередач.

6.3 Влияние ПЭВМ на физиологию человека

Работа за компьютером подразумевает длительное напряжение на зрительные органы и опорно-двигательный аппарат. Если не контролировать время пребывания за ЭВМ, то это может привести к нарушению зрения, искривлению позвоночника, а также нарушению кровообращения в организме.

Нагрузка на зрительный аппарат может вызвать нарушения в работе ЦНС (центральной нервной системы). При длительной работе за компьютером снижается фокусировка и острота зрения, увеличивается чувствительность глаз, они могут слезиться и краснеть.

Длительное нахождения человека в одной позе приводит к быстрому развитию утомляемости.

При работе за компьютером стоит обратить внимание на:

- высоту рабочей зоны и стула;
- наличие опорной спинки и подлокотников;

					ПГУ1.090501.13.001 ПЗ	Лист
						59
Изм.	Лист	№ докум.	Подпись	Дата		

- угол наклона головы при работе;
- угол наклона экрана компьютера.

Огромное влияние на здоровье человека оказывает уровень шума в помещении. Согласно СанПиН 2.2.3670-20 "Санитарно-эпидемиологические требования к условиям труда" [21], эквивалентный уровень шума не должен превышать 50 дБА. Если шум свыше нормы будет длительно присутствовать на рабочем месте сотрудника, то будут ухудшены его условия труда и нанесен вред здоровью. А именно:

- снижается концентрация внимания;
- повышается усталость;
- снижается работоспособность сотрудника.

Для того, чтобы снизить уровень шума в помещении используется шумоизоляция.

6.4 Техника безопасности при работе с персональным компьютером

На основе типовой инструкции по охране труда при работе на ПК (персональном компьютере) ТОИ Р-45-084-01 [22], которая была утверждена Приказом Минсвязи Российской Федерации №162 от 02.07.2001, деятельность на ЭВМ делится на 3 группы:

- группа А – чтение информации с монитора подразумевает два перерыва по 15 минут, без учета обеденного перерыва;
- группа Б – набор текста подразумевает перерывы по 10 минут раз в час;
- группа В – творческий труд подразумевает перерывы по 15 минут раз в час.

Выделенное время рекомендуется уделять физической нагрузке, такой как зарядка или гимнастика для глаз.

На основании всего вышесказанного, можно сделать вывод, что для сохранения здоровья сотрудников и ограничения от негативного влияния

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		60

ПЭВМ необходимо использовать только сертифицированную и качественную технику. Обеспечивать сотрудникам рациональный режим труда и перерывов.

Необходимо отметить, что сертифицированная техника не является полной гарантией безопасности условий труда, необходимо регулярно проводить проверку всей используемой техники.

6.5 Техника безопасности при работе с рентгеновским оборудованием

При получении рентгеновского снимка используется следующее оборудование: рентгеновский аппарат и плоскопанельный детектор рентгеновского излучения. Плоскопанельный детектор регистрирует излучение и сам по себе не может нанести вред здоровью человека.

В отличие от рентгеновского аппарата, работа с которым должна производиться в отдельном оборудованном помещении под управлением квалифицированного персонала. Работа с рентгеновским аппаратом регулируется приказом Министерства здравоохранения РФ [23] и стандартом ГОСТ Р МЭК 61223-2-11-2001 [24].

					ПГУ1.090501.13.001 ПЗ	Лист
						61
Изм.	Лист	№ докум.	Подпись	Дата		

Заключение

В ходе выполнения дипломного проекта была изучена предметная область, а именно работа врачей-рентгенологов. На основе ее анализа было сформулировано техническое задание для разработки приложения для анализа и обработки рентгеновских снимков. Далее были выбраны средства разработки, с помощью которых было разработано приложение.

Приложения для анализа и обработки рентгеновских снимков реализовано в соответствии с техническим заданием.

В разделе экономики была произведена оценка затрат на разработку и внедрение приложения.

В разделе безопасность жизнедеятельности были проанализированы и описаны общие принципы обеспечения безопасности на рабочем месте оператора программного приложения контроля выполнения задач.

Задание на дипломное проектирование выполнено в полном объеме.

					ПГУ1.090501.13.001 ПЗ	Лист
						62
Изм.	Лист	№ докум.	Подпись	Дата		

Список использованных источников

1. Oleg S. Pianykh Digital Imaging and Communications in Medicine (DICOM): A Practical Introduction and Survival Guide [Текст] / Oleg S. Pianykh — 2-е изд. — Берлин: Springer, 2009 — 417 с.
2. Российский MedTech взял курс на повышение / [Электронный ресурс] // Коммерсант : [сайт]. — URL: <https://www.kommersant.ru/doc/6616413> (дата обращения: 02.04.2025).
3. Клетте Рейнхард Компьютерное зрение. Теория и алгоритмы [Текст] / Клетте Рейнхард — . — Москва: ДМК Пресс, 2019 — 506 с.
4. Искусственный интеллект в столичной медицине будет использоваться более чем в 50 направлениях лучевых исследований / [Электронный ресурс] // mos.ru : [сайт]. — URL: <https://www.mos.ru/mayor/themes/8991050/> (дата обращения: 02.04.2025).
5. Эксперимент по использованию инновационных технологий в области компьютерного зрения для анализа медицинских изображений и дальнейшего применения этих технологий в системе здравоохранения. / [Электронный ресурс] // mosmed.ai : [сайт]. — URL: <https://mosmed.ai/ai/> (дата обращения: 03.04.2025).
6. Врачам диагностировали нехватку / [Электронный ресурс] // Коммерсант : [сайт]. — URL: <https://www.kommersant.ru/doc/6691140> (дата обращения: 06.04.2025).
7. Kim, J., Harper, A., McCormack, V Global patterns and trends in breast cancer incidence and mortality across 185 countries [Текст] / Kim, J., Harper, A., McCormack, V // Nature. — 2025. — № 31. — С. 1154–1162.
8. Д. А. Рябчиков, А. В. Петровский Рак молочной железы : руководство для врачей [Текст] / Д. А. Рябчиков, А. В. Петровский — . — Москва: ГЭОТАР-Медиа, 2023 — 400 с.
9. Шумакова, Т. А., Солнцева, И. А., Сафронова, О. Б., Савелло, В. Е., Серебрякова, С. В., Маков, М. Ю. Применение международной классификации BI-RADS в маммологической практике [Текст] / Т. А. Шумакова, И. А.

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		63

Солнцева, О. Б. Сафронова, В. Е. Савелло, С. В. Серебрякова, М. Ю. Маков — 4-е изд. — Санкт-Петербург: Элби, 2023 — 208 с.

10. Цифровой рентген: преимущества перед пленкой / [Электронный ресурс] // : [сайт]. — URL: <https://primamed-dv.ru/article/cifrovoj-rentgen-preimushchestva-pered-plenкой> (дата обращения: 04.05.2025).

11. Методические рекомендации по использованию системы BI-RADS при маммографическом обследовании / Методические рекомендации под редакцией д.м.н., профессора, член-корреспондента РАН, Заслуженного деятеля науки РФ, А. Ю. Васильева. — Москва. — 2017. — 23 с.

12. Страуструп Бьерн Язык программирования C++ [Текст] / Страуструп Бьерн — 4-е изд.. — Москва: Бином, 2024 — 1216 с.

13. TIOBE Index for May 2025 / [Электронный ресурс] // TIOBE : [сайт]. — URL: <https://www.tiobe.com/tiobe-index/> (дата обращения: 11.05.2025).

14. The Top Programming Languages 2024 / [Электронный ресурс] // IEEE Spectrum : [сайт]. — URL: <https://spectrum.ieee.org/top-programming-languages-2024> (дата обращения: 12.05.2025).

15. Брайан У. Керниган; Алан А. Донован Язык программирования Go [Текст] / Брайан У. Керниган; Алан А. Донован — . — Москва: Диалектика, 2020 — 432 с.

16. StarUML2 documentation / [Электронный ресурс] // StarUML : [сайт]. — URL: <https://staruml.readthedocs.io/en/latest/index.html> (дата обращения: 28.05.2025).

17. Kay Droessler, Hartmut Kay SQLite 3 - Einstieg in Die Datenbankwelt. [Текст] / Kay Droessler, Hartmut Kay — . — : Lulu Press, 2011 — 104 с.

18. Шлее Макс Qt 5.10. Профессиональное программирование на C++ [Текст] / Шлее Макс — . — Санкт-Петербург: БХВ, 2018 — 1072 с.

19. Свидзиньски Р. CMake для C++. Современный подход [Текст] / Свидзиньски Р. — . — Москва: ДМК Пресс, 2025 — 442 с.

20. ГОСТ 12.0.0003-74 ССБТ. Библиографическая запись. Опасные и вредные производственные факторы. Классификация (с Изменением № 1). Изд-во стандартов, 2002. — 5 с.

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		64

21. СанПиН 2.2.3670-20. Санитарно-эпидемиологические требования к условиям труда. Изд-во стандартов, 2012. – 47с.

22. Типовая инструкция по охране труда при работе на персональном компьютере. – 2001. – 3 с.

23. Приказ Министерства здравоохранения РФ от 9 июня 2020 г. N 560н "Об утверждении Правил проведения рентгенологических исследований" (с изменениями и дополнениями)

24. ГОСТ Р МЭК 61223-2-11-2001 Оценка и контроль эксплуатационных параметров рентгеновской аппаратуры в отделениях (кабинетах) рентгенодиагностики. Часть 2-11. Испытания на постоянство параметров. Аппараты для общей прямой рентгенографии.

					ПГУ1.090501.13.001 ПЗ	Лист
						65
Изм.	Лист	№ докум.	Подпись	Дата		

Приложение А

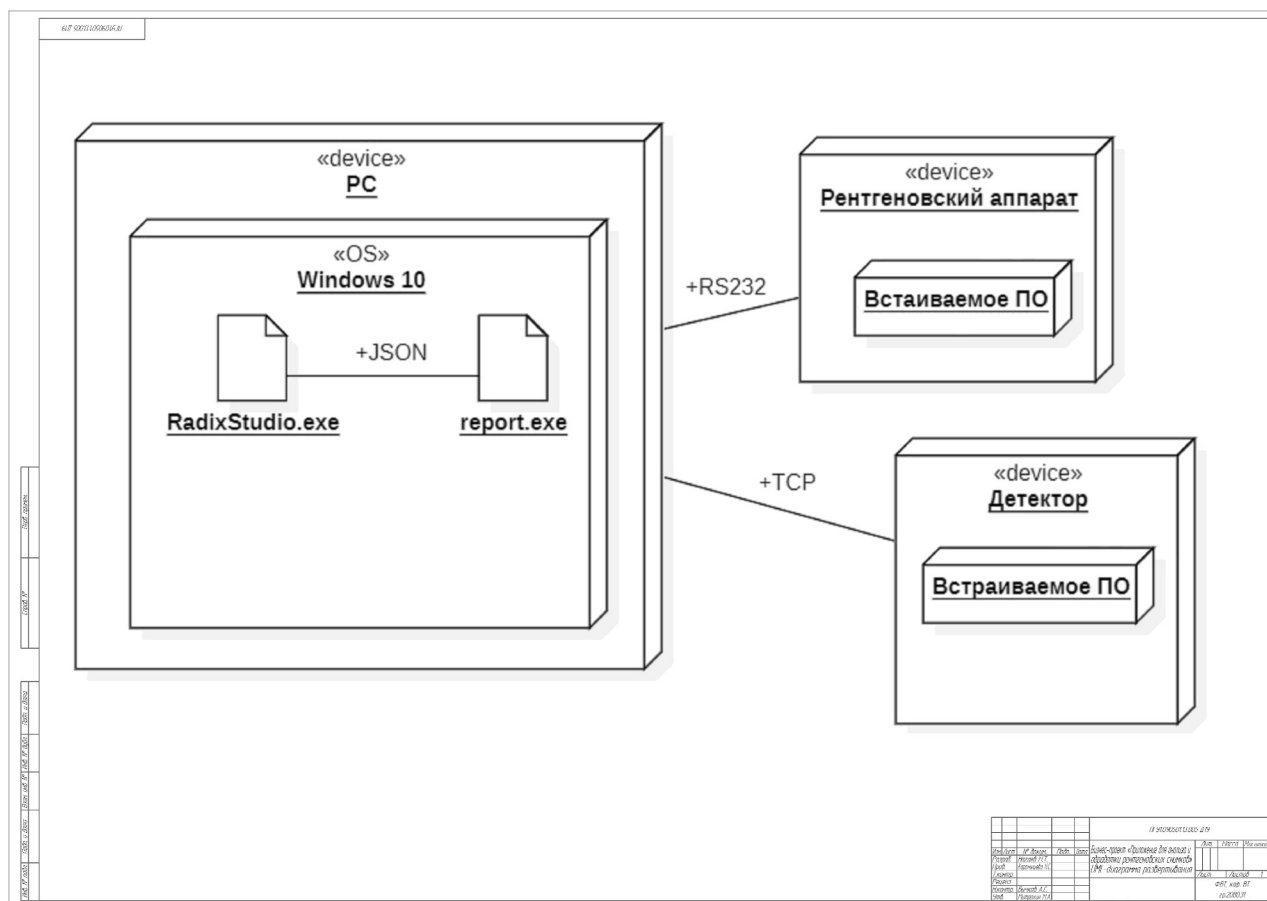
(обязательное)

Приложение для анализа и обработки снимков

UML-диаграммы

					ПГУ1.090501.13.001 ПЗ	Лист
						66
Изм.	Лист	№ докум.	Подпись	Дата		

Приложение А.1 – Диаграмма развертывания



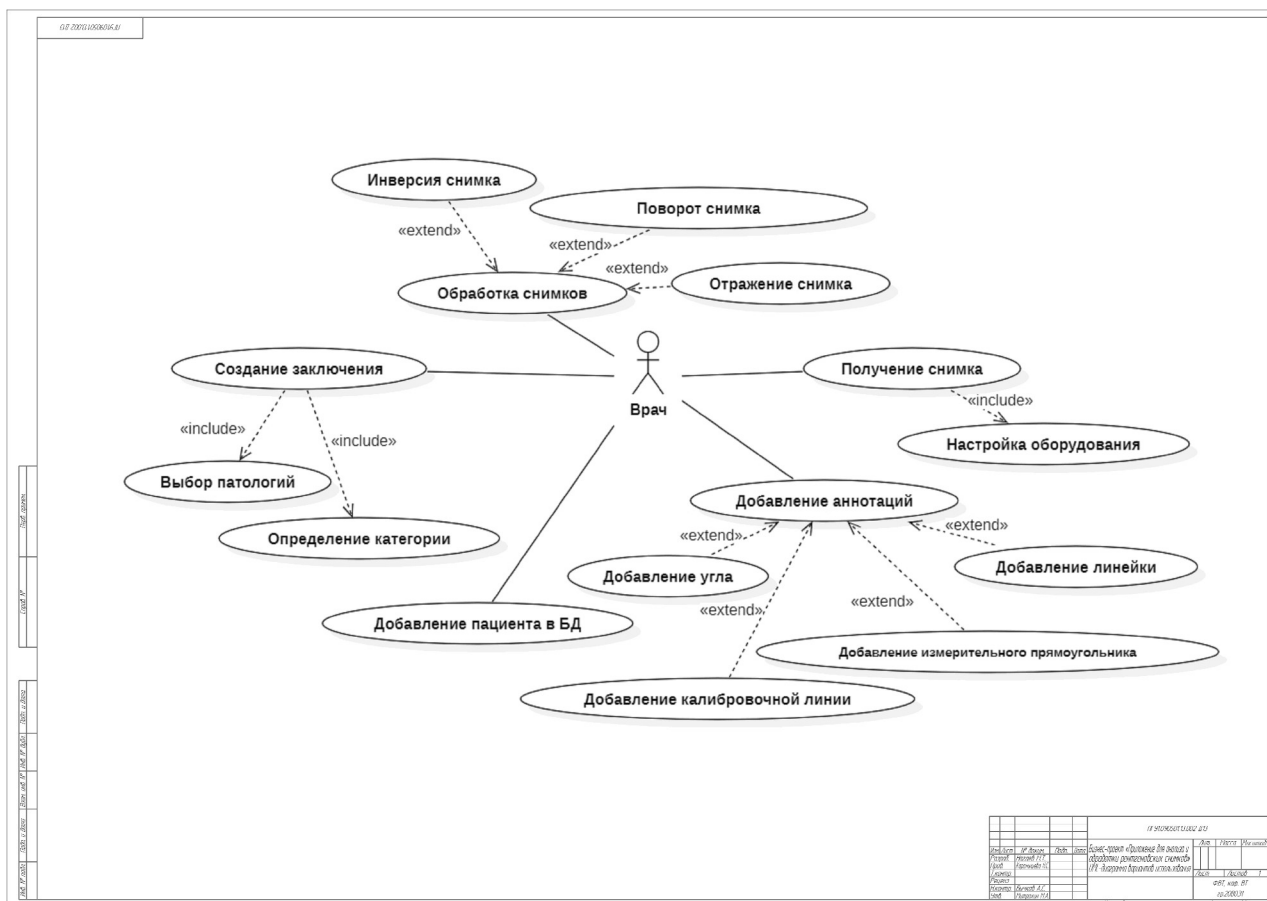
Приложение А.2 – Диаграмма вариантов использования

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ1.090501.13.001 ПЗ

Лист

67



Приложение А.3 – Логическая модель данных

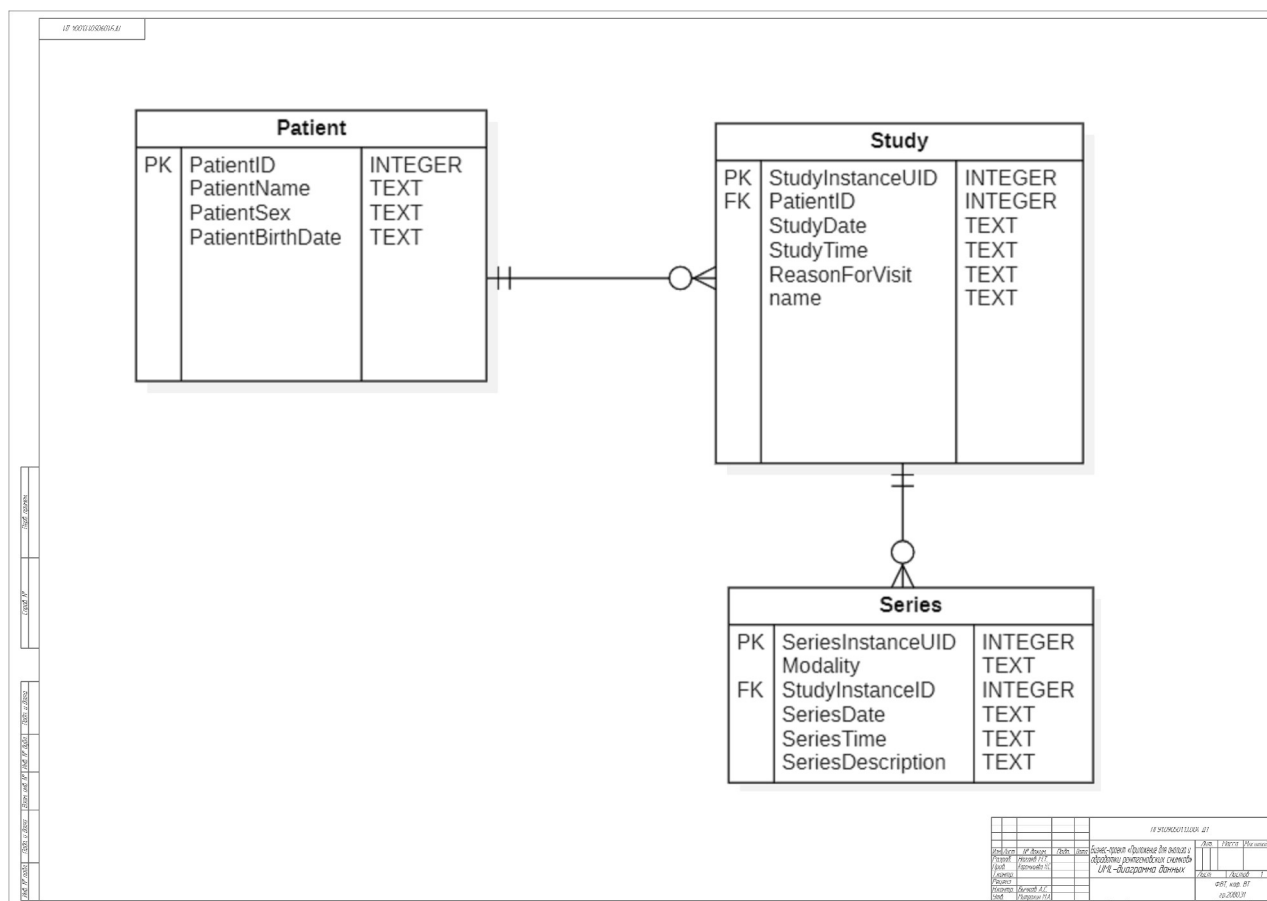
Изм.	Лист	№ докум.	Подпись	Дата

ПГУ1.090501.13.001 ПЗ

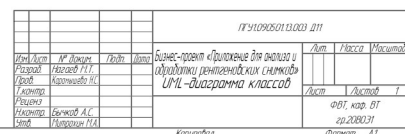
Лист

68

Приложение А.4 – Физическая модель данных



Приложение А.5 – Диаграмма классов



Приложение Б

(обязательное)

