Министерство образования Республики Беларусь

Учреждение образования

ГРОДНЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИМЕНИ ЯНКИ КУПАЛЫ

Физико-технический факультет

Кафедра информационных систем и технологий

К защите допустить*:*

Заведующий кафедрой ИСиТ

\_\_\_\_\_\_\_\_\_\_\_\_\_Ю.Р.Бейтюк

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_20\_\_г.

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к дипломному проекту

на тему

**Разработка системы автоматического распознавания, учёта и анализа пиков интенсивности рассеяния люминесцентных наночастиц на образцах изображений раковых клеток**

ГРГУ ДП 1-38 02 01 08 009 ПЗ

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Я.Ю. Равлушевич

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Н.Н. Бич

Нормоконтролер \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Ю.Р. Бейтюк

Рецензент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Кропочёва Л.В.

Гродно 2022

**РЕФЕРАТ**

Дипломный проект «Разработка системы автоматического распознавания, учёта и анализа пиков интенсивности рассеяния люминесцентных наночастиц на образцах изображений раковых клеток» содержит 101 страницы, 27 изображение, 2 таблицы, 18 источников литературы.

Цель данной работы: создать ПО(веб-приложения) для распознавания, учёта, анализа, визуализации пиков и реализация статистики для сравнения образцов раковых клеток разных возрастов.

Методы исследования: для реализации проекта использовались текстовый редактор Visual Studio Code и Jupyter Notebook, языки программирования JavaScript и Python.

Полученные результаты:

* веб-приложение.
* Настроенный веб-сервер, сохраняющий информацию о пиках на образцах раковых клеток разных возрастов БД.

Область применения: проект может быть использован в учреждениях здравоохранения и медицинских вузах.

**RESUME**

The diploma project "Development of a system for automatic recognition, accounting and analysis of the intensity peaks of scattering of fluorescent nanoparticles on cancer cell image samples" contains 101 pages, 27 images, 2 tables, 18 literature sources.

The purpose of this work: to create software (web applications) for recognition, accounting, analysis, visualization of peaks and the implementation of statistics for comparing samples of cancer cells of different ages.

Research methods: Visual Studio Code and Jupyter Notebook text editor, JavaScript and Python programming languages were used to implement the project.

The results obtained are:

* web application.
* Configured web server that stores information about peaks on cancer cell samples of different ages in a database.

Scope: the project can be used in healthcare institutions and medical universities.

Оглавление

[ВВЕДЕНИЕ 6](#_Toc105357944)

[1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ 8](#_Toc105357945)

[1.1 Люминесцентный анализ 8](#_Toc105357946)

[1.2 Люминесцентные наночастицы 11](#_Toc105357947)

[2. КРАТКОЕ ОПИСАНИЕ ТЕХНОЛОГИЙ ПРОЕКТА 12](#_Toc105357948)

[2.1 Язык программирования JavaScript 12](#_Toc105357949)

[2.2 Язык программирования Python 13](#_Toc105357950)

[2.3 React 14](#_Toc105357951)

[2.4 Flask 14](#_Toc105357952)

[2.5 MySQL 15](#_Toc105357953)

[2.6 Используемое программное обеспечение Python 16](#_Toc105357954)

[3. ПРАКТИЧЕСКАЯ ЧАСТЬ 20](#_Toc105357955)

[3.1 Общая структура 20](#_Toc105357956)

[3.2 Реализация клиентской части 21](#_Toc105357957)

[3.3.1 Начало работы с React 22](#_Toc105357958)

[3.3.2 Рендеринг элементов 23](#_Toc105357959)

[3.3.2 Главный компонент App 25](#_Toc105357960)

[3.3 Реализация серверной части 39](#_Toc105357961)

[3.3.1 Создание Flask сервера 40](#_Toc105357962)

[3.3.2 Методы нахождения пиков 43](#_Toc105357963)

[3.3.3 Критерий Крускала-Уоллиса 45](#_Toc105357964)

[3.3.4 Структура БД 46](#_Toc105357965)

[3.3.5 Реализация модулей 49](#_Toc105357966)

[**4.** ТЕХНИКА БЕЗОПАСНОСТИ ПРИ РАБОТЕ ЗА ПЕРСОНАЛЬНЫМ КОМПЬЮТЕРОМ 58](#_Toc105357967)

[4.1 Требования по электрической безопасности 58](#_Toc105357968)

[4.2 Особенности электропитания системного блока 59](#_Toc105357969)

[4.3 Система гигиенических требований 60](#_Toc105357970)