Приложение для анализа и обработки рентгеновских снимков

SQL-скрипт

					ПГУ1.090501.13.001 ПЗ	Лист
						72
Изм.	Лист	№ докум.	Подпись	Дата		

```
PRAGMA foreign_keys=on;
```

```
CREATE TABLE `Patient` (  
    `PatientID` INTEGER NOT NULL,  
    `PatientName` TEXT NOT NULL,  
    `PatientSex` TEXT NOT NULL,  
    `PatientBirthDate` TEXT NOT NULL,  
    `Flags` SMALLINT NOT NULL DEFAULT 0,  
    PRIMARY KEY (`PatientID`)  
);
```

```
CREATE TABLE `Study` (  
    `StudyInstanceUID` INTEGER NOT NULL,  
    `StudyID` TEXT,  
    `PatientID` INTEGER NOT NULL,  
    `AccessionNumber` TEXT NOT NULL DEFAULT '',  
    `ReasonForVisit` TEXT NOT NULL,  
    `dicomPath` TEXT NOT NULL,  
    `name` TEXT NOT NULL,  
    `Flags` SMALLINT NOT NULL DEFAULT 0,  
    PRIMARY KEY (`StudyInstanceUID`)  
);
```

```
CREATE TABLE `Series` (  
    `SeriesInstanceUID` INTEGER NOT NULL,  
    `Modality` TEXT NOT NULL DEFAULT 'DX',  
    `StudyInstanceID` INTEGER NOT NULL,  
    `SeriesDate` TEXT NOT NULL,  
    `SeriesTime` TEXT NOT NULL,
```

					ПГУ1.090501.13.001 ПЗ	Лист
						73
Изм.	Лист	№ докум.	Подпись	Дата		


```
`SeriesDescription` TEXT NOT NULL DEFAULT '',  
  
PRIMARY KEY (`SeriesInstanceUID`)  
  
FOREIGN KEY (StudyInstanceID) REFERENCES Study  
(StudyInstanceUID) ON DELETE CASCADE  
  
);
```

					ПГУ1.090501.13.001 ПЗ	Лист
						74
Изм.	Лист	№ докум.	Подпись	Дата		

Приложение В

(обязательное)

Приложение для анализа и обработки рентгеновских снимков

Результаты испытаний

					ПГУ1.090501.13.001 ПЗ	Лист
						75
Изм.	Лист	№ докум.	Подпись	Дата		

Радикс Студия

+

Создать

Поиск

Рисунок В.1 – Список обследований

Радикс Студия

+

Создать

Поиск

Новое исследование

Фамилия

Манжурин

Имя

Андрей

Отчество

Владимирович

Дата рождения

21042003

Пол

Мужской

Женский

Описание

Пара ваших личных пояснений, чтобы было проще вспомнить детали или другой врач смог быстро войти в курс дела.

СоздатьОтменить

Рисунок В.2 – Создание обследования

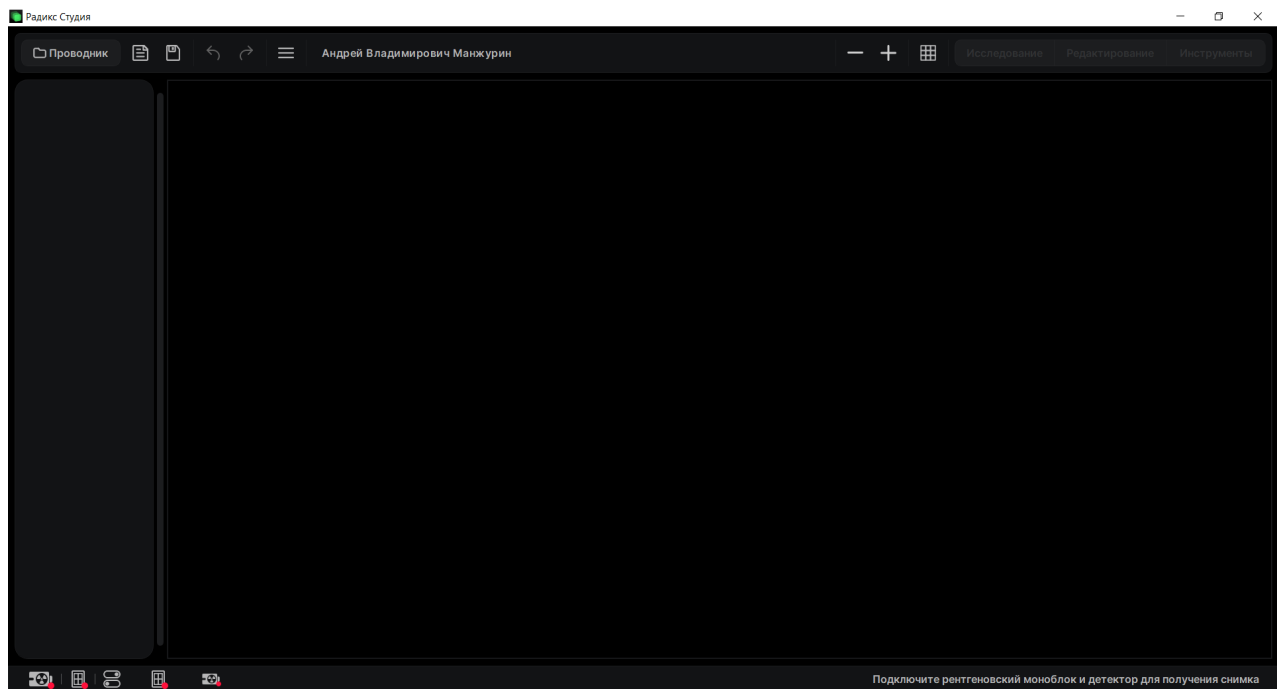


Рисунок В.3 – Экран обследования

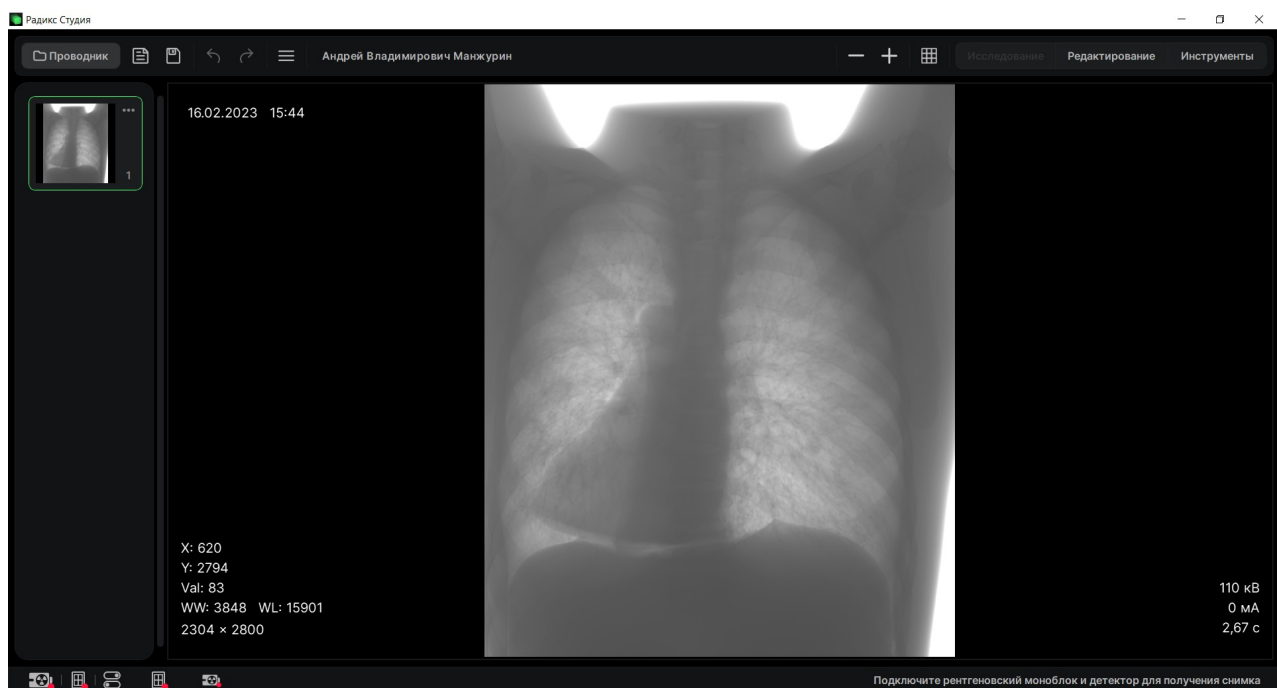


Рисунок В.4 – Добавление снимка

					ПГУ1.090501.13.001 ПЗ	Лист
						77
Изм.	Лист	№ докум.	Подпись	Дата		



Рисунок В.5 – Применение фильтров

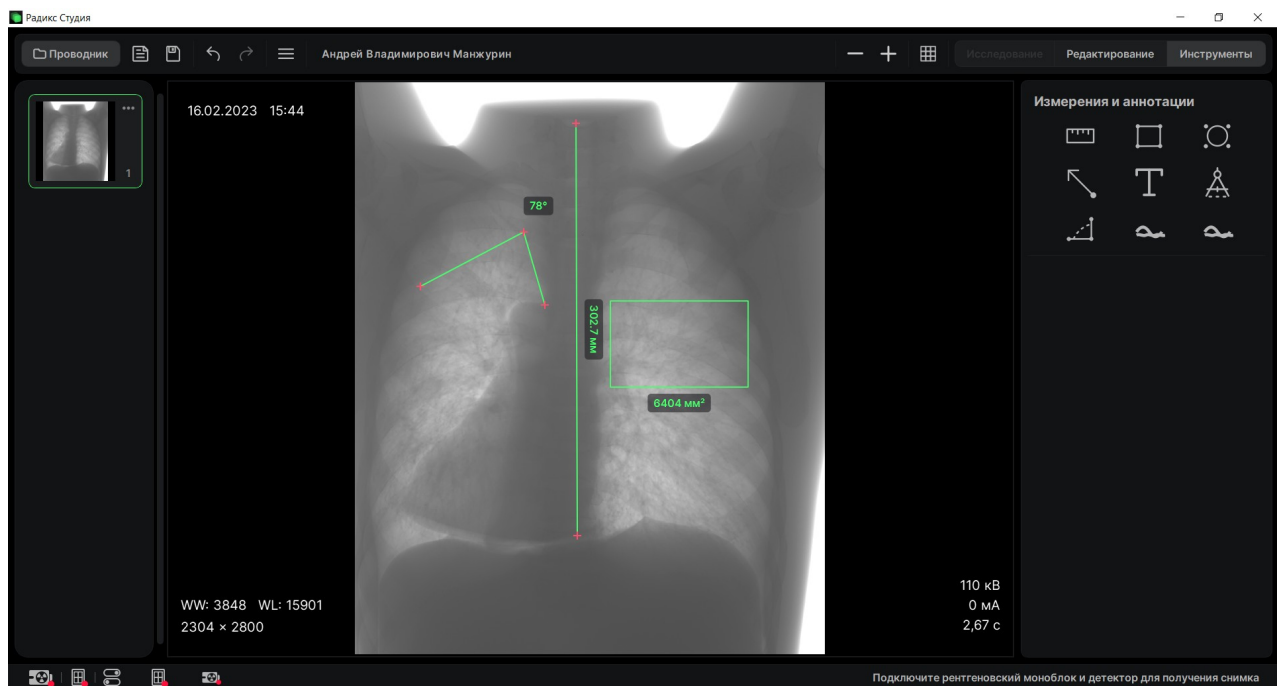


Рисунок В.6 – Графические инструменты

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		78



Рисунок В.7 – Кнопка обрезки снимка

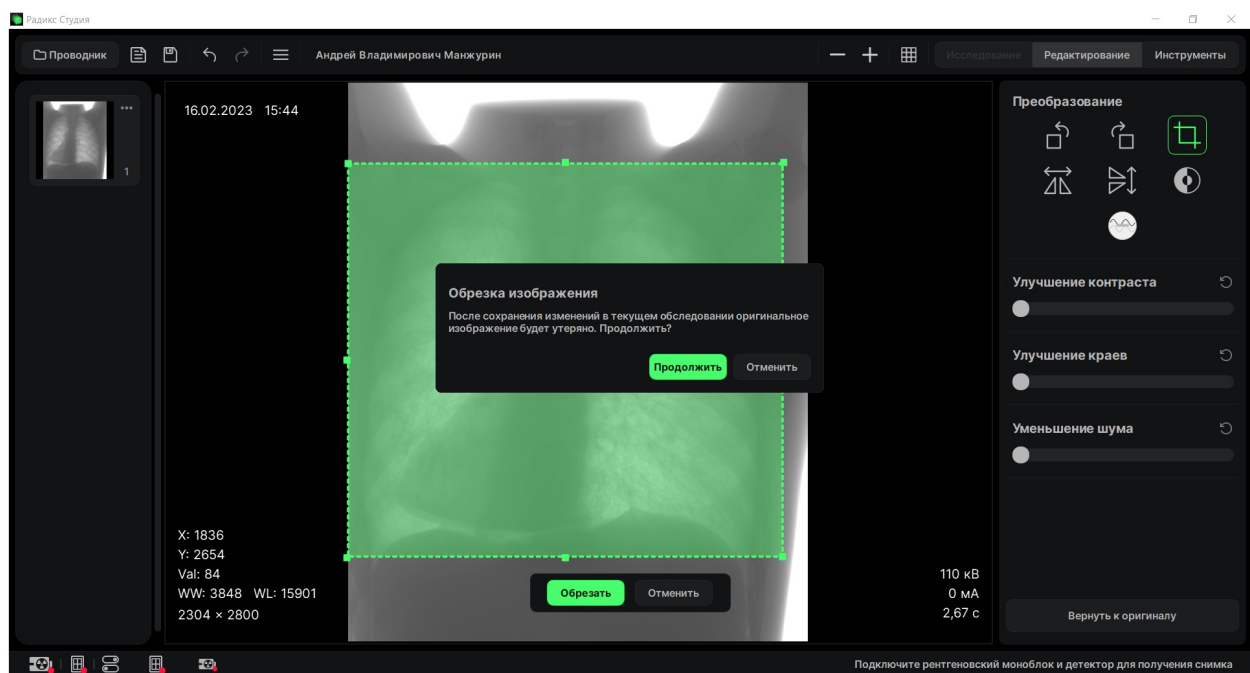


Рисунок В.8 – Подтверждение обрезки

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		79



Рисунок В.9 – Обрезанный снимок



Рисунок В.10 – Контекстное меню

					ПГУ1.090501.13.001 ПЗ	Лист
						80
Изм.	Лист	№ докум.	Подпись	Дата		



Рисунок В.11 – Дублирование снимка

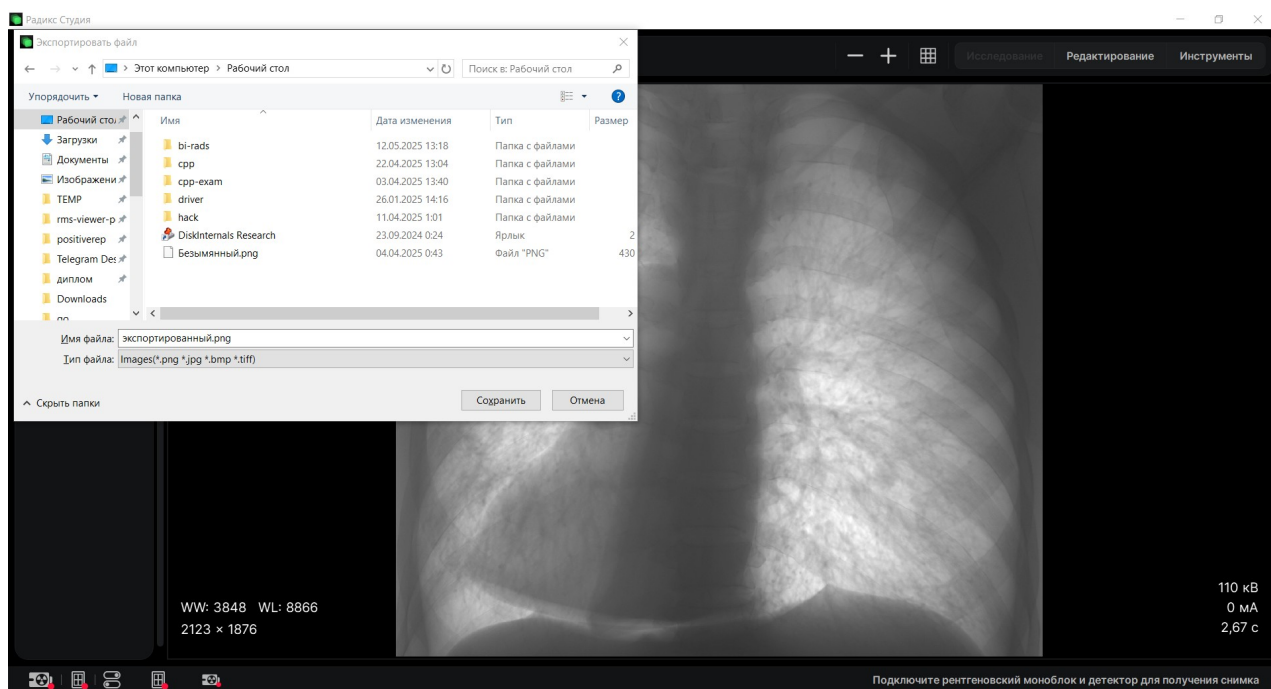


Рисунок В.12 – Экспорт снимка

					ПГУ1.090501.13.001 ПЗ	Лист
						81
Изм.	Лист	№ докум.	Подпись	Дата		

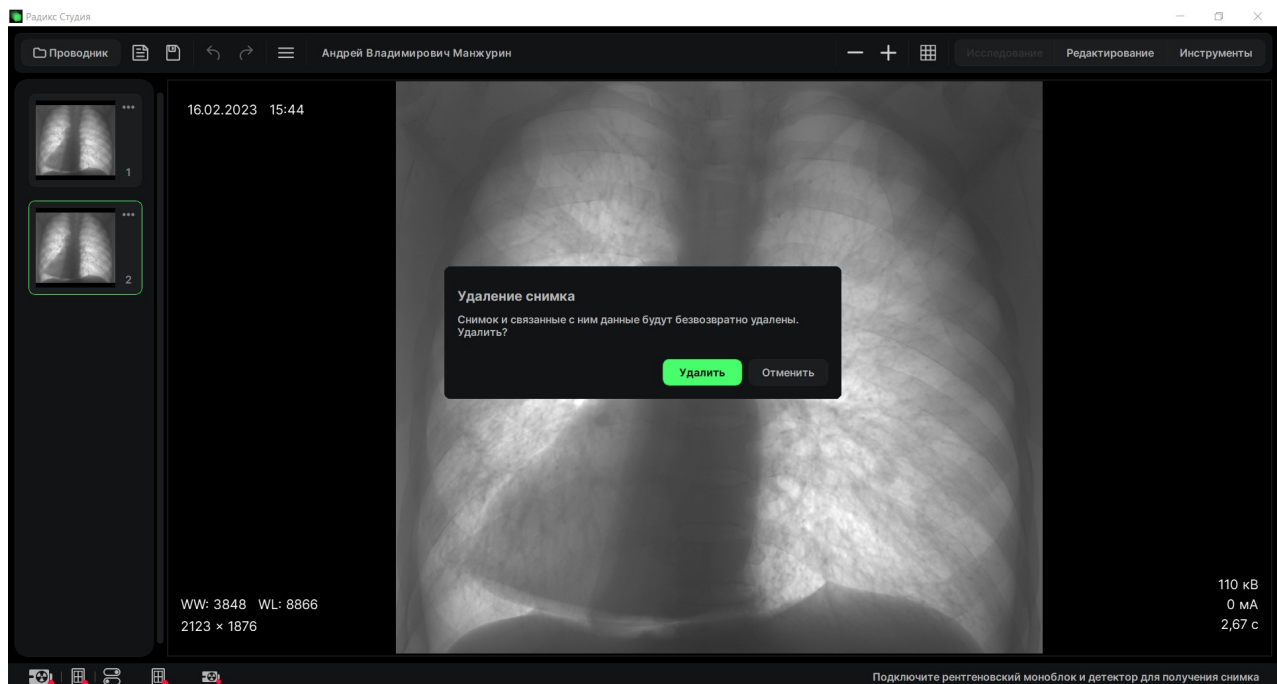


Рисунок В.13 – Удаление снимка



Рисунок В.14 – Кнопка главного меню

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		82

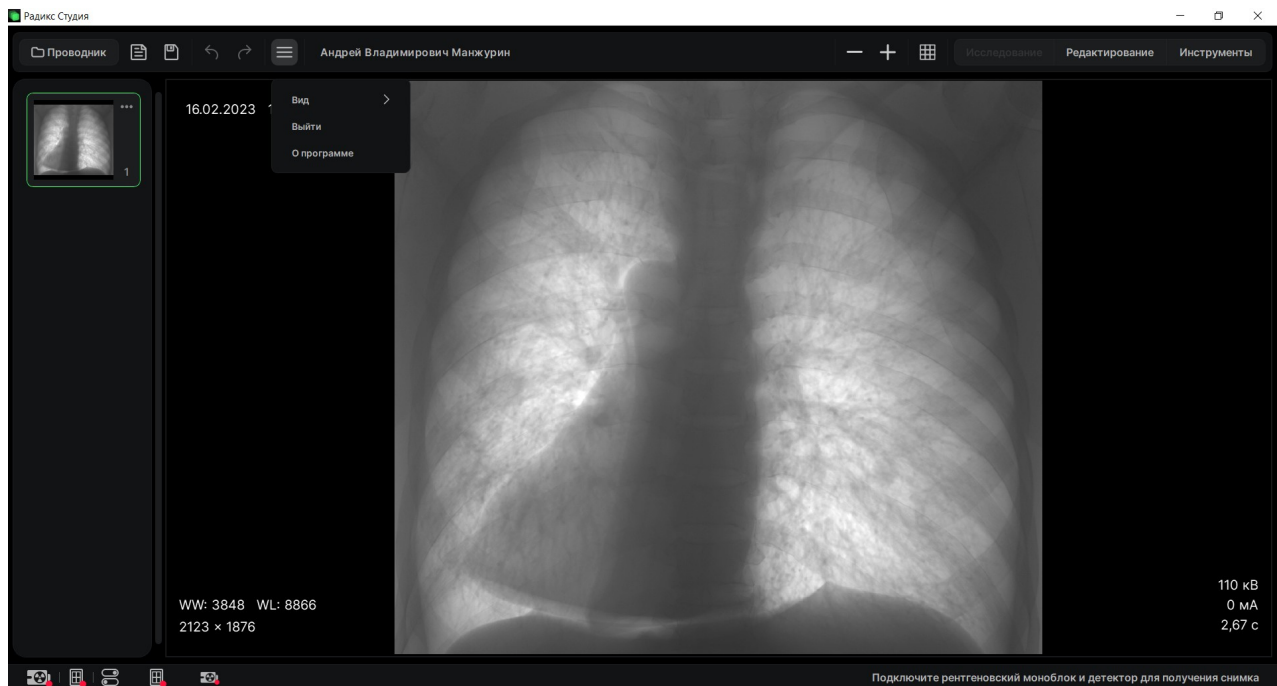


Рисунок В.15 – Главное меню

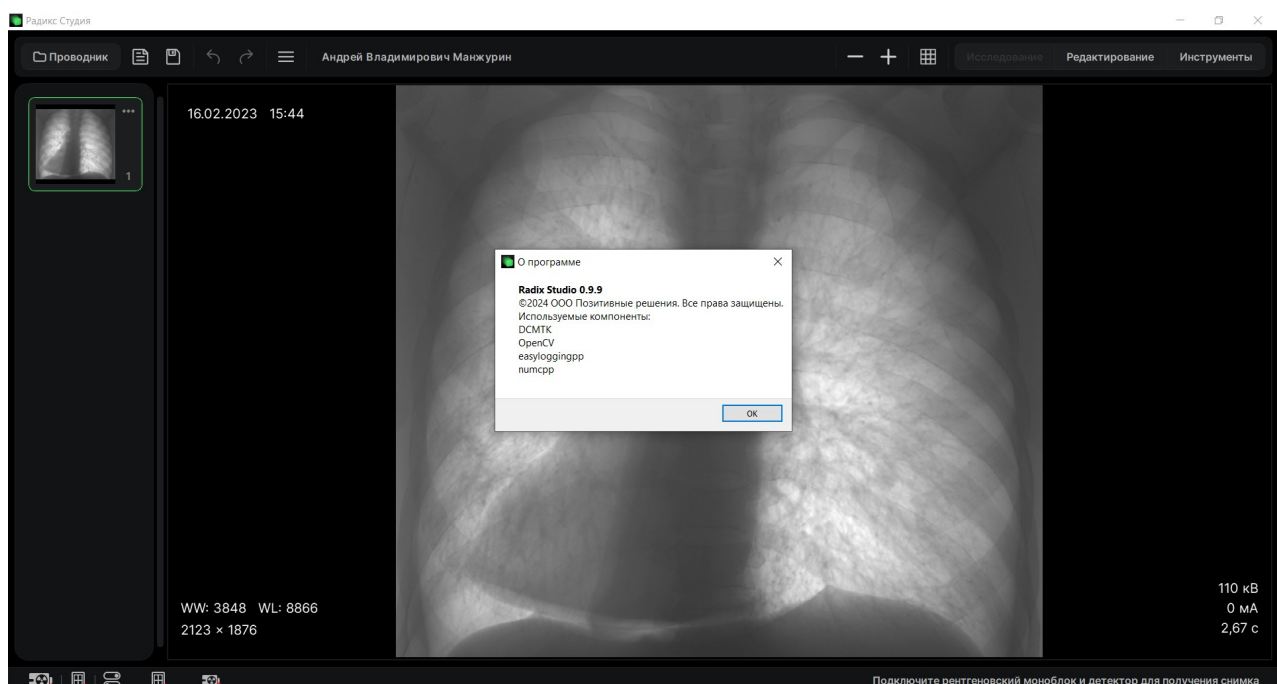


Рисунок В.16 – Диалог «О программе»

					ПГУ1.090501.13.001 ПЗ	Лист
						83
Изм.	Лист	№ докум.	Подпись	Дата		

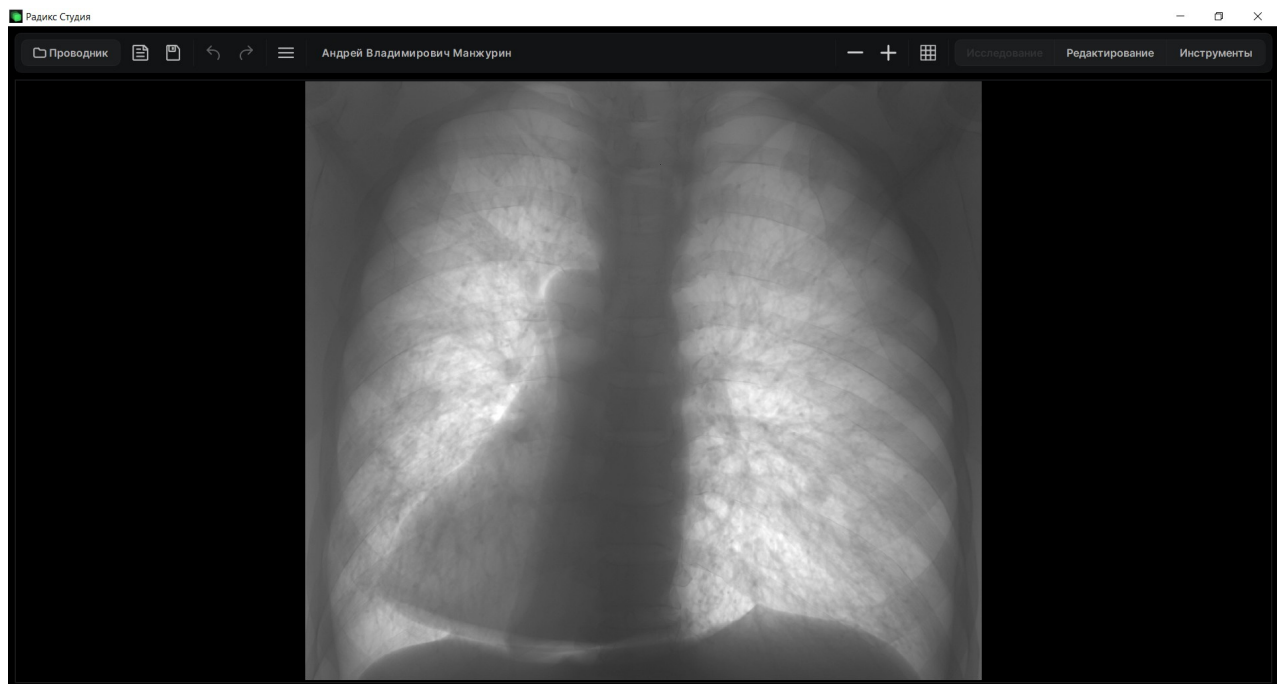


Рисунок В.17 – Меню «Вид»



Рисунок В.18 – Получение снимка

					ПГУ1.090501.13.001 ПЗ	Лист
						84
Изм.	Лист	№ докум.	Подпись	Дата		

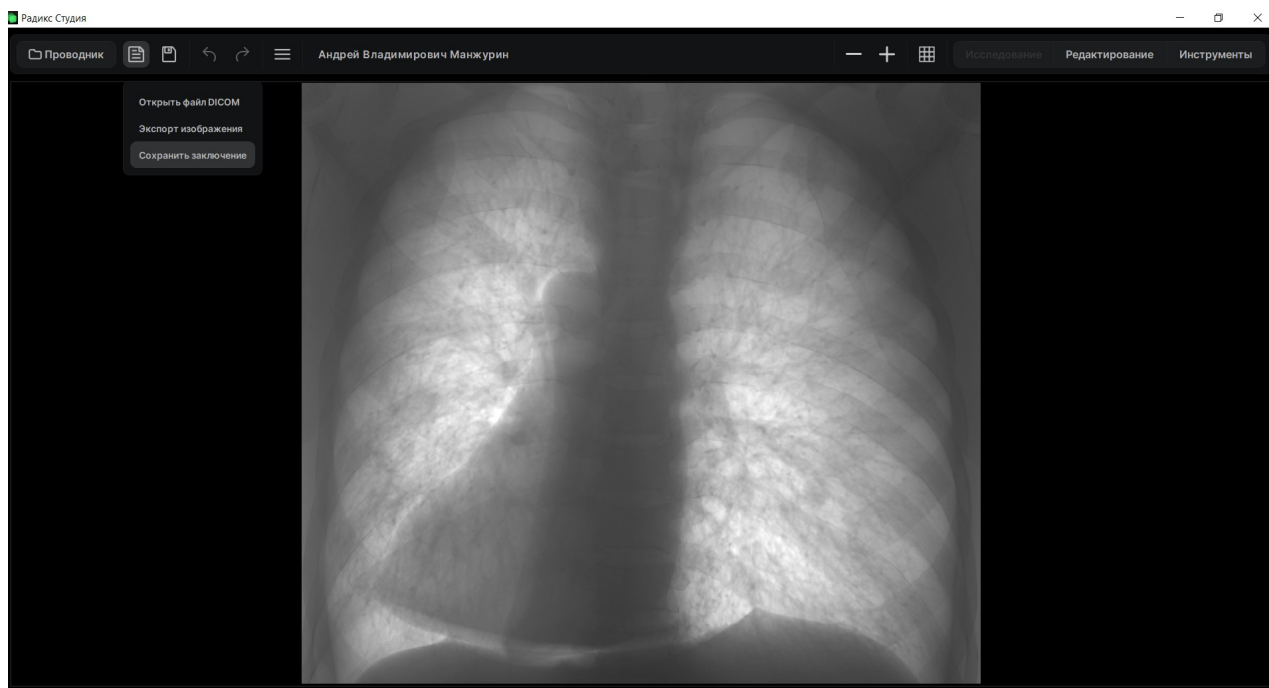


Рисунок В.19 – Сохранение заключения

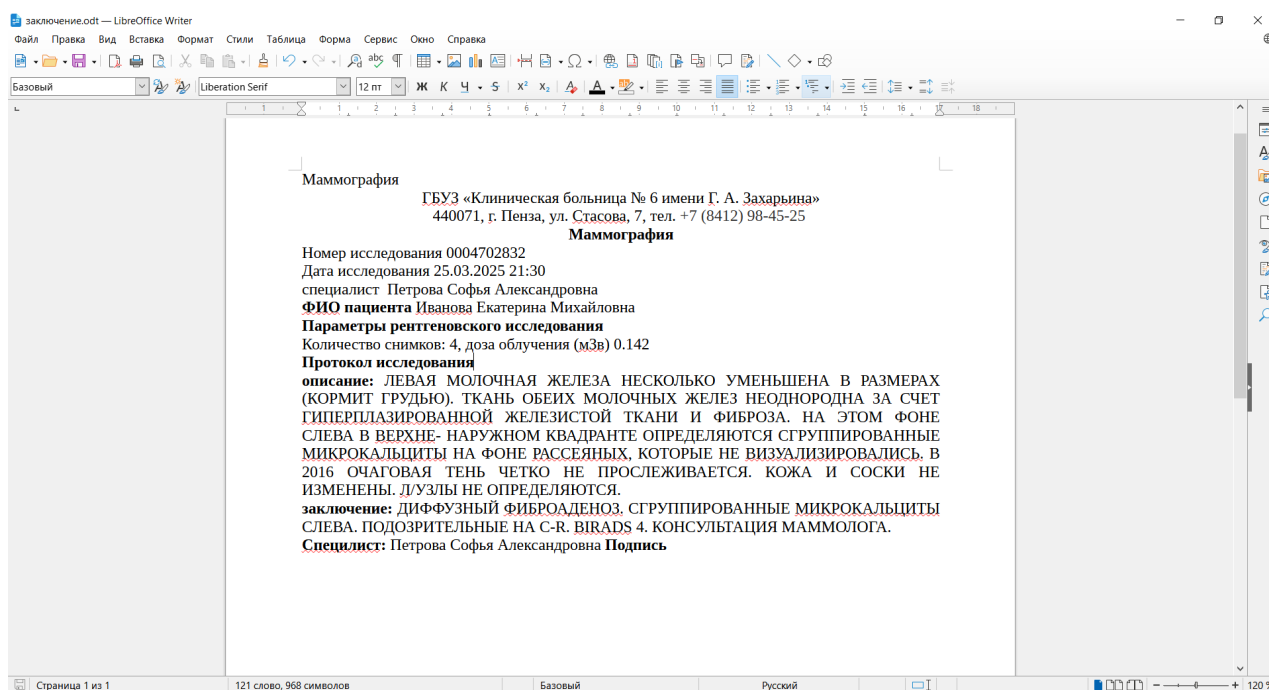


Рисунок В.20 – Врачебное заключение

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		85

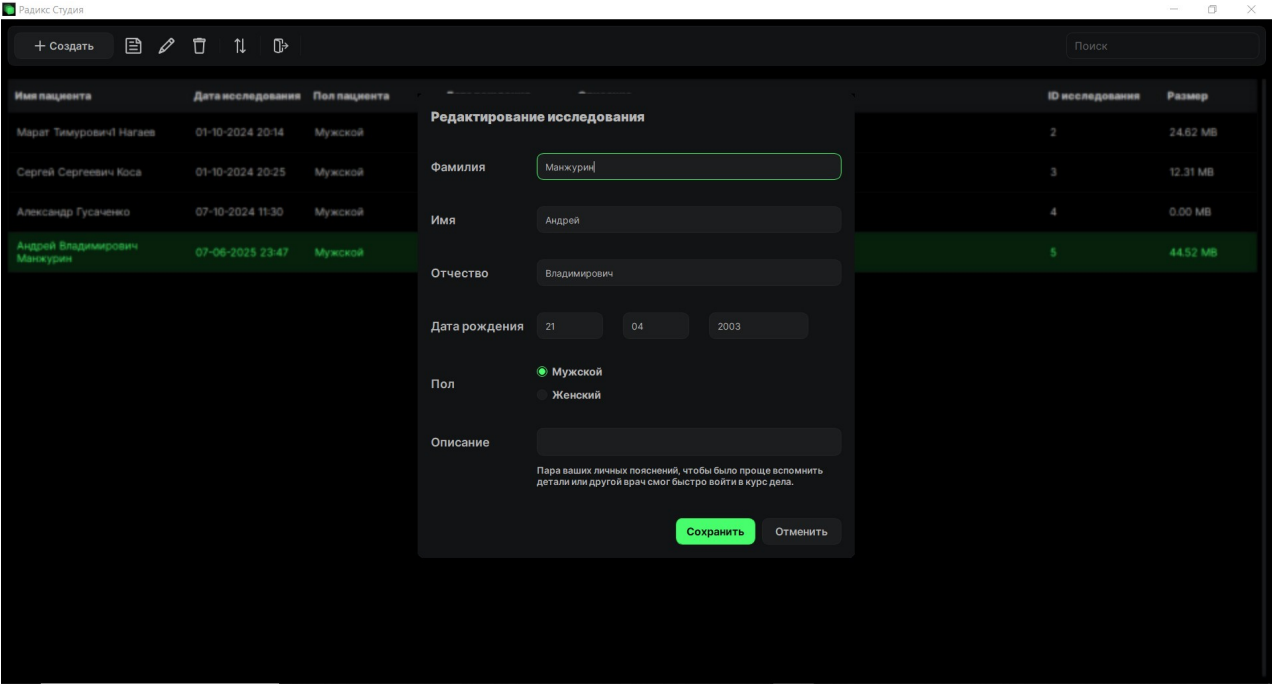


Рисунок В.21 – Редактирование обследования

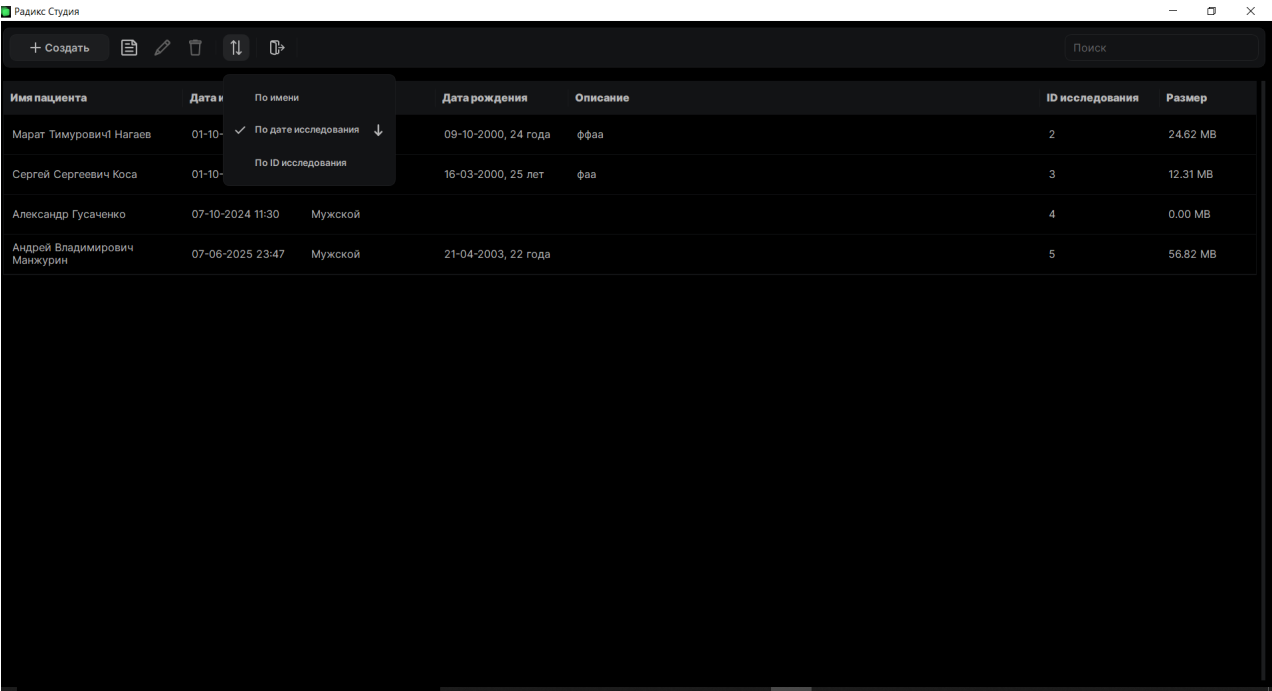


Рисунок В.22 – Изменение типа сортировки

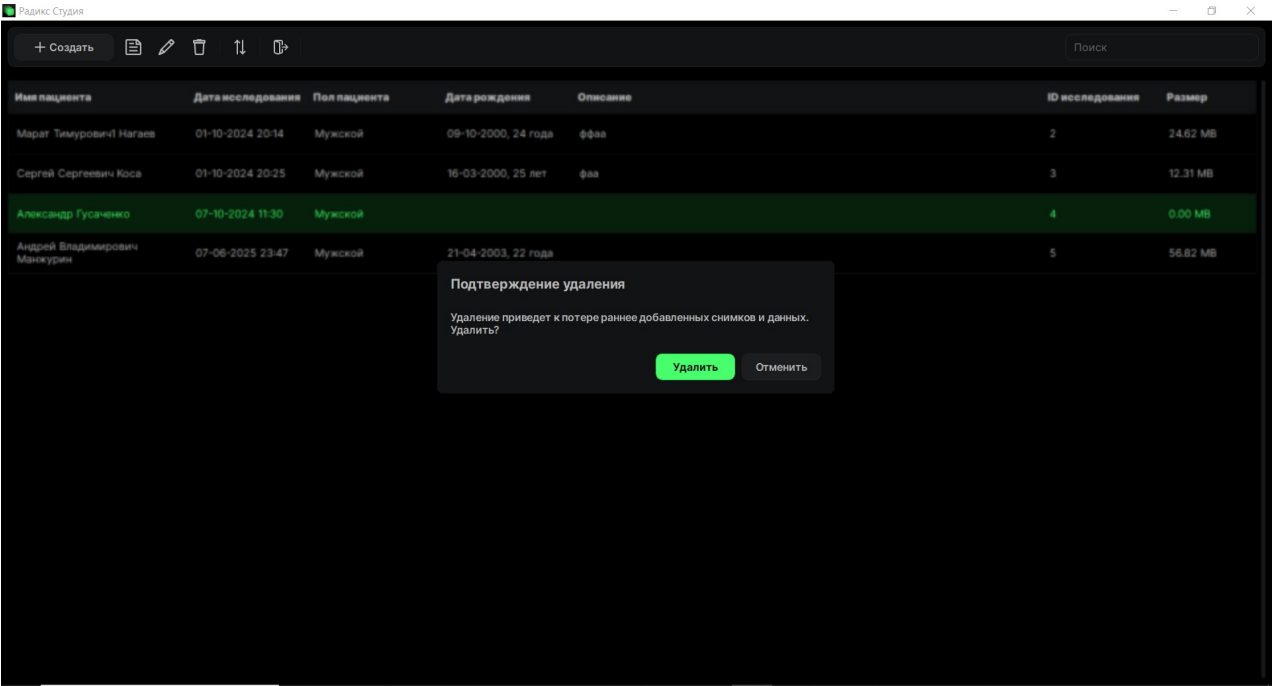


Рисунок В.23 – Удаление обследования

Приложение Г

(обязательное)

Приложение для анализа и обработки рентгеновских снимков

Листинг программы

					ПГУ1.090501.13.001 ПЗ	Лист
						88
Изм.	Лист	№ докум.	Подпись	Дата		

```

#include <iostream>
#include "database_screen.h"
#include "../src/DICOMDatabase.h"

#include <QJsonDocument>
#include <QJsonObject>
#include <QJsonArray>

std::vector<Record> ParseJSON(std::vector<std::string> data){
    std::vector<Record> records;
    for(auto jsonString: data){
        QJsonDocument jsonDoc =
QJsonDocument::fromJson(QByteArray::fromRawData(jsonString.c_str()
, jsonString.size()));

        qDebug() << jsonString.data();
        if (!jsonDoc.isNull()) {
            if (jsonDoc.isObject()) {
                QJsonObject jsonObj = jsonDoc.object();
                Record record;

                // Извлекаем значения из объекта JSON и присваиваем их переменным
                // вашей структуры Record
                record.name = jsonObj["name"].toString();
                record.dateChange = jsonObj["dateChange"].toString();
                record.inspector = jsonObj["inspector"].toString();
                record.object = jsonObj["object"].toString();

                qint64 unixTime = jsonObj["unixTime"].toInt();
                QDateTime dateTime;
                dateTime.setTime_t(unixTime);
                record.dateCreate = dateTime.toString("dd-MM-yyyy");
                record.size = jsonObj["size"].toString();

                records.push_back(record);
            }
        }
    }
}

```

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		89


```

    }
    return records;
}

DatabaseScreen::DatabaseScreen(QWidget *parent) :
    QWidget(parent)
{
    contentLayout = new QVBoxLayout(this);
    contentLayout->setMargin(0);

    // read json-manifests from from file
    std::vector<std::string> data;
    dicom_database.getDatabasesStats(data);

    // confert json-manifests into the structures
    std::vector<Record> records = ParseJson(data);

    RExplorer * explorer = new RExplorer(records, this);
    contentLayout->addWidget(explorer);

    connect(explorer, &RExplorer::clickedReturnToImageViewer,
this, [=]() {
    emit onReturnToImageViewer();
});
}

DatabaseScreen* DatabaseScreen::instance = 0;
DatabaseScreen* DatabaseScreen::getInstance(QWidget *parent)
{
    if (instance == 0){
        instance = new DatabaseScreen(parent);
    }
}

```

					ПГУ1.090501.13.001 ПЗ	Лист
						90
Изм.	Лист	№ докум.	Подпись	Дата		

```

        return instance;
    }

DatabaseScreen::~DatabaseScreen()
{
    //      delete main;
}

#include "qboxlayout.h"
#include <iostream>
#define DICOMDATABASE_H
#include <QDir>
#include <QLabel>
#include <QMessageBox>
#include "examination_screen.h"
#include "../components/Menu.h"
#include "../components/RBottomMenu.h"
#include "../components/RTopMenu.h"
#include "../components/RExaminationMain.h"
#include "../components/RHardware.h"
#include "../components/RExaminationLoader.h"
#include "../ui_examination.h" //TODO: check it
#include "../src/DICOMDatabase.h"
//#include "../components/.h"
#include "components/RExaminationImgsPreview.h"
#include "components/rexplorer.h"

ExaminationScreen::ExaminationScreen(QWidget
*parent):QWidget(parent){
    __rendered__ = false;
    lockRendering = false;
    imagePanel = nullptr;
    hardware = nullptr;
    manipulations = nullptr;
    examinationMain = nullptr;
    topMenu = nullptr;
    manipulations = nullptr;

```

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		91

```

dcms_paths = new QStringList();
jpgs_paths = new QStringList();

contentLayout = new QVBoxLayout(this);
contentLayout->setMargin(0);
//    contentLayout->setAlignment(Qt::AlignTop);
//render();
}

/* Null, because instance will be initialized on demand. */
ExaminationScreen* ExaminationScreen::instance = 0;
ExaminationScreen* ExaminationScreen::getInstance(QWidget *parent)
{
    if (instance == 0){
        instance = new ExaminationScreen(parent);
    }
    return instance;
}

void ExaminationScreen::setLockRendering(bool l){
    lockRendering = l;
    render();
}

void ExaminationScreen::render(){
    if (lockRendering) return;

    if (true){
        //TODO: FIXME: Очень странный баг:
        //Если создать RExaminationMain например, то коннекты
        внутри не будут работать

```

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		92

```

        //TODO: add args to examination main
        //TODO: refactor it!
        QStringList examinations =
DICOMDatabase::getExaminationsNames();
        dicom_database.loadProject(examinations[0]);
        dicom_database.getProjectData(json_data);
        dicom_database.getImages(dcms_paths, jpgs_paths);

        auto rows = DICOMDatabase::getExaminationCount();

        //FIXME:
        //RExaminationMain* rm = new
RExaminationMain(this, dcms_paths, jpgs_paths /*, img_table*/);
        //rm->setGeometry(0,100,580,650); //650, 550 for 4 | 580
for 3
        /*RExaminationImgsPreview* ip = new
RExaminationImgsPreview(this, jpgs_paths);
        ip->setGeometry(0,100,600,200);*/
    }
    //      //TODO: delete if..else and loader
    else {
    //          RExaminationLoader* loader = new
RExaminationLoader(this);
    //          loader->setGeometry(0, 100, 550, 650);
    }

    topMenu = new RTopMenu(this);
    contentLayout->insertWidget(0, topMenu);

    QHBoxLayout * workLayout = new QHBoxLayout;
    contentLayout->insertLayout(1, workLayout);

    bottomMenu = new RBottomMenu(this);
    contentLayout->insertWidget(2, bottomMenu);

```

```

//RHardware Pane

//    QRect pane_size(1300,100,300,690);
//    imagePanel = new RPane(this);
//    imagePanel->setGeometry(pane_size);
//    imagePanel->setMaximumHeight(200);
//    workLayout->addWidget(imagePanel);

//    //p->setGeometry(500,100,800,800);
//    //Radix

    examinationMain = new RExaminationMain(this);
    examinationMain->setObjectName("examinationMain");
    workLayout->addWidget(examinationMain);

////    hardware = new RHardware(this);
/////    hardware->setGeometry(1320,100,400,800); //850
////    manipulations = new RManipulations(this);
/////    manipulations->setGeometry(1320,100,400,800);
////    manipulations->hide();
////    workLayout->addWidget(hardware);

//    /*RExaminationImgsPreview* ip = new
RExaminationImgsPreview(this,
//    RExaminationImgsPreview* im = new
RExaminationImgsPreview(this, jpgs_paths);
//    im->setGeometry(10,100,500,100);*/
//    //im->setGeometry();
//    //Инструменты
}

void ExaminationScreen::keyPressEvent(QKeyEvent *event){
    if (lockRendering) return;
    int key = event->key();

```

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		94

```

//FIXME
if (key == Qt::Key_R){
    this->manipulations->hide();
    this->hardware->show();
}
else if (key == Qt::Key_1){
    img_table->setRowCount(1);
    img_table->setColumnCount(1);
}
else if (key == Qt::Key_2){
    img_table->setRowCount(1);
    img_table->setColumnCount(2);
}
else if (key == Qt::Key_M){
    std::cout<<"Menu"<<std::endl;
}
QWidget::keyPressEvent(event);
}

ExaminationScreen::~ExaminationScreen(){
    if (imagePanel!=nullptr) delete imagePanel;
    //if (instruments!=nullptr) delete instruments;
    if (hardware!=nullptr) delete hardware;
    if (examinationMain!=nullptr) delete examinationMain;
    if (topMenu!=nullptr) delete topMenu;
    if (manipulations!=nullptr) delete manipulations;
    //if (saiObj!=nullptr) delete saiObj;
}

#ifdef EXAMINATIONSCREEN_H
#define EXAMINATIONSCREEN_H

#include <QWidget>
#include <QKeyEvent>
#include <QVBoxLayout>
#include <QHBoxLayout>
#include <nlohmann/json.hpp>

```

```

#include "components/Pane.h"
#include "components/RExaminationTable.h"
#include "../components/RHardware.h"
#include "../components/RManipulations.h"
#include "../components/RBottomMenu.h"
#include "components/rexplorer.h"
//class RHardware;

class RExaminationMain; //forward declaration
class RTopMenu;

class ExaminationScreen : public QWidget
{
    Q_OBJECT

public:
    RExaminationTable* img_table;
    /**
     * Метод доступа к объекту
     * \return Статический объект класса ExaminationScreen
     */
    static ExaminationScreen* getInstance(QWidget *parent =
nullptr);
    void render();
    void setLockRendering(bool l);
    nlohmann::json json_data;
    QStringList* dcms_paths;
    QStringList* jpgs_paths;
    ExaminationScreen(QWidget *parent = nullptr);
    ~ExaminationScreen();
    static ExaminationScreen* instance; ///< статический объект
    класса
    //private before
    bool __rendered__;
    bool lockRendering;
    RPane* imagePanel;

```

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		96

```

//RInstruments* instruments;
RHardware* hardware;
RManipulations* manipulations;
RExaminationMain* examinationMain;
RTopMenu* topMenu;
RBottomMenu * bottomMenu;

private:
    QVBoxLayout * contentLayout;
    QHBoxLayout * workLayout;
    std::vector<Record> records;

protected:
    void keyPressEvent(QKeyEvent *event) override;
};

#endif // EXAMINATIONSCREEN_H

#pragma once

#include <QWidget>
#include "../components/Pane.h"
#include "../components/RDatabaseTopMenu.h"
#include "../components/RDatabaseMain.h"
#include "../components/RDatabaseTable.h"
#include "../components/RClickableLabel.h"
#include "../components/RDatabaseTable.h"
#include "../components/REdit.h"
#include "../components/rexplorer.h"
#include <vector>

class DatabaseScreen : public QWidget
{
    Q_OBJECT

public:
    /**
     * Метод доступа к объекту
     * \return Статический объект класса RadixController

```

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		97


```

*/
static DatabaseScreen* getInstance(QWidget *parent = nullptr);
static DatabaseScreen* instance; ///< статический объект
класса
~DatabaseScreen();

RDatabaseTable* databaseTable;
RDatabaseMain* main;
RDatabaseTopMenu* tm;
RClickableLabel* change_examination;
QLabel* total_examinations;
QLabel* total_number;
QLabel* total_storage;
QLabel* storage_number;
RExplorer * explorer;

signals:
    void onReturnToImageViewer();

private:
    DatabaseScreen(QWidget *parent = nullptr);

    QVBoxLayout * contentLayout;

    std::vector<Record> records;
};
#include <QDir>
#include <QString>
#include <QStandardPaths>
#include "DICOMDatabase.h"
#include "dcmtk/ofstd/offile.h"
#include "thirdparty/easylogging++.h"
#include <fstream>

QString _getBasePath(){

```

```

        QString path =
QStandardPaths::writableLocation(QStandardPaths::DocumentsLocation
);

        //Для юниттестов не трогаем основную папку
#ifdef _UNITTESTS
            path = "C:/Radix1";
            return path;
#endif
        //FIXME:
#ifdef _WIN32
            path = "C:";
#endif
        return path+'/'+"Radix";
    }

DICOMDatabase::DICOMDatabase() {
    _setupPaths("Test");
    /*if (_checkProject("Test")){
        std::cout<<"Error while loading project"<<std::endl;
    }*/
}

DICOMDatabase::DICOMDatabase(QString name) {
    _setupPaths(name);
    /*if (_checkProject(name)){
        std::cout<<"Error while loading project"<<std::endl;
    }*/
}

void DICOMDatabase::addDCM(QString file) {
    //Add image to database
    //1. Copy DCM
    QFile::copy(file,dcm_path);
    //2. Create converted image

}

```

```

/*void DICOMDatabase::deleteDCM(QString file){
    QString path = appendPath(dcm_path,file);
    LOG(INFO) << "deleting dcm..." << path.toStdString();
    QFile::remove(path);
}*/

void DICOMDatabase::deleteDCM(QString file){
    LOG(INFO) << "deleting dcm..." << file.toStdString();
    QFile::remove(file);
}

void DICOMDatabase::addPixelData(QString pixelDataFile, QString
dcmFile, int width, int height){
    LOG(INFO) << "addPixelData ";
    LOG(INFO) << pixelDataFile.toStdString().c_str();
    LOG(INFO) << dcmFile.toStdString().c_str();
    Image2Dcm i2d;
    OFCondition cond;
    E_TransferSyntax writeXfer = EXS_LittleEndianImplicit;
    DcmDataset* resultDset;
    DcmFileFormat fileformat;
    //TODO: replace it with manual tag inserts
    //mars1417v.dcm
    OFCondition status = fileformat.loadFile("C:\\english.dcm");
    if (!status.good()){
        LOG(INFO)<<"Problem with C:\\english.dcm";
        LOG(INFO)<< status.text();
        return;
    }
    resultDset = fileformat.getDataset();
    //resultDset->putAndInsertString(DCM_PatientName, "Brandon
Lara");
    resultDset->putAndInsertUint16(DCM_Rows,height);
    resultDset->putAndInsertUint16(DCM_Columns,width);
    size_t size = height*width;//???

```

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		100

```

        uint16_t* data = new uint16_t[size];
        LOG(INFO) << pixelDataFile.toStdString().c_str();
        FILE* pixelStream =
fopen(pixelDataFile.toStdString().c_str(),"rb");
        if (!pixelStream){
            LOG(INFO)<<"pixelStream error";
            return;
        }
        int c = fread(data,sizeof(uint16_t),size,pixelStream);
        LOG(INFO) << "Read " << c << "bytes";
        LOG(INFO) << "Length is" << size << "bytes";
        fclose(pixelStream);
        resultDset->putAndInsertUint16Array(DCM_PixelData, data,
size);
        DcmFileFormat dcmff(resultDset);
        LOG(INFO)<<"Saving...";
        cond = dcmff.saveFile(dcmFile.toStdString().c_str(),
writeXfer/*, EET_ExplicitLength*/);
        if(cond.bad()) LOG(INFO) << "Fatal error while saving";
        else LOG(INFO)<<"Zaebis!";
    }

void DICOMDatabase::mergeExaminations(QString dir){
    QString dcmPath = appendPath(project_path,"DCM");
    QDir dcm_dir(dcmPath);
    QStringList pattern = {"*.dcm"};
    QStringList dcms = dcm_dir.entryList(pattern);
    for (QString& file:dcms){
        QFile::copy(file, dcm_path);
    }
}

void DICOMDatabase::addFiles(QStringList files){
    for (QString& file:files){
        QFile::copy(file, dcm_path);
    }
}

```

```

}

//5 -> C:\\Hello\\DCMS\\5.dcm
QString DICOMDatabase::getDcmFilenameByNumber(int number) {
    return dcm_path+"/"+QString::number(number)+".dcm";
}

//5 -> C:\\Hello\\DCMS\\.converted\\5.jpg
QString DICOMDatabase::getJpgFilenameByNumber(int number) {
    return dcm_path+"/"+".converted"+"/"+QString::number(number)
+".jpg";
}

bool DICOMDatabase::getImages(QStringList* dcms, QStringList*
jpgs){
    QStringList pattern = {"*.dcm"};
    QDir dcm_dir(dcm_path);
    if (!dcm_dir.exists()){
        LOG(ERROR)<<"ERROR: Coudn't find DCM folder (located at
"<<dcm_path.toStdString()<<")";
        return false;
    }
    (*dcms) = dcm_dir.entryList(pattern);
    *jpgs = *dcms; //same filenames, but another extension
(instead of 3.dcm we have 3.jpg)
    if (dcms->size()==0){
        LOG(WARNING)<<"WARN: the project has zero size";
        LOG(INFO)<<"dcm_path is
"<<dcm_path.toStdString()<<std::endl;
        jpgs = dcms; //empty StringList too
        return true;
    }
    QString converted_path = appendPath(dcm_path, ".converted");
    QDir converted_dir(converted_path);
    if (!converted_dir.exists()){
        LOG(INFO)<<"creating dir .converted";

```

					ПГУ1.090501.13.001 ПЗ	Лист
						102
Изм.	Лист	№ докум.	Подпись	Дата		

```

        dcm_dir.mkpath(converted_path);
    }
    //We should have full path, not just 3.jpg or 3.dcm
    for (int i=0;i<dcms->size();i++){
        QString elem = (*dcms)[i];
        (*dcms)[i] = dcm_path+'/'+elem;
        (*jpgs)[i] = converted_path+'/'+elem.replace("dcm","jpg");
    }
    for (int i=0;i<dcms->size();i++) {
        std::string fname = (*dcms)[i].toUtf8().data();
        //          bool result = dcm2jpg((*dcms)[i].toStdString(),
        (*jpgs)[i].toStdString());
        LOG(INFO)<<"Calling dcm2jpg";
        bool result = dcm2jpg((*dcms)[i].toUtf8().data(), (*jpgs)
[i].toUtf8().data());
        if (!result){
            LOG(ERROR)<<"ERROR: dcm2jpg returned false";
            return false;
        }
    }
    assert(jpgs->size()==dcms->size());
    return true;
}

void DICOMDatabase::_setupPaths(QString name){
    project_name = name;
    base_path = _getBasePath();
    project_path = appendPath(base_path,project_name);
    dcm_path = appendPath(project_path,"DCM");
    //We should check project
}

bool DICOMDatabase::loadProject(QString name){
    _setupPaths(name);
    QString manifest_path = appendPath(base_path,project_name)
+'/'+"meta.json";

```

					ПГУ1.090501.13.001 ПЗ	Лист
						103
Изм.	Лист	№ докум.	Подпись	Дата		

```

//      std::ifstream json_file(manifest_path.toStdString());
std::ifstream json_file(manifest_path.toUtf8().data());
//json data;
try{
    m_data = nlohmann::json::parse(json_file);
    std::string name = m_data["name"];
    std::cout<<"name: "<<name<<std::endl;
    std::string date = m_data["date"];
    std::cout<<"date: "<<date<<std::endl;
    //Test is a project name, DCM is a inner folder with dicom
batches
    QDir dir(dcm_path);
    if (!dir.exists()){
        //throw;
        std::cout<<"Project dir was not found"<<std::endl;
    }

}
//handles JSON error
catch(...){
    std::cout<<"Error with file"<<std::endl;
    return false;
}
//Now we can assume project structure is ok
return true;
}

void DICOMDatabase::getProjectData(nlohmann::json& data){
    data = m_data;
}

//static functions

/*void DICOMDatabase::getDCMfromPixelData(QString filepath, int
width = 2304, int height = 2800){
    LOG(INFO)<<"getDCMfromPixelData";

```

					ПГУ1.090501.13.001 ПЗ	Лист
						104
Изм.	Лист	№ докум.	Подпись	Дата		

```

Image2Dcm i2d;
OFCondition cond;
E_TransferSyntax writeXfer = EXS_LittleEndianImplicit;
DcmDataset* resultDset;
DcmFileFormat fileformat;
//TODO: replace it with manual tag inserts
//mars1417v.dcm
OFCondition status = fileformat.loadFile("C:\\english.dcm");
if (!status.good()){
    LOG(INFO)<<"Problem with C:\\english.dcm";
    LOG(INFO)<< status.text();
    return;
}
resultDset = fileformat.getDataset();
//resultDset->putAndInsertString(DCM_PatientName, "Brandon
Lara");
resultDset->putAndInsertUint16(DCM_Rows,height);
resultDset->putAndInsertUint16(DCM_Columns,width);
size_t size = height*width;//???
uint16_t* data = new uint16_t[size];
LOG(INFO) << filepath.toStdString().c_str();
FILE* pixelStream =
fopen(filepath.toStdString().c_str(),"rb");
if (!pixelStream){
    LOG(INFO)<<"pixelStream error";
    return;
}
int c = fread(data,sizeof(uint16_t),size,pixelStream);
LOG(INFO) << "Read " << c << "bytes";
LOG(INFO) << "Length is" << size << "bytes";
fclose(pixelStream);
resultDset->putAndInsertUint16Array(DCM_PixelData, data,
size);
DcmFileFormat dcmff(resultDset);
LOG(INFO)<<"Saving...";
cond = dcmff.saveFile("D:\\fromPanel2.dcm", writeXfer//,

```

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		105


```

EET_ExplicitLength//);
    if(cond.bad()) LOG(INFO) << "Fatal error while saving";
    else LOG(INFO)<<"Zaebis!";
    //New code:
    //DicomImage lNewImg(resultDset, resultDset-
>getOriginalXfer());
    //lNewImg.setMinMaxWindow(0); //NOTE: Enable minmax
    //lNewImg.writeJPG("D:\\frompanel2.jpg");
}*/

//static functions

bool DICOMDatabase::createProject(nlohmann::json json_data){
    /*
        1. mkdir inside of base_path
        2. mkdir DCM
        3. touch meta.json
        4. fill meta.json
    */
    std::string name = json_data["name"];
    QString base_path = _getBasePath();
    QString project_path =
appendPath(base_path,QString::fromStdString(name));
    QString dcm_path = appendPath(project_path,"DCM");
    QDir dir(project_path);
    if (dir.exists()){
        std::cout<<"This project folder is already
exists"<<std::endl;
        return false;
    }
    dir.mkpath(project_path);
    dir.mkpath(dcm_path);
    //QFile file(project_path + '/' + "meta.json");
    //file.open(QIODevice::WriteOnly);
    QString metaPath = project_path + '/' + "meta.json";
    std::ofstream file(metaPath.toStdString());

```

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		106

```

        file << std::setw(4) << json_data << std::endl;
        file.close();
        return true;
    }

bool DICOMDatabase::createProject(QJsonObject json_data){
    /*
        1. mkdir inside of base_path
        2. mkdir DCM
        3. touch meta.json
        4. fill meta.json
    */
    QString name = json_data["name"].toString();
    QString base_path = _getBasePath();
    QString project_path = appendPath(base_path, name);
    QString dcm_path = appendPath(project_path, "DCM");
    QDir dir(project_path);
    if (dir.exists()){
        LOG(WARNING)<<"This project folder is already exists";
        LOG(INFO)<<"Examination name: "<<name.toStdString();
        return false;
    }
    dir.mkpath(project_path);
    dir.mkpath(dcm_path);
    QString metaPath = project_path + '/' + "meta.json";
    QFile file(metaPath);
    file.open(QIODevice::WriteOnly);
    //std::ofstream file(metaPath.toStdString());
    QJsonDocument doc(json_data);
    file.write(doc.toJson(QJsonDocument::Compact));
    file.close();
    return true;
}

//TODO: test it!

bool DICOMDatabase::editProject(QString name, nlohmann::json

```

					ПГУ1.090501.13.001 ПЗ	Лист
						107
Изм.	Лист	№ докум.	Подпись	Дата		

```

json_data){
    QString base_path = _getBasePath();
    QString project_path = appendPath(base_path,name);
    QString metaPath = project_path + '/' + "meta.json";
    std::ofstream file(metaPath.toStdString());
    if(file.is_open()){
        LOG(INFO)<<"Normal file opening";
    } else {
        LOG(INFO)<<"Error while file opening";
        return false;
    }
    file << std::setw(4) << json_data << std::endl;
    file.close();
    return true;
}

bool DICOMDatabase::editProject(QString oldName, QJsonObject
json_data){
    QString base_path = _getBasePath();
    QString newName = json_data["name"].toString();
    QString project_path = appendPath(base_path,newName);
    if (newName!=oldName){
        //Смена названия обследования
        QString oldPath = appendPath(base_path,oldName);
        QFile::rename(oldPath, project_path);
    }
    QString metaPath = project_path + '/' + "meta.json";
    QFile file(metaPath);
    file.open(QIODevice::WriteOnly);
    if(!file.isOpen()){
        LOG(WARNING)<<"editProject failed, project_path is:
"<<project_path.toStdString();
        return false;
    }
    QJsonDocument doc(json_data);
    file.write(doc.toJson(QJsonDocument::Compact));
}

```

```

        file.close();
        return true;
    }

QStringList DICOMDatabase::getExaminationsNames() {
    QDir folder(_getBasePath());
    QStringList c = folder.entryList(QDir::Dirs |
QDir::NoDotAndDotDot);
    return c;
}

int DICOMDatabase::getExaminationCount() {
    return getExaminationsNames().count();
}

qint64 dirSize(QString dirPath) {
    qint64 size = 0;
    QDir dir(dirPath);
    //calculate total size of current directories' files
    QDir::Filters fileFilters = QDir::Files|QDir::System|
QDir::Hidden;
    for(QString& filePath : dir.entryList(fileFilters)) {
        QFile fi(dir, filePath);
        size+= fi.size();
    }
    //add size of child directories recursively
    QDir::Filters dirHardware = QDir::Dirs|QDir::NoDotAndDotDot|
QDir::System|QDir::Hidden;
    for(QString& childDirPath : dir.entryList(dirHardware))
        size+= dirSize(dirPath + '/' + childDirPath);
    return size;
}

QString DICOMDatabase::getDatabasesSize() {
    qint64 size = dirSize(_getBasePath());
    QStringList units = {"Б", "КБ", "МБ", "ГБ"};

```

					ПГУ1.090501.13.001 ПЗ	Лист
						109
Изм.	Лист	№ докум.	Подпись	Дата		

```

    int i;
    double outputSize = size;
    for(i=0; i<units.size()-1; i++) {
        if(outputSize<1024) break;
        outputSize/=1024;
    }
    return QString("%0 %1").arg(outputSize, 0, 'f',
2).arg(units[i]);
}

//TODO: test it
void DICOMDatabase::getDatabaseManifest(QString name, QJsonObject&
jsonData){
    QDir folder(_getBasePath()+ '/' +name);
    QString manifestPath = folder.path()+ '/' +"meta.json";
    QFile manifestFile(manifestPath);
    manifestFile.open(QIODevice::ReadOnly | QIODevice::Text);
    QByteArray raw = manifestFile.readAll();
    manifestFile.close();
    LOG(INFO)<<manifestPath.toStdString();
    jsonData=QJsonDocument::fromJson(raw).object();
}

/*nlohmann::json*/ void
DICOMDatabase::getDatabasesStats(std::vector<std::string>& data){
    QStringList names = getExaminationsNames();
    data.reserve(names.count());
    for (QString& name:names){
        /*std::string json_text = R"({
            "name":"Radix лучший!",
            "date":"22.03.2023",
            "year":2022,
            "owner":"Marat Nagayev",
            "description":"Test desc",
            "status":"Назначен",
            "date":"13.10.2022",

```

					ПГУ1.090501.13.001 ПЗ	Лист
						110
Изм.	Лист	№ докум.	Подпись	Дата		

```

        "time":"9:59",
        "inspector":317
    });*/
    QDir folder(_getBasePath()+ '/' +name);
    QFile manifestFile(folder.path()+ '/' +"meta.json");
    manifestFile.open(QIODevice::ReadOnly | QIODevice::Text);
    //          std::string raw =
manifestFile.readAll().toStdString();
    QString manifestData = manifestFile.readAll();
    std::string raw(manifestData.toUtf8().data());
    data.push_back(raw);
    manifestFile.close();

    }
}

bool DICOMDatabase::deleteExamination(QString examinationName) {
    LOG(INFO)<<"called deleteExaminaion with
"<<examinationName.toStdString();
    QString basePath = _getBasePath();
    QString projectPath = appendPath(basePath,examinationName);
    std::cout<<projectPath.toStdString()<<std::endl;
    QDir dir(projectPath);
    return dir.removeRecursively();
}

#include <iostream> //std::cout
#include <fstream>
#include "ConfigManager.h"

ConfigManager::ConfigManager() {
    //Checks if file doesn't exist create it and write
defaultContent
    std::ifstream f("/home/marat/Documents/Radix/config.json");
    bool fileExists = f.good();
    if (!fileExists){
        std::cout << "creating radix confing..." << std::endl;

```

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		111

```

        jsonContent["port"]="dynamic";
        jsonContent["koef"]=0.0;
        jsonContent["modes"]=nlohmann::json::array();
        //TODO: write it to file
    }
    else {
        jsonContent = nlohmann::json::parse(f);
        return;
    }
    //TODO: write content to file
}

nlohmann::json& ConfigManager::getModes(){
    return jsonContent["modes"];
}

std::string ConfigManager::getPort(){
    return jsonContent["port"];
}

float ConfigManager::getKoef(){
    return jsonContent["koef"];
}

#pragma once

#include <nlohmann/json.hpp>

#define defaultPort "dynamic"
#define defaultKoef 0.0

class ConfigManager{
public:
    ConfigManager();
    nlohmann::json& getModes();
    std::string getPort();
    float getKoef();

```

					ПГУ1.090501.13.001 ПЗ	Лист
						112
Изм.	Лист	№ докум.	Подпись	Дата		

```

private:
    nlohmann::json jsonContent;
};

#ifdef CONFIGMANAGER_H
    ConfigManager configManager;
#else
    extern ConfigManager configManager;
#endif

#include "ImgLibrary.h"
#include "thirdparty/easylogging++.h"
#include "src/utils.h"
#include <QMessageBox>
#include <string>

ImgLibrary::ImgLibrary() {
    vector<operation_s> ops = {};
    history.pointer = 0;
    history.heavyOps = 0;
    history.ops = ops;
    img = nullptr;
    path = "";
    //Magick::Image img(path);
}

ImgLibrary::ImgLibrary(string _path) {
    vector<operation_s> ops = {};
    history.pointer = 0;
    history.ops = ops;
    path = _path;
    img=new Magick::Image(_path);
    img->magick("jpg");
}

ImgLibrary::~ImgLibrary() {
    if (img!=nullptr){

```

					ПГУ1.090501.13.001 ПЗ	Лист
						113
Изм.	Лист	№ докум.	Подпись	Дата		


```

        delete img;
    }
}

string ImgLibrary::getPath(){
    return this->path;
}

bool isHeavy(int opcode){
    return (opcode>IMG_INVERT);
}

void ImgLibrary::addOperation(operation_s operation){
    if (this->historyEnabled){
        //TODO: test it
        if (history.heavyOps==HEAVYLIMIT){
            //Превысили лимит на тяжелые операции
            QMessageBox::warning(nullptr, "Лимит", "Превышен лимит
на тяжелые операции");
            LOG(INFO)<<"Heavy limit exceed";
            this->history.ops.pop_back();
            this->history.pointer--;
        }
        if (isHeavy(operation.code)){
            LOG(INFO)<<"inc heavy";
            history.heavyOps++;
        }
        //Случай если мы в середине истории
        if (history.pointer==history.ops.size()){
            LOG(INFO)<<"append to history";
            this->history.ops.push_back(operation);
        } else {
            LOG(INFO)<<"middle of the history";
            //Забываем альтернативную ветку истории
            if (history.pointer<history.ops.size()){
                LOG(INFO)<<"forget";
            }
        }
    }
}

```

```

        history.ops.resize(history.pointer+1);
    }
    this->history.ops[this->history.pointer]=operation;
}
this->history.pointer++;
}
}

//TODO: test it
void ImgLibrary::doOperation(operation_s op, bool mode){
    int opcode = op.code;
    //Нам нужно отключить добавление в историю операций вроде
    rotate
    //Чтобы мы просто повернули картинку, без добавления в историю
    this->historyEnabled = false;
    switch(opcode){
        case IMG_ROTATE:{
            int arg = atoi(op.args.c_str());
            if (mode==UNDOMODE){
                //if (arg<0) arg = 360-(-arg); //-90 => 270
                //this->rotate(360-arg);
                this->rotate(-arg);
            } else {
                this->rotate(arg);
            }
        }
        break;
        case IMG_VERTICAL:
            this->vertical();
            break;
        case IMG_HORIZONTAL:
            this->horizontal();
            break;
        case IMG_INVERT:
            this->invertColors();
            break;
    }
}

```

```

        default:
            LOG(WARNING)<<"Unknown opcode for cancelOperation:
" << opcode;
        }
        this->historyEnabled = true;
    }

void ImgLibrary::undo() {
    LOG(INFO)<<"undo";
    int count = this->history.ops.size();
    if (count==0 || history.pointer==0) {
        //Отменять нечего
        LOG(INFO)<<"count: " << count <<"pointer: " << history.pointer;
        QMessageBox::warning(nullptr, "UNDO", "Отменять нечего");
        return;
    }
    //NOTE: получаем текущий элемент

    operation_s operation = this->history.ops[this-
>history.pointer-1];
    if (!isHeavy(operation.code)) {
        LOG(INFO)<<"Canceled operation#" << this->history.pointer-1;
        LOG(INFO)<<"size: " << this->history.ops.size();
        doOperation(operation, UNDOMODE);
    } else {
        //Get pixel data
        Magick::Blob blob = operation.blob;
        Magick::Image* img1 = new Magick::Image(blob);
        img = std::move(img1);
        img->write(this->path);
    }
    this->history.pointer--;
}

void ImgLibrary::redo() {
    int count = this->history.ops.size();

```

					ПГУ1.090501.13.001 ПЗ	Лист
						116
Изм.	Лист	№ докум.	Подпись	Дата		

```

LOG(INFO)<<"redo: "<<count<<" "<<history.pointer;
if (count==0 || history.pointer==count){
    //Повторять нечего
    QMessageBox::warning(nullptr,"REDO","Повторять нечего");
    return;
}
//NOTE: получаем последний элемент не извлекая его
operation_s operation = this->history.ops[this-
>history.pointer];
if (!isHeavy(operation.code)){
    doOperation(operation,REDOMODE);
} else {
    Magick::Blob blob = operation.blob;
    Magick::Image* img1 = new Magick::Image(blob);
    img = std::move(img1);
    img->write(this->path);
}
this->history.pointer++;
}

void ImgLibrary::undoAll(){
    LOG(INFO)<<"undoAll";
    string dicomPath = getDcmFromImgPath(this->path);
    dcm2jpg(dicomPath,this->path); // "C://Radix//1.png"
}

void ImgLibrary::save(){
    //Сохраняет jpg как dcm
    std::string dicomPath = getDcmFromImgPath(path);
    LOG(INFO)<<"Saving (path: "<<path<<")";
    int ret = img2dcm(path,dicomPath);
    if (!ret){
        LOG(WARNING)<<"img2dcm doesn't work";
    }
    //Обнуляем историю
    vector<operation_s> ops = {};

```

					ПГУ1.090501.13.001 ПЗ	Лист
						117
Изм.	Лист	№ докум.	Подпись	Дата		

```

        history.pointer = 0;
        history.heavyOps = 0;
        history.ops = ops;
    }

bool ImgLibrary::hasUnsavedChanges() {
    return history.ops.size() != 0;
}

void ImgLibrary::rotate(int degress) {
    img->rotate(degress);
    string args = std::to_string(degress);
    operation_s operation = {};
    operation.code=IMG_ROTATE;
    operation.args=args;
    this->addOperation(operation);
    img->write(this->path);
}

void ImgLibrary::vertical() {
    img->flip();
    operation_s operation = {};
    operation.code=IMG_VERTICAL;
    operation.args="";
    this->addOperation(operation);
    img->write(this->path);
}

void ImgLibrary::horizontal() {
    img->flop();
    operation_s operation = {};
    operation.code=IMG_HORIZONTAL;
    operation.args="";
    this->addOperation(operation);
    img->write(this->path);
}

```

```

void ImgLibrary::invertColors(){
    img->negate();
    operation_s operation = {};
    operation.code=IMG_INVERT;
    operation.args="";
    this->addOperation(operation);
    img->write(this->path);
}

```

//от -100 до 100 => [0,200]

```

void ImgLibrary::setBrightness(int value){
    operation_s operation = {};
    operation.code=IMG_BRIGHTNESS;
    operation.args="";
    img->write(&operation.blob);
    this->addOperation(operation);
    img->modulate(value+100,100,100);
    img->write(this->path);
}

```

//уменьшаем или увеличиваем

```

void ImgLibrary::setContrast(bool value){
    operation_s operation = {};
    operation.code=IMG_CONTRAST;
    operation.args="";
    img->write(&operation.blob);
    this->addOperation(operation);
    img->contrast(value);
    img->write(this->path);
}

```

//[0,10] В будущем возможно [0,50] если скомпилировать с
USE_OPENMP

					ПГУ1.090501.13.001 ПЗ	Лист
						119
Изм.	Лист	№ докум.	Подпись	Дата		

```

void ImgLibrary::reduceNoise(int value){
    operation_s operation = {};
    operation.code=IMG_NOISE;
    operation.args="";
    img->write(&operation.blob);
    this->addOperation(operation);
    img->reduceNoise(value);
    img->write(this->path);
}

```

```

void ImgLibrary::setSharpen(int value){
    operation_s operation = {};
    operation.code=IMG_SHARPEN;
    operation.args="";
    img->write(&operation.blob);
    this->addOperation(operation);
    img->sharpen(0.0, value);
    img->write(this->path);
}

```

```

ImgToolsManager::ImgToolsManager(){
    hashTable = new ImgToolsManagerContainer;
}

```

```

ImgToolsManager::~~ImgToolsManager(){
    delete hashTable;
}

```

```

ImgToolsManager* ImgToolsManager::instance = nullptr;

```

```

ImgToolsManager* ImgToolsManager::getInstance()
{
    if (instance == nullptr){
        instance = new ImgToolsManager();
    }
}

```

					ПГУ1.090501.13.001 ПЗ	Лист
						120
Изм.	Лист	№ докум.	Подпись	Дата		

```

    }
    return instance;
}

ImgLibrary* ImgToolsManager::get(QString path){
    //return hashTable[path];
    return hashTable->value(path);
}

void ImgToolsManager::set(QString path, ImgLibrary* il){
    hashTable->insert(path,il);
}

void ImgToolsManager::saveAll(){
    for(auto val: hashTable->keys()){
        auto tool = hashTable->value(val);
        tool->save();
    }
}

bool ImgToolsManager::hasUnsavedChanges(){
    for(auto val: hashTable->keys()){
        auto tool = hashTable->value(val);
        if (tool->hasUnsavedChanges()) return true;
    }
    return false;
}

#ifdef ImgLibrary_H
#define ImgLibrary_H

#include <Magick++.h>
#include <QString>
#include <QMap>

using std::string;
using std::vector;

```

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		121


```

const int HEAVYLIMIT = 2;

const bool UNDOMODE = true;
const bool REDOMODE = false;

//Canceled operations
const int IMG_ROTATE = 1;
const int IMG_VERTICAL = 2;
const int IMG_HORIZONTAL = 3;
const int IMG_INVERT = 4;
//Non canceled ops
const int IMG_BRIGHTNESS = 5;
const int IMG_CONTRAST = 6;
const int IMG_NOISE = 7;
const int IMG_SHARPEN = 8;

typedef struct operation {
    int code; //IMG_ROTATE
    std::string args; //"90"
    Magick::Blob blob; //Set if we can't revert
    /*
        Image my_image;
        Pixels my_pixel_cache(my_image);
        pixels=malloc(sizeof());
        pixels=my_pixel_cache;
    */
} operation_s;

typedef struct history{
    vector<operation> ops;
    int heavyOps; //счетчик тяжелых операций
    int pointer; //указатель на какой версии изображения мы сейчас
} history_s;

/**

```

					ПГУ1.090501.13.001 ПЗ	Лист
						122
Изм.	Лист	№ докум.	Подпись	Дата		

```

* Проверяет является ли операция "тяжелой"
* \note "Тяжелая" операция это та, для которой мы храним Blob
* \param[in] opcode - код операции
* \return true - тяжелая, false в других случаях
*/
bool isHeavy(int opcode);

class ImgLibrary{
private:
    string path;
    Magick::Image* img;
    history_s history;
    bool historyEnabled;
    /**
    * Отменяет последнее действие
    * \note Работает только с "легкими" операциями
    */
    void doOperation(operation_s op, bool mode);
    /**
    * Добавляет операцию в историю
    * \param[in] operation - добавляемая операция
    * \param[in] угол поворота
    */
    void addOperation(operation_s operation);
public:
    ImgLibrary();
    ImgLibrary(string _path);
    ~ImgLibrary();
    string getPath();
    /**
    * Отменяет последнее действие
    * \note Отменяемое действие берется из стека history
    */
    void undo();
    /**
    * Повторяет последнее действие

```

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		123

```

* \note Повторяемое действие добавляется в history
*/
void redo();
/**
Отменяет все преобразования
*/
void undoAll();
void save();
/**
* Проверяет наличие несохраненных изменений
* \return true, если есть несохраненные, иначе - false
*/
bool hasUnsavedChanges();
/**
* Поворачивает изображение
* \note Отменяемое действие
* \param[in] path путь к изображению
* \param[in] угол поворота
*/
void rotate(int degress);
/**
* Вертикально отразить
* \note Отменяемое действие
* \param[in] path путь к изображению
*/
void vertical();
/**
* Горизонтально отразить
* \note Отменяемое действие
* \param[in] path путь к изображению
*/
void horizontal();
/**
* Инвертировать цвета
* \note Отменяемое действие
* \param[in] path путь к изображению

```

```

*/
void invertColors();
/**
 * Задаёт яркость
 * \param[in] path путь к изображению
 * \param[in] значение яркости (от -100 до 100, 0 - знач. по
умолчанию)
*/
void setBrightness(int value);
/**
 * Изменяет контрастность
 * \note Произвольное значение задать нельзя!
 * \param[in] path путь к изображению
 * \param[in] value true - увеличить, false - уменьшить
контрастность
*/
void setContrast(bool value);
/**
 * Уменьшает шум
 * \param[in] path путь к изображению
 * \param[in] value 0 - изначально, 10 - максимум
*/
void reduceNoise(int value);
/**
 * Настройка резкости
 * \param[in] path путь к изображению
 * \param[in] value 0 - изначально, 10 - максимум
*/
void setSharpen(int value);
};

using ImgToolsManagerContainer = QMap<QString, ImgLibrary*>;

class ImgToolsManager{
private:
    ImgToolsManagerContainer* hashTable;

```

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		125

```

    ImgToolsManager();

    static ImgToolsManager* instance; ///< статический объект
    класса
public:
    static ImgToolsManager* getInstance();
    ~ImgToolsManager();
    ImgLibrary* get(QString path);
    void set(QString path, ImgLibrary* il);
    void saveAll();
    bool hasUnsavedChanges();
};

#endif

#include <iostream>
#include <QMessageBox>
#include "RadixController.h"
#include "thirdparty/easylogging++.h"
//for sleep
#include <chrono>
#include <thread>

//Change byteorder https://stackoverflow.com/a/2637138
/*
int getByteorder(){
    volatile uint32_t i=0x01234567;
    // return 0 for big endian, 1 for little endian.
    return (*((uint8_t*)&i)) == 0x67;
}
*/

//Radix uses little-endian

using namespace std;

uint16_t CRC16(std::vector<uint8_t> data, int length){
    int8_t i, j;

```

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		126

```

uint16_t CRC = 0xFFFF;
for (i = 0; i < length; i++){
    CRC ^= ((uint16_t)data[i]);
    for (j = 0; j < 8; j++){
        if ((CRC & 0x0001) == 1){
            CRC >>= 1;
            CRC ^= 0xA001;
        }
        else CRC >>= 1;
    }
}
return CRC;
}

RadixController::RadixController(){
    voltage = 0;
    time = 0;
    impulse = 0;
    freq = 0;
    rconf.minVoltage = 30;
    rconf.maxVoltage = 150;
    //_d = true;
    error = RADIX_NO_ERROR;
    //QMessageBox::information(this,"Radix","Setup begin");
    //Setup port
    vector<serial::PortInfo> ports = serial::list_ports();

    vector<serial::PortInfo>::iterator iter = ports.begin();

    //QString portName = "/dev/ttyS0";
    std::string portName = "COM10";
    if (ports.size()==0){
        LOG(INFO)<<"No available ports!\n";
        error = RADIX_PORTS_NOT_FOUND_ERROR;
    } else {
        LOG(INFO) << "We found something!";
    }
}

```

```

        LOG(INFO) << "Ports count:" << ports.size();
        //\\.\COM5
        while( iter != ports.end() )
        {
            serial::PortInfo device = *iter++;
            LOG(INFO)<<device.port<<" "<<device.description<<"
"<<device.hardware_id;
            /*printf( "(%s, %s, %s)\n", device.port.c_str(),
device.description.c_str(),
                device.hardware_id.c_str() );*/
        }
    }
    if (port.isOpen()) return;
    port.setTimeout(100, 250, 0, 250, 0);
    port.setBaudrate(115200);
    port.setPort(portName);
    //port.setBytesize(1024);//TODO: check value
    //port.setReadBufferSize(1024);//FIXME: check value
    try{
        port.open();
    }
    catch(serial::IOException e){
        LOG(INFO)<<"Port open error";
        LOG(INFO)<<e.what();
        error = RADIX_CONNECTION_ERROR;
    }

    //Not needed: default values:
    //port.setStopBits, port.setDataBits, port.setParity
}

/* Null, because instance will be initialized on demand. */
RadixController* RadixController::instance = 0;

RadixController* RadixController::getInstance()
{

```

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		128

```

    if (instance == 0){
        instance = new RadixController();
    }
    return instance;
}

PortResult RadixController::exp() {
    mutex.lock();
    std::vector<uint8_t> bytes;
    bytes.reserve(14);
    bytes[0] = sync_byte;
    bytes[1] = 0x0a; //length
    bytes[2] = 0x01; //cmd code
    bytes[3] = 0x01; //write
    bytes[4] = 66; //66%=100kV
    bytes[5] = 0;
    bytes[6] = 10; //freq, max 1000
    bytes[7] = 0;
    bytes[8] = 1; //pulse
    bytes[9] = 0;
    bytes[10] = 0; //delay
    bytes[11] = 255;
    bytes[12] = 0; //
    bytes[13] = 0;
    uint16_t crc = CRC16(bytes,11);
    bytes[12] = (uint8_t)(crc & 0xff);
    bytes[13] = (uint8_t)(crc >> 8);
    port.write(bytes);
    port.flush();
    std::vector<uint8_t> data;
    port.read(data,20);
    LOG(INFO)<<"execute exp!!!!!!!!!!!!!!!!!!";
    //LOG(INFO)<<(data.toHexString().c_str());
    mutex.unlock();
    return data;
}

```

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		129


```

RadixController::~RadixController() {
    LOG(INFO)<<"Closing RadixController port";
    port.close();
}

int RadixController::getExpoTime() {
    return impulse/freq;
}

PortResult RadixController::SetDelay(int16_t time) {
    this->time=time;
    std::vector<uint8_t> args;
    args.resize(2);
    args[0] = (uint8_t) (time & 0xff); //0x03
    args[1] = (uint8_t) (time>>8); //0x0
    return executeNew(Cmd_SetDelay,args,3);
}

PortResult RadixController::SetVoltage(int16_t voltage) {
    this->voltage=voltage;
    //Преобразуем киловольты в проценты
    PortResult args;
    if ( (voltage<rconf.minVoltage) ||
(voltage>rconf.maxVoltage) ) return args;
    voltage = (voltage/rconf.maxVoltage)*100; //(1.0*voltage/100);
    args.resize(2);
    args[0] = (uint8_t) (voltage & 0xff); //25
    args[1] = (uint8_t) (voltage>>8);
    return executeNew(Cmd_SetVoltage,args,3);
}

PortResult RadixController::SetImpulse(int16_t impulse) {
    this->impulse=impulse;
    PortResult args;
    args.resize(2);

```

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		130

```

    args[0] = (uint8_t) (impulse & 0xff); //57
    args[1] = (uint8_t) (impulse>>8);
    return executeNew(Cmd_SetImpulse,args,3);
}

PortResult RadixController::SetFrequency(int16_t freq){
    this->freq=freq;
    PortResult args;
    args.resize(2);
    args[0] = (uint8_t) (freq & 0xff); //57
    args[1] = (uint8_t) (freq>>8);
    return executeNew(Cmd_SetFrequency,args,3);
}

PortResult RadixController::executeNew(char commandId, bool
readData){
    mutex.lock();
    PortResult bytes;
    bytes.resize(5);
    bytes[0] = sync_byte;
    bytes[1] = 1; //length
    bytes[2] = commandId;
    bytes[3] = 0;
    bytes[4] = 0;
    uint16_t crc = CRC16(bytes,3);
    bytes[3] = (uint8_t) (crc & 0xff);
    bytes[4] = (uint8_t) (crc >> 8);
    try{
        port.write(bytes);
        port.flush();
    }
    catch(...) {
        LOG(WARNING)<<"Error while writing!";
        mutex.unlock();
        PortResult emptyData = {};
        return emptyData;
    }
}

```

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		131

```

    }

//std::this_thread::sleep_for(std::chrono::milliseconds(2200));
    if (!readData){
        mutex.unlock();
        return bytes;
    }
    PortResult data;
    data.resize(60);
    try{
        port.read(data, 60);
    }
    catch(...) {
        LOG(WARNING)<<"Error while reading!";
        mutex.unlock();
        PortResult emptyData = {};
        return emptyData;
    }
    LOG(INFO)<<"executeNew with 1 length command";
    std::string tmp;
    for (auto e: data){
        tmp.push_back(e);
    }
    LOG(INFO)<<"Response: "<<tmp;
    mutex.unlock();
    return data;
    //QString hex = data.toHex();
    //QFile f("bat.bin");
    f.open(QIODevice::WriteOnly);
    f.write(data);
    f.close();*/
    //return;
}

double RadixController::ReadBattery() {

```

					ПГУ1.090501.13.001 ПЗ	Лист
						132
Изм.	Лист	№ докум.	Подпись	Дата		

```

PortResult data = executeNew(Cmd_ReadBattery,true);
if (data.empty()) LOG(INFO)<<"Empty battery response";
//else LOG(INFO)<<data.toHex().toString();
constexpr int NUM_CELLS = 12;
int16_t cells[NUM_CELLS];
int u = 0;
int ptr = 4;
for (int i=0;i<12;i++){
    cells[i] = (int16_t) (data[ptr]&0xff);
    cells[i] |= (int16_t) (data[ptr+1]<<8);
    ptr+=2;
    std::cout << cells[i] << std::endl;
    u+=cells[i];
}
double ans = (2.0/105.0)*u-(14039/21);
LOG(INFO)<<"Battery: " <<to_string(ans);
return ans;
}

```

```

PortResult RadixController::executeNew(char commandId, PortResult
args, int length){
    //std::cout<<"Set Delay"
    mutex.lock();
    PortResult bytes;
    int total = 4+length; //4=sync_byte,length,2 bytes for crc
    bytes.resize(total); //4+3(command length)
    bytes[0] = sync_byte;
    bytes[1] = length;
    bytes[2] = commandId;
    for (int i=0;i<length;i++){
        bytes[3+i]=args[i];
    }
    bytes[total-2] = 0; //crc
    bytes[total-1] = 0; //crc
    uint16_t crc = CRC16(bytes,2+length); //5 is sync_byte,
command_length (2 bytes together) + command_length (3 bytes) = 5

```

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		133

```

char crc_l = (uint8_t) (crc&0xff);
char crc_m = (uint8_t) (crc>>8);
bytes[total-2] = crc_l;
bytes[total-1] = crc_m;
try{
    port.write(bytes);
    port.flush();
}
catch(...) {
    LOG(WARNING)<<"Error while writing!";
    mutex.unlock();
    PortResult emptyData = {};
    return emptyData;
}
//const char* m = "Sleeping for 2200 ms";
//std::cout<<m<<std::end;
PortResult readedData;
readedData.resize(total);
try{
    port.read(readedData,total);
}
catch(...) {
    LOG(WARNING)<<"Error while writing!";
    mutex.unlock();
    PortResult emptyData = {};
    return emptyData;
}
LOG(INFO)<<("executeNew data!!!!!!!!!!!!!!!!\n");
//LOG(INFO)<<(readedData.toHex().toStdString().c_str());
//qDebug() << readedData.toHex();
mutex.unlock();
return readedData;
//TODO: get the result
}

```

```

PortResult RadixController::StartXRay() {
    LOG(INFO) << ("Start XRAY");
    return executeNew(Cmd_StartXRay, false);
}

PortResult RadixController::StopXRay() {
    return executeNew(Cmd_StopXRay, true);
}

PortResult RadixController::PowerOff() {
    return executeNew(Cmd_PowerOff1, false);
}

#pragma once
#ifndef RADIX_H
#define RADIX_H

#include <iostream>
#include <QMutex>
#include <serial/serial.h>
//for sleep
#include <chrono>
#include <thread>

typedef std::vector<uint8_t> PortResult;

//Change byteorder https://stackoverflow.com/a/2637138
/*
int getByteorder() {
    volatile uint32_t i=0x01234567;
    // return 0 for big endian, 1 for little endian.
    return (*((uint8_t*)&i)) == 0x67;
}
*/

//Radix uses little-endian

```

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		135

```

//typedef void ExecutionResult;

//char sync_byte = 0x55;

#define sync_byte 0x55

#define RADIX_NO_ERROR 0
#define RADIX_PORTS_NOT_FOUND_ERROR 1
#define RADIX_CONNECTION_ERROR 2

//const int Cmd_PwCFG = 0x05; ???
const int Cmd_ReadBattery = 0x06; //XRAY_GET_BATT
const int Cmd_SetVoltage = 0x07;
const int Cmd_SetFrequency = 0x08;
const int Cmd_SetDelay = 0x09;
const int Cmd_SetImpulse = 0x0A;
const int Cmd_StartXRay = 0x0B;
const int Cmd_StopXRay = 0x0C;
//const int Cmd_ReadTemperature
const char Cmd_PowerOff1 = 0x0D;
const int Cmd_SetPin = 0x0E; //XRAY_PIN_CFG
const int Cmd_Sensors = 0x0F; //Temperature??
const int Cmd_RequestLog = 0x10;
/* ???
const byte SAVE_TO_FLASH = 0x17;
const byte CMD_EXTERNAL_TRIGGER_CFG = 0x30;
const byte CMD_XRAY_GEN_CONTINUES = 0x31;
const byte CMD_PULSES_TOTAL = 0x32;
*/

const int Err_NotImplemented1 = 0x81;
const int Err_CRC = 0x82;
const int Err_InvalidParam = 0x83;
const int Err_Ok = 0xAA;

uint16_t CRC16(PortResult data, int length);

```

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		136

```

void TRACE1(const char* x);

/**
 * \brief Код, управляющий радиксом
 * Реализуется паттерн синглтон, см. метод getInstance
 * Сделан с помощью библиотеки serial
 * \note Потокбезопасный: внутри мьютекс
 * \author Марат Нагаев
 *
 * Этот класс позволяет управлять Радиксом - посылать ему команды
и получать результат.
 */

typedef struct radixconfig {
    int minVoltage;
    int maxVoltage;
} radixconfig_s;

class RadixController{
private:
    radixconfig_s rconf;
    QMutex mutex; ///< мьютекс, защищающий отправку команд
    serial::Serial port; ///< порт, на котором "висит" радикс
    static RadixController* instance; ///< статический объект
    класса
    RadixController();
public:
    int voltage,time,impulse,freq; ///< актуальные значения
    параметров
    ~RadixController();
    int error; ///< поле, описывающее последнюю ошибку
    /**
     * Метод доступа к объекту
     * \return Статический объект класса RadixController
     */
    static RadixController* getInstance();

```

					ПГУ1.090501.13.001 ПЗ	Лист
						137
Изм.	Лист	№ докум.	Подпись	Дата		


```

PortResult exp();
/**
 * Возвращает время экспозиции в мс
 * \note Функцию можно вызывать только после установки
параметров!
 * Иначе возможно исключение (деление на 0)
 * \return Экспозиция (импульсы/частота)
 */
int getExpoTime();
/**
 * Получает заряд батареи
 * \return заряд батареи в процентах
 */
double ReadBattery();
/**
 * Устанавливает напряжение
 * \param[in] voltage Напряжение в кВ от 30 до 150
 * \return Ответ радикса в виде vector<uin8_t>
 */
PortResult SetVoltage(int16_t voltage);
/**
 * Устанавливает задержку
 * \param[in] time Задержка от 2с
 * \return Ответ радикса в виде vector<uin8_t>
 */
PortResult SetDelay(int16_t time);
/**
 * Устанавливает кол-во импульсов
 * \param[in] impulse кол-во импульсов
 * \return Ответ радикса в виде vector<uin8_t>
 */
PortResult SetImpulse(int16_t impulse);
/**
 * Устанавливает частоту
 * \param[in] freq Частота в Гц
 * \return Ответ радикса в виде vector<uin8_t>

```

```

*/
PortResult SetFrequency(int16_t freq);
/**
* Запускает генерацию
* \warning Убедитесь в отсутствии людей поблизости!
* \note Ответ - заглушка, т.к мы не можем ждать получения
данных
* \return Ответ радикса в виде vector<uin8_t>
*/
PortResult StartXRay();
/**
* Останавливает генерацию
* \note Команда используется только для предотвращения аварии
* Генерация прекращается сама
* \return Ответ радикса в виде vector<uin8_t>
*/
PortResult StopXRay();
/**
* Выключает радикс
* \return Возвращаемое значение отсутствует
*/
PortResult PowerOff();
/**
* Посылает команду радиксу
* \note Потокбезопасная: внутри мьютекс
* \param[in] commandId id команды (число)
* \param[in] args Аргументы - массив байт
* \param[in] length кол-во аргументов
* \return Ответ радикса в виде vector<uin8_t>
*/
PortResult executeNew(char commandId, PortResult args, int
length);
/**
* Посылает радиксу команду длиной 1 байт
* \param[in] commandId id команды (число)
* \param[in] читать данные или нет

```

```

* \note Мы не читаем ответ радикаса когда запускаем генерацию
* \return Ответ радикаса в виде vector<uin8_t>
*/

PortResult executeNew(char commandId, bool readData);
};

#endif

#include "dcmtk/ofstd/offile.h"
#include "dcmtk/dcndata/dcfilefo.h" //DcmFileFormat
#include <dcmtk/dcmimage/dcmimage.h> //DicomImage
#include <dcmtk/dcndata/libi2d/i2d.h>
#include "dcmtk/dcndata/libi2d/i2dplsc.h"
#include "dcmtk/dcndata/libi2d/i2djpgs.h" //JpegSource
#include "dcmtk/dcmjpeg/dipijpeg.h" //DiJpegPlugin
#include <dcmtk/dcndata/dctk.h> //defines like DCM_PixelData
//#include <dcmtk/dcndata/libi2d/i2djpgs.h>
#include <QDir>
#include <thirdparty/easylogging++.h>
#include <Windows.h>
#include "utils.h"

bool dcm2jpg(string dicomPath, string saveAs){
    LOG(INFO)<<"dcm2img: "<<dicomPath<<" "<<saveAs;
    DcmFileFormat fileformat;
    const char * fname = dicomPath.c_str();
    OFCondition status = fileformat.loadFile(fname);
    DcmDataset *dataset;
    dataset = fileformat.getDataset();
    //dataset-
    >putAndInsertString(DCM_PhotometricInterpretation,"MONOCHROME2");
    if (!status.good()) return false;
    LOG(INFO)<<"before complain";
    DicomImage lNewImg(dataset, dataset->getOriginalXfer());
    lNewImg.setMinMaxWindow(0); //NOTE: Enable minmax
    //lNewImg.writePng(&saveAs[0]);
    DiJPEGPlugin *plugin = new DiJPEGPlugin();
    unsigned int opt_quality = 100;//default value is 90

```

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		140

```

    plugin->setQuality(opt_quality);
    lNewImg.writePluginFormat(plugin, saveAs.c_str());
    LOG(INFO)<<"after";
    return true;
}

bool img2dcm(string input, string output){
    LOG(INFO)<<"img2dcm: "<<input<<" "<<output;
    Image2Dcm i2d;
    OFCondition cond;
    I2DImgSource* inputPlug = new I2DJpegSource();
    //I2DImgSource* inputPlug = new I2DJpgSource();
    //see
https://github.com/InsightSoftwareConsortium/DCMTK/blob/master/dcmdata/apps/img2dcm.cc#L480
    I2DOutputPlug* outPlug = new I2DOutputPlugSC(); //legacy
    plugin
        inputPlug->setImageFile(&input[0]); // we are guaranteed to
    have at least one input file
    E_TransferSyntax writeXfer; //FIXME
    DcmDataset* resultDset = NULL;
    cond = i2d.convertFirstFrame(inputPlug, outPlug, 1,
    resultDset, writeXfer);
    if(cond.bad()){
        return false;
    }
    DcmFileFormat dcmff(resultDset);
    cond = dcmff.saveFile(output.c_str(), writeXfer,
    EET_ExplicitLength, EGL_recalcGL, EPD_noChange,
    OFstatic_cast(Uint32, 0), OFstatic_cast(Uint32, 0),
    EWM_fileformat);
    string command = "gdcconv --raw " + output + " " + output;
    WinExec(command.c_str(), SW_HIDE); //FIXME: dirty hack
    return cond.good();
}

```

```

string getDcmFromImgPath(string path){
    string dicomPath =
changeExtension(path,"dcm");  //.converted/1.dcm
    string converted = ".converted/";
    int index = dicomPath.find(converted);
    dicomPath=dicomPath.replace(index,converted.size(),"");
    return dicomPath;
}

QString getDcmFromImgPath(QString path){
    QString dicomPath = path.replace(".converted","");
    dicomPath = changeExtension(dicomPath, "dcm");
    return dicomPath;
}

std::string changeExtension(std::string filename, std::string ext)
{
    return filename.substr(0,filename.find_last_of('.')+1)+ext;
}

QString changeExtension(QString filename, QString ext){
    return filename.mid(0,filename.lastIndexOf('.')+1)+ext;
}

QString appendPath(const QString& path1, const QString& path2){
    return QDir::cleanPath(path1 + '/' + path2);
}

#ifdef UTILS1_H
#define UTILS1_H

#include <string>
#include <QString>

using std::string;

bool dcm2jpg(string dicom_path, string saveAs);

```

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		142

```

bool img2dcm(string input, string output);
QString getDcmFromImgPath(QString path);
string getDcmFromImgPath(string path);
string changeExtension(string filename, string ext);
QString changeExtension(QString filename, QString ext);
QString appendPath(const QString& path1, const QString& path2);

#endif
#include "RHardware.h"

RHardware::RHardware(QWidget *parent) : QWidget(parent)
{
    setupUi(this);
}

void RHardware::setupUi(QWidget *RHardware)
{
    if (RHardware->objectName().isEmpty())
        RHardware->setObjectName(QString::fromUtf8("RHardware"));
    RHardware->resize(260, 1069);
    RHardware->setMinimumSize(QSize(260, 600));
    RHardware->setMaximumSize(QSize(260, 16777215));
    RHardware->setStyleSheet(QString::fromUtf8("
        *{
            background-color: #212121;
            border-radius: 10px;
            font-size: 14px;
            font-family: Arial;
            color: #CFCFCF;
        }
    "));
}

```

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		143

```

"\\n"
"QSpinBox {\\n"
"    border: 0px

solid #ccc;\\n"

" background-

color: #313131;\\n"

" border-radius:

10px;\\n"

" color:#CFCFCF;

\\n"

" padding: 0px

10px;\\n"

"}\\n"
"QSpinBox::up-

button, QSpinBox::down-button {\\n"

"    width: 0; \\n"
"    height: 0;\\n"
"    border: none;

"    background:

transparent; \\n"

"}\\n"
"\\n"
"QSpinBox::up-

button::down-arrow {\\n"

"    content: "\\n";

"}\\n"
"\\n"
"QSpinBox::down-

button::down-arrow {\\n"

"    content: "\\n";

"}\\n"
"\\n"
"QPushButton{

```

					ПГУ1.090501.13.001 ПЗ	Лист
						144
Изм.	Лист	№ докум.	Подпись	Дата		

```
\n"

color:#313131;\n"

\n"
```

```
" background-

" color:#CFCFCF;

"}\n"

"\n"

"QPushButton{
```

```
\n"

color:#4A4A4A;\n"

\n"
```

```
" background-

" color:#CFCFCF;

"}\n"

"\n"

"\n"
```

```
"QSplitter::handle:horizontal {\n"

color: #4A4A4A; \n"
```

```
" background-

" width: 2px;\n"

"}\n"

"\n"
```

```
"QSplitter::handle:vertical {\n"

color: #4A4A4A;\n"

\n"
```

```
" background-

" height: 2px;

"}\n"

"\n"

"\n"

"#vtgCtrlWidget,
```

```
#impCtrlWidget, #freqCtrlWidget, #de"
```

```
"layCtrlWidget{\n"

" background-
```

					ПГУ1.090501.13.001 ПЗ	Лист
						145
Изм.	Лист	№ докум.	Подпись	Дата		


```

color: #313131;\n"

                                "}\n"
                                "\n"
                                "\n"
                                "#vtgUnitLabel,
#impUnitLabel,#freqUnitLabel,#delayUnitLabel {\n"
                                " background-
color:transparent;\n"

                                "}\n"
                                "\n"
                                "QLabel {\n"
                                " font-

weight:bold;\n"

                                "}\n"
                                ""));

    QVBoxLayout = new QVBoxLayout(RHardware);
    QVBoxLayout-
>setObjectName(QString::fromUtf8("verticalLayout"));
    QVBoxLayout->setContentsMargins(10, 10, 10, 10);
    splitter = new QSplitter(RHardware);
    splitter->setObjectName(QString::fromUtf8("splitter"));
    splitter->setOrientation(Qt::Vertical);
    voltageWidget = new QWidget(splitter);
    voltageWidget-
>setObjectName(QString::fromUtf8("voltageWidget"));
    voltageWidget->setMinimumSize(QSize(0, 100));
    voltageWidget->setMaximumSize(QSize(16777215, 100));
    QVBoxLayout_2 = new QVBoxLayout(voltageWidget);
    QVBoxLayout_2->setSpacing(10);
    QVBoxLayout_2-
>setObjectName(QString::fromUtf8("verticalLayout_2"));
    QVBoxLayout_2->setContentsMargins(0, 10, 0, 10);
    vtgLabel = new QLabel(voltageWidget);
    vtgLabel->setObjectName(QString::fromUtf8("vtgLabel"));

    QVBoxLayout_2->addWidget(vtgLabel);

```

```

    vtgCtrlWidget = new QFrame(voltageWidget);
    vtgCtrlWidget-
>setObjectName(QString::fromUtf8("vtgCtrlWidget"));
    vtgCtrlWidget->setMinimumSize(QSize(50, 50));
    vtgCtrlWidget->setMaximumSize(QSize(16777215, 50));
    horizontalLayout_2 = new QHBoxLayout(vtgCtrlWidget);
    horizontalLayout_2-
>setObjectName(QString::fromUtf8("horizontalLayout_2"));
    horizontalLayout_2->setContentsMargins(0, 0, 0, 0);
    vtgSpinBox = new QSpinBox(vtgCtrlWidget);
    vtgSpinBox->setObjectName(QString::fromUtf8("vtgSpinBox"));
    vtgSpinBox->setMinimumSize(QSize(80, 30));
    vtgSpinBox->setMaximumSize(QSize(80, 30));
    vtgSpinBox->setMaximum(10000);

    horizontalLayout_2->addWidget(vtgSpinBox);

    vtgUnitLabel = new QLabel(vtgCtrlWidget);
    vtgUnitLabel-
>setObjectName(QString::fromUtf8("vtgUnitLabel"));
    vtgUnitLabel->setMinimumSize(QSize(35, 30));
    vtgUnitLabel->setMaximumSize(QSize(35, 30));

    horizontalLayout_2->addWidget(vtgUnitLabel);

    vtgDecreaseBtn = new QPushButton(vtgCtrlWidget);
    vtgDecreaseBtn-
>setObjectName(QString::fromUtf8("vtgDecreaseBtn"));
    vtgDecreaseBtn->setMinimumSize(QSize(30, 30));
    vtgDecreaseBtn->setMaximumSize(QSize(30, 30));

    horizontalLayout_2->addWidget(vtgDecreaseBtn);

    impIncreaseBtn = new QPushButton(vtgCtrlWidget);
    impIncreaseBtn-

```

```

>setObjectName(QString::fromUtf8("impIncreaseBtn"));
    impIncreaseBtn->setMinimumSize(QSize(30, 30));
    impIncreaseBtn->setMaximumSize(QSize(30, 30));

    horizontalLayout_2->addWidget(impIncreaseBtn);

    verticalLayout_2->addWidget(vtgCtrlWidget);

    splitter->addWidget(voltageWidget);
    impWidget = new QWidget(splitter);
    impWidget->setObjectName(QString::fromUtf8("impWidget"));
    impWidget->setMinimumSize(QSize(0, 100));
    impWidget->setMaximumSize(QSize(16777215, 100));
    verticalLayout_3 = new QVBoxLayout(impWidget);
    verticalLayout_3->setSpacing(10);
    verticalLayout_3-
>setObjectName(QString::fromUtf8("verticalLayout_3"));
    verticalLayout_3->setContentsMargins(0, 10, 0, 10);
    impLabel = new QLabel(impWidget);
    impLabel->setObjectName(QString::fromUtf8("impLabel"));

    verticalLayout_3->addWidget(impLabel);

    impCtrlWidget = new QFrame(impWidget);
    impCtrlWidget-
>setObjectName(QString::fromUtf8("impCtrlWidget"));
    impCtrlWidget->setMinimumSize(QSize(50, 50));
    impCtrlWidget->setMaximumSize(QSize(16777215, 50));
    horizontalLayout_4 = new QHBoxLayout(impCtrlWidget);
    horizontalLayout_4->setSpacing(10);
    horizontalLayout_4-
>setObjectName(QString::fromUtf8("horizontalLayout_4"));
    horizontalLayout_4->setContentsMargins(0, 0, 0, 0);
    impSpinBox = new QSpinBox(impCtrlWidget);
    impSpinBox->setObjectName(QString::fromUtf8("impSpinBox"));

```

```

impSpinBox->setMinimumSize(QSize(80, 30));
impSpinBox->setMaximumSize(QSize(80, 30));
impSpinBox->setMaximum(10000);

horizontalLayout_4->addWidget(impSpinBox);

impUnitLabel = new QLabel(impCtrlWidget);
impUnitLabel-
>setObjectName(QString::fromUtf8("impUnitLabel"));
impUnitLabel->setMinimumSize(QSize(35, 0));
impUnitLabel->setMaximumSize(QSize(35, 16777215));

horizontalLayout_4->addWidget(impUnitLabel);

impDecreaseBtn = new QToolButton(impCtrlWidget);
impDecreaseBtn-
>setObjectName(QString::fromUtf8("impDecreaseBtn"));
impDecreaseBtn->setMinimumSize(QSize(30, 30));
impDecreaseBtn->setMaximumSize(QSize(30, 30));

horizontalLayout_4->addWidget(impDecreaseBtn);

impIncreaseBtn_2 = new QToolButton(impCtrlWidget);
impIncreaseBtn_2-
>setObjectName(QString::fromUtf8("impIncreaseBtn_2"));
impIncreaseBtn_2->setMinimumSize(QSize(30, 30));
impIncreaseBtn_2->setMaximumSize(QSize(30, 30));

horizontalLayout_4->addWidget(impIncreaseBtn_2);

verticalLayout_3->addWidget(impCtrlWidget);

splitter->addWidget(impWidget);
freqWidget = new QWidget(splitter);
freqWidget->setObjectName(QString::fromUtf8("freqWidget"));

```

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		149

```

    freqWidget->setMinimumSize(QSize(0, 100));
    freqWidget->setMaximumSize(QSize(16777215, 100));
    verticalLayout_4 = new QVBoxLayout(freqWidget);
    verticalLayout_4->setSpacing(10);
    verticalLayout_4-
>setObjectName(QString::fromUtf8("verticalLayout_4"));
    verticalLayout_4->setContentsMargins(0, 10, 0, 10);
    freqLabel = new QLabel(freqWidget);
    freqLabel->setObjectName(QString::fromUtf8("freqLabel"));

    verticalLayout_4->addWidget(freqLabel);

    freqCtrlWidget = new QFrame(freqWidget);
    freqCtrlWidget-
>setObjectName(QString::fromUtf8("freqCtrlWidget"));
    freqCtrlWidget->setMinimumSize(QSize(50, 50));
    freqCtrlWidget->setMaximumSize(QSize(16777215, 50));
    horizontalLayout_3 = new QHBoxLayout(freqCtrlWidget);
    horizontalLayout_3->setSpacing(10);
    horizontalLayout_3-
>setObjectName(QString::fromUtf8("horizontalLayout_3"));
    horizontalLayout_3->setContentsMargins(0, 0, 0, 0);
    freqSpinBox = new QSpinBox(freqCtrlWidget);
    freqSpinBox->setObjectName(QString::fromUtf8("freqSpinBox"));
    freqSpinBox->setMinimumSize(QSize(80, 30));
    freqSpinBox->setMaximumSize(QSize(80, 30));
    freqSpinBox->setMaximum(10000);

    horizontalLayout_3->addWidget(freqSpinBox);

    freqUnitLabel = new QLabel(freqCtrlWidget);
    freqUnitLabel-
>setObjectName(QString::fromUtf8("freqUnitLabel"));
    freqUnitLabel->setMinimumSize(QSize(35, 0));
    freqUnitLabel->setMaximumSize(QSize(35, 16777215));

```

```

horizontalLayout_3->addWidget(freqUnitLabel);

freqDecreaseBtn = new QPushButton(freqCtrlWidget);
freqDecreaseBtn-
>setObjectName(QString::fromUtf8("freqDecreaseBtn"));
freqDecreaseBtn->setMinimumSize(QSize(30, 30));
freqDecreaseBtn->setMaximumSize(QSize(30, 30));

horizontalLayout_3->addWidget(freqDecreaseBtn);

freqIncreaseBtn = new QPushButton(freqCtrlWidget);
freqIncreaseBtn-
>setObjectName(QString::fromUtf8("freqIncreaseBtn"));
freqIncreaseBtn->setMinimumSize(QSize(30, 30));
freqIncreaseBtn->setMaximumSize(QSize(30, 30));

horizontalLayout_3->addWidget(freqIncreaseBtn);

verticalLayout_4->addWidget(freqCtrlWidget);

splitter->addWidget(freqWidget);
delayWidget = new QWidget(splitter);
delayWidget->setObjectName(QString::fromUtf8("delayWidget"));
delayWidget->setMinimumSize(QSize(0, 100));
delayWidget->setMaximumSize(QSize(16777215, 100));
verticalLayout_6 = new QVBoxLayout(delayWidget);
verticalLayout_6->setSpacing(10);
verticalLayout_6-
>setObjectName(QString::fromUtf8("verticalLayout_6"));
verticalLayout_6->setContentsMargins(0, 10, 0, 10);
delayLabel = new QLabel(delayWidget);
delayLabel->setObjectName(QString::fromUtf8("delayLabel"));

verticalLayout_6->addWidget(delayLabel);

```

```

        delayCtrlWidget = new QFrame(delayWidget);
        delayCtrlWidget-
>setObjectName(QString::fromUtf8("delayCtrlWidget"));
        delayCtrlWidget->setMinimumSize(QSize(50, 50));
        delayCtrlWidget->setMaximumSize(QSize(16777215, 50));
        horizontalLayout_5 = new QHBoxLayout(delayCtrlWidget);
        horizontalLayout_5->setSpacing(10);
        horizontalLayout_5-
>setObjectName(QString::fromUtf8("horizontalLayout_5"));
        horizontalLayout_5->setContentsMargins(0, 0, 0, 0);
        delaySpinBox = new QSpinBox(delayCtrlWidget);
        delaySpinBox-
>setObjectName(QString::fromUtf8("delaySpinBox"));
        delaySpinBox->setMinimumSize(QSize(80, 30));
        delaySpinBox->setMaximumSize(QSize(80, 30));
        delaySpinBox->setMaximum(10000);

        horizontalLayout_5->addWidget(delaySpinBox);

        delayUnitLabel = new QLabel(delayCtrlWidget);
        delayUnitLabel-
>setObjectName(QString::fromUtf8("delayUnitLabel"));
        delayUnitLabel->setMinimumSize(QSize(35, 0));
        delayUnitLabel->setMaximumSize(QSize(35, 16777215));

        horizontalLayout_5->addWidget(delayUnitLabel);

        delayDecreaseBtn = new QToolButton(delayCtrlWidget);
        delayDecreaseBtn-
>setObjectName(QString::fromUtf8("delayDecreaseBtn"));
        delayDecreaseBtn->setMinimumSize(QSize(30, 30));
        delayDecreaseBtn->setMaximumSize(QSize(30, 30));

        horizontalLayout_5->addWidget(delayDecreaseBtn);

        delayIncreaseBtn = new QToolButton(delayCtrlWidget);

```

```

    delayIncreaseBtn-
>setObjectName(QString::fromUtf8("delayIncreaseBtn"));
    delayIncreaseBtn->setMinimumSize(QSize(30, 30));
    delayIncreaseBtn->setMaximumSize(QSize(30, 30));

    horizontalLayout_5->addWidget(delayIncreaseBtn);

    verticalLayout_6->addWidget(delayCtrlWidget);

    splitter->addWidget(delayWidget);
    panelSettingWidget = new QWidget(splitter);
    panelSettingWidget-
>setObjectName(QString::fromUtf8("panelSettingWidget"));
    panelSettingWidget->setMinimumSize(QSize(0, 100));
    panelSettingWidget->setMaximumSize(QSize(16777215, 100));
    verticalLayout_8 = new QVBoxLayout(panelSettingWidget);
    verticalLayout_8->setSpacing(10);
    verticalLayout_8-
>setObjectName(QString::fromUtf8("verticalLayout_8"));
    verticalLayout_8->setContentsMargins(0, 10, 0, 10);
    flatPanelLabel = new QLabel(panelSettingWidget);
    flatPanelLabel-
>setObjectName(QString::fromUtf8("flatPanelLabel"));
    QFont font;
    font.setFamily(QString::fromUtf8("Arial"));
    font.setBold(true);
    font.setWeight(75);
    flatPanelLabel->setFont(font);

    verticalLayout_8->addWidget(flatPanelLabel);

    flatPanelSettingsBtn = new QPushButton(panelSettingWidget);
    flatPanelSettingsBtn-
>setObjectName(QString::fromUtf8("flatPanelSettingsBtn"));
    flatPanelSettingsBtn->setMinimumSize(QSize(200, 40));

```



```

flatPanelSettingsBtn->setMaximumSize(QSize(16777215, 40));

verticalLayout_8->addWidget(flatPanelSettingsBtn);

splitter->addWidget(panelSettingWidget);
snapContainer = new QWidget(splitter);
snapContainer-
>setObjectName(QString::fromUtf8("snapContainer"));
verticalLayout_7 = new QVBoxLayout(snapContainer);
verticalLayout_7-
>setObjectName(QString::fromUtf8("verticalLayout_7"));
verticalLayout_7->setContentsMargins(0, 10, -1, 10);
verticalSpacer = new QSpacerItem(20, 181,
QSizePolicy::Minimum, QSizePolicy::Expanding);

verticalLayout_7->addItem(verticalSpacer);

makeSnapshotBtn = new QPushButton(snapContainer);
makeSnapshotBtn-
>setObjectName(QString::fromUtf8("makeSnapshotBtn"));
makeSnapshotBtn->setMinimumSize(QSize(200, 40));
makeSnapshotBtn->setMaximumSize(QSize(16777215, 40));

verticalLayout_7->addWidget(makeSnapshotBtn);

splitter->addWidget(snapContainer);

verticalLayout->addWidget(splitter);

retranslateUi(RHardware);
QObject::connect(vtgDecreaseBtn, SIGNAL(clicked()),
vtgSpinBox, SLOT(stepDown()));
QObject::connect(impIncreaseBtn, SIGNAL(clicked()),
vtgSpinBox, SLOT(stepUp()));
QObject::connect(impDecreaseBtn, SIGNAL(clicked()),

```

```

impSpinBox, SLOT(stepDown()));
    QObject::connect(impIncreaseBtn_2, SIGNAL(clicked()),
impSpinBox, SLOT(stepUp()));
    QObject::connect(freqDecreaseBtn, SIGNAL(clicked()),
freqSpinBox, SLOT(stepDown()));
    QObject::connect(freqIncreaseBtn, SIGNAL(clicked()),
freqSpinBox, SLOT(stepUp()));
    QObject::connect(delayDecreaseBtn, SIGNAL(clicked()),
delaySpinBox, SLOT(stepDown()));
    QObject::connect(delayIncreaseBtn, SIGNAL(clicked()),
delaySpinBox, SLOT(stepUp()));

    QMetaObject::connectSlotsByName(RHardware);
}

void RHardware::retranslateUi(QWidget *RHardware)
{
    RHardware->
>setWindowTitle(QCoreApplication::translate("RHardware", "Form",
nullptr));
    vtgLabel->setText(QCoreApplication::translate("RHardware",
"\320\235\320\260\320\277\321\200\321\217\320\266\320\265\320\275\
320\270\320\265", nullptr));
    vtgUnitLabel->setText(QCoreApplication::translate("RHardware",
"kB", nullptr));
    vtgDecreaseBtn->
>setText(QCoreApplication::translate("RHardware", "<", nullptr));
    impIncreaseBtn->
>setText(QCoreApplication::translate("RHardware", ">", nullptr));
    impLabel->setText(QCoreApplication::translate("RHardware",
"\320\230\320\274\320\277\321\203\320\273\321\214\321\201\321\213"
, nullptr));
    impUnitLabel->setText(QCoreApplication::translate("RHardware",
"\320\230\320\274\320\277", nullptr));
    impDecreaseBtn->
>setText(QCoreApplication::translate("RHardware", "<", nullptr));

```

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		155

```

        impIncreaseBtn_2-
>setText(QCoreApplication::translate("RHardware", ">", nullptr));
        freqLabel->setText(QCoreApplication::translate("RHardware",
"\320\247\320\260\321\201\321\202\320\276\321\202\320\260",
nullptr));
        freqUnitLabel-
>setText(QCoreApplication::translate("RHardware",
"\320\223\321\206", nullptr));
        freqDecreaseBtn-
>setText(QCoreApplication::translate("RHardware", "<", nullptr));
        freqIncreaseBtn-
>setText(QCoreApplication::translate("RHardware", ">", nullptr));
        delayLabel->setText(QCoreApplication::translate("RHardware",
"\320\227\320\260\320\264\320\265\321\200\320\266\320\272\320\260"
, nullptr));
        delayUnitLabel-
>setText(QCoreApplication::translate("RHardware",
"\321\201\320\265\320\272", nullptr));
        delayDecreaseBtn-
>setText(QCoreApplication::translate("RHardware", "<", nullptr));
        delayIncreaseBtn-
>setText(QCoreApplication::translate("RHardware", ">", nullptr));
        flatPanelLabel-
>setText(QCoreApplication::translate("RHardware",
"\320\237\320\273\320\276\321\201\320\272\320\276\320\277\320\260\
320\275\320\265\320\273\321\214\320\275\321\213\320\271
\320\264\320\265\321\202\320\265\320\272\321\202\320\276\321\200",
nullptr));
        flatPanelSettingsBtn-
>setText(QCoreApplication::translate("RHardware",
"\320\235\320\260\321\201\321\202\321\200\320\276\320\271\320\272\
320\270
\320\277\320\276\320\264\320\272\320\273\321\216\321\207\320\265\3
20\275\320\270\321\217", nullptr));
        makeSnapshotBtn-
>setText(QCoreApplication::translate("RHardware",

```

```

"\320\241\320\264\320\265\320\273\320\260\321\202\321\214
\321\201\320\275\320\270\320\274\320\276\320\272", nullptr));
}
#include "components/RManipulations.h"
#include <Magick++.h>
#include "src/ImgLibrary.h"
#include "RChanger.h"
#include "RExaminationMain.h"
#include "screens/examination_screen.h"
#include "thirdparty/easylogging++.h"

RManipulations::RManipulations(QWidget *parent) : QWidget(parent)
{
    setupUi(this);

    connect(rotateBtn, &QPushButton::released, this, [] () {
        LOG(INFO) << "Rotating...";
        RExaminationViewer* element =
RExaminationViewer::getSelectedWidget();
        QString imagePath = element->getImagePath();
        ImgLibrary* imgTool = ImgToolsManager::getInstance()-
>get(imagePath);
        imgTool->rotate(-90);
        element->reloadImage();
    });

    connect(verticalFlipBtn, &QPushButton::released, this, [] () {
        LOG(INFO) << "Vertical flip";
        RExaminationViewer* element =
RExaminationViewer::getSelectedWidget();
        QString imagePath = element->getImagePath();
        ImgLibrary* imgTool = ImgToolsManager::getInstance()-
>get(imagePath);
        imgTool->vertical();
        element->reloadImage();
    });
}

```

					ПГУ1.090501.13.001 ПЗ	Лист
						157
Изм.	Лист	№ докум.	Подпись	Дата		

```

});

connect(horizontalFlipBtn, &QPushButton::released, this, [] () {
    LOG(INFO) << "Horizontal flip...";
    RExaminationViewer* element =
RExaminationViewer::getSelectedWidget();
    QString imagePath = element->getImagePath();
    ImgLibrary* imgTool = ImgToolsManager::getInstance()-
>get(imagePath);
    imgTool->horizontal();
    element->reloadImage();
});

connect(colorInverseBtn, &QPushButton::released, this, [] () {
    LOG(INFO) << "Inverting...";
    RExaminationViewer* element =
RExaminationViewer::getSelectedWidget();
    QString imagePath = element->getImagePath();
    ImgLibrary* imgTool = ImgToolsManager::getInstance()-
>get(imagePath);
    imgTool->invertColors();
    element->reloadImage();
});

connect(cancelBtn, &QPushButton::released, this, [] () {
    LOG(INFO) << "Cancel all";
    RExaminationViewer* element =
RExaminationViewer::getSelectedWidget();
    QString imagePath = element->getImagePath();
    ImgLibrary* imgTool = ImgToolsManager::getInstance()-
>get(imagePath);
    imgTool->undoAll();
    element->reloadImage();
});
}

//report.exe на Golang:

```

					ПГУ1.090501.13.001 ПЗ	Лист
						158
Изм.	Лист	№ докум.	Подпись	Дата		

```

package main

import (
    "os"
    "encoding/json"
    "fmt"

    "github.com/fumiama/go-docx"
)

type ExaminationType struct {
    ImagesCount int
    Patient string
    Doctor string
    Dose float64
}

type PatologyType struct {
    Type string
    Location string
    Args []string
}

type BiradsType struct {
    Examination ExaminationType
    Indication string
    Density string
    Pathologies []PatologyType
    BiradsNumber string
    Conclusion string
}

func readJSON(data *BiradsType){
    f, err := os.Open("go.json")
    if err != nil {
        fmt.Println(err)
    }
}

```

					ПГУ1.090501.13.001 ПЗ	Лист
						159
Изм.	Лист	№ докум.	Подпись	Дата		

```

    }
    defer f.Close()

    err = json.NewDecoder(f).Decode(&data)
    if err != nil {
        fmt.Println(err)
    }
    fmt.Println(data.Indication)
    //fmt.Printf("%+v", data)
}

func main() {
    var data BiradsType
    readJSON(&data)
    w := docx.New().WithDefaultTheme().WithA4Page()
    //Заголовок
    para1 := w.AddParagraph()
    para1.AddText("Маммография").Size("24")
    //Адрес
    para2 := w.AddParagraph().Justification("center")
    para2.AddText("ГБУЗ «Клиническая больница № 6 имени Г. А.
Захарьина»\n").Size("24")
    para2.AddText(" 440071, г. Пенза, ул. Стасова, 7, тел. +7
(8412) 98-45-25\n").Size("24")
    para2.AddText(" Маммография\n").Size("24").Bold()
    //Основная часть
    para3 := w.AddParagraph().Justification("left")
    para3.AddText("Номер исследования 0004702832\n").Size("24")
    para3.AddText("Дата исследования 25.03.2025
21:30\n").Size("24")
    para3.AddText( fmt.Sprintf("специалист
%s\n",data.Examination.Doctor) ).Size("24")
    para3.AddText("ФИО Пациента ").Bold().Size("24")
    para3.AddText(data.Examination.Patient).Size("24")
    para3.AddText("\nПараметры рентгеновского
исследования\n").Bold().Size("24")

```

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		160

```

    para3.AddText( fmt.Sprintf("Количество снимков: %d, доза
облучения (мЗв) %f\n", data.Examination.ImagesCount,
data.Examination.Dose) ).Size("24")

    para3.AddText("Протокол исследования\n").Bold().Size("24")
    //
    para4 := w.AddParagraph().Justification("left")
    para4.AddText("Описание:\n").Bold().Size("24")
    description:=fmt.Sprintf("Структура: %s", data.density)
    for _, pathology := range data.patologies {
        ptype := pathology.type
        switch ptype {
            case "кальцинат":
                description+="Кальцинат, расположенный в "
        }
    }
    para4.AddText("Заключение: ").Bold().Size("24")
    para4.AddText(data.Conclusion).Size("24")
    para4.AddText("\nСпециалист: ").Bold().Size("24")
    para4.AddText(data.Examination.Doctor).Size("24")
    f, err := os.Create("generated.docx")
    // save to file
    if err != nil {
        panic(err)
    }
    _, err = w.WriteTo(f)
    if err != nil {
        panic(err)
    }
    err = f.Close()
    if err != nil {
        panic(err)
    }
}

```


Приложение Д

(обязательное)

Приложение для анализа и обработки рентгеновских снимков

Презентация

					ПГУ1.090501.13.001 ПЗ	Лист
						162
Изм.	Лист	№ докум.	Подпись	Дата		

Бизнес-проект: Приложение для анализа и обработки рентгеновских снимков

Выполнил: студент группы 20ВОЭ1 Нагаев М.Т.
Руководитель: к.т.н, доцент каф. ВТ Карамышева Н.С.

Слайд Г.1 – Тема дипломного проекта

> Цель

Создание приложения для анализа и обработки рентгеновских снимков в условиях роста медицинских данных для увеличения производительности труда и упрощения постановки диагноза



2

Слайд Г.2 – Цель дипломного проекта

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ1.090501.13.001 ПЗ

Лист

163

> Сравнительный анализ существующих решений

1

Iray IDemo - получение снимков без редактирования и заключений

2

Osirix Dicom Viewer поддерживает создание заключений, просмотр и редактирование снимков, но нет адаптации под российский рынок и получения снимков

3

Квант.МИС Лайт - российское ПО для создания врачебных заключений

4

Наше приложение совмещает получение и редактирование снимков, поддерживает формирование заключений и полностью учитывает локальные требования.

3

Слайд Г.3 – Анализ существующих решений

> Выбор технологий разработки: C++, Qt и Go



C++ обеспечивает эффективную обработку изображений и управление оборудованием, что критично для нашего приложения.

Qt используется для кроссплатформенного GUI с настраиваемыми диалоговыми окнами, а Go отвечает за модуль генерации врачебных заключений.

4

Слайд Г.4 – Выбор технологий разработки

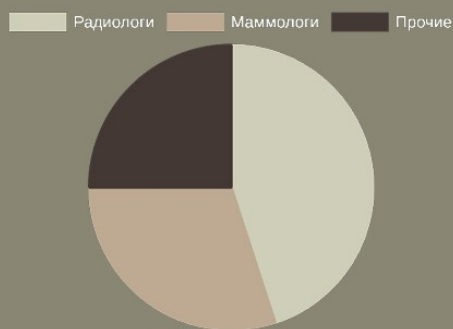
Изм.	Лист	№ докум.	Подпись	Дата

ПГУ1.090501.13.001 ПЗ

Лист

164

> Анализ целевой аудитории:
специализации и потребности



Решение обеспечивает полный цикл работы врача: от получения и обработки снимка до создания заключения

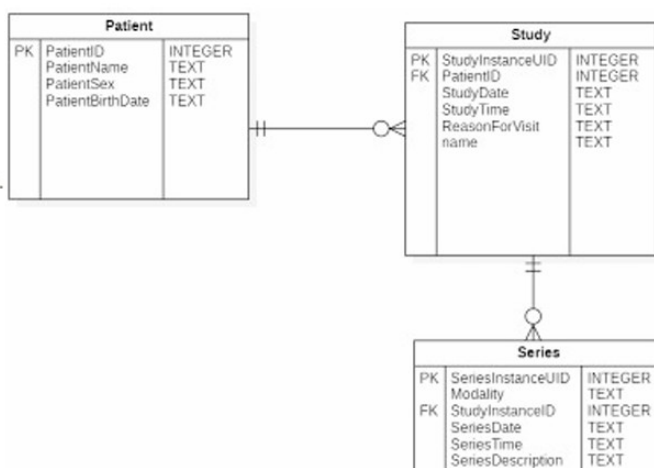
Возможность для малых клиник использовать решение с существующим оборудованием и сэкономить

5

Слайд Г.5 – Анализ целевой аудитории

> Физическая модель базы данных

В базе предусмотрены таблицы Пациент, Обследование, Серия с необходимыми атрибутами для хранения полной информации.



6

Слайд Г.6 – Физическая модель БД

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ1.090501.13.001 ПЗ

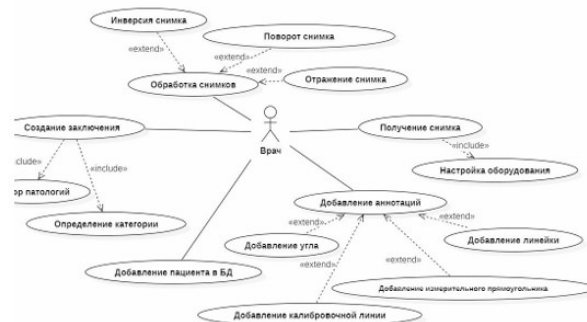
Лист

165

> **Диаграмма вариантов использования:
функциональность системы**

Основные функции

Врач может получать, обрабатывать снимки, добавлять аннотации и формировать врачебные заключения



7

Слайд Г.7 – Диаграмма вариантов использования

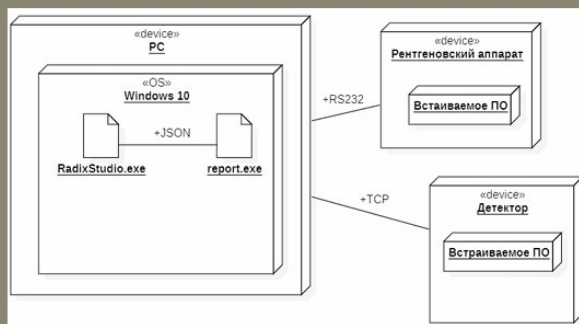
Изм.	Лист	№ докум.	Подпись	Дата

ПГУ1.090501.13.001 ПЗ

Лист

166

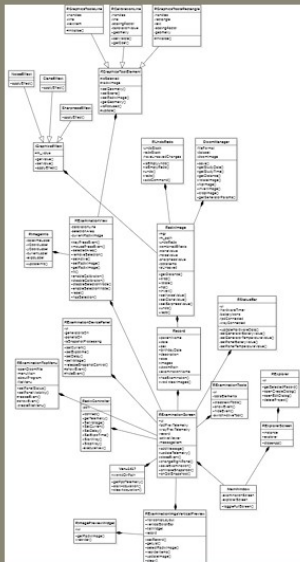
> Диаграмма развертывания: архитектура и компоненты



Система включает приложение под ПК, вспомогательное приложение для формирования заключений и рентгенологическое оборудование

Для связи между приложениями используется JSON, между оборудованием - RS232 и UDP

Слайд Г.8 – Диаграмма развертывания



Ключевые классы обеспечивают обработку изображений, пользовательский интерфейс и управление базой данных для эффективной работы приложения.

Диаграмма классов: структура и взаимодействия

Слайд Г.9 – Диаграмма классов

> Заключение

Разработано приложение для врачей-радиологов
Повышена производительность труда врача
Возможности расширения:
- добавление ИИ в анализ;
- новые медицинские аннотации;
- PACS

10

Слайд Г.10 – Заключение

					ПГУ1.090501.13.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		168