[4.4 Требования к рабочему месту 61](#_Toc105357971)

[4.5 Требования к организации пространства 63](#_Toc105357972)

[5. Энергосбережение 64](#_Toc105357973)

[5.1 Основные способы экономии энергии 64](#_Toc105357974)

[5.2 Энергосберегающие режимы работы компьютера 65](#_Toc105357975)

[5.3 Энергоэффективность в серверных помещениях 66](#_Toc105357976)

[5.4 Стандарты энергосберегающих технологий 66](#_Toc105357977)

[5.4.1 Стандарт усовершенствованной системы управления 67](#_Toc105357978)

[питанием (Advanced Power Management — APM) 67](#_Toc105357979)

[5.4.2 Стандарт усовершенствованной системы управления 69](#_Toc105357980)

[питанием (Advanced Power Management—APM) 69](#_Toc105357981)

[5.5 Энергосберегающие технологии в мобильных ПК 69](#_Toc105357982)

[5.5.1 Технология Enhanced Intel SpeedStep 70](#_Toc105357983)

[5.5.2 Режимы Deep Sleep и Deeper Sleep 73](#_Toc105357984)

[5.5.3 Технология Intel Mobile Voltage Positioning (IMVP) 74](#_Toc105357985)

[6. ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ 75](#_Toc105357986)

[6.1 Оптимальные характеристики оборудования 75](#_Toc105357987)

[6.2 Расчет себестоимости программного обеспечения 76](#_Toc105357988)

[ЗАКЛЮЧЕНИЕ 84](#_Toc105357989)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 85](#_Toc105357990)

[ПРИЛОЖЕНИЕ 87](#_Toc105357991)

# ВВЕДЕНИЕ

Опухолевые клетки - патологические клетки многоклеточного организма, являющиеся частью доброкачественной или злокачественной опухоли. Рак является одной из основных причин смертности людей. По данным ВОЗ за 2015 год около 8,8 млн человек скончались, причиной чаще всего являлись метастазы раковых клеток на другие органы и несвоевременное обращение за медицинской помощью.

Эта тема является актуальной проблемой нашего время. В современном мире с каждым годом увеличивается уровень онкологических заболеваний. Это обусловлено не только экологической обстановкой, но и образом жизни современного человека. В настоящее время активно изучаются механизмы образования злокачественных опухолей, а также этапы лечения.

Смертность от онкологических заболеваний можно снизить, если выявлять и лечить их на ранних стадиях, а для этого должны быть проведены некоторые исследования. Основными методами диагностики и анализа раковых клеток являются: УЗИ, лучевая диагностика, эндоскопия, опухолевые маркеры, гистологические и цитологические методы.

Целью данного курсового проекта является разработка программного решения для автоматического распознавания, учёта и анализа пиков интенсивности рассеяния люминесцентных наночастиц на образцах изображений раковых клеток, с целью меньшего затрата сил и времени для нахождения пиков на образцах изображений раковых клеток для дальнейшего анализа и обработки.

Объект исследования: пики на графиках для каждой точки на образцах изображений раковых клеток

Предмет исследования: нахождения значений интенсивности и волнового числа для каждого пика на графике для каждой точки на образцах изображений раковых клеток

Для разработки проекта необходимо решить следующие задачи:

1. Выбор языков программирования
2. Выбор программного обеспечения
3. Описать методы для нахождения пиков и выбрать подходящий
4. Написать программный код для распознавания, учёта, анализа, визуализации пиков
5. Реализация статистики для сравнения образцов раковых клеток разных возрастов

# 1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

## **1.1 Люминесцентный анализ**

«Люминесценция - (от [лат.](https://ru.wikipedia.org/wiki/%D0%9B%D0%B0%D1%82%D0%B8%D0%BD%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA)lumen свет и -escent суффикс, означающий слабое действие) это нетепловое свечение вещества, наблюдаемое после поглощения им энергии возбуждения. Впервые явление люминесценции было описано в [восемнадцатом веке» [1].](https://ru.wikipedia.org/wiki/XVIII_%D0%B2%D0%B5%D0%BA)

«Люминофоры это специально синтезируемые материалы, которые люминесцируют при различных способах возбуждениях, это явлениеиспользуется для практических целей» [1].

«Люминесцентный анализ это методы анализа, использующие явление люминесценции. Для возникновения люминесценции чаще всего используют ультрафиолетовое излучение, например, применяют кварцевую лампу. Чувствительность данного метода может достигать 10-10 моль/л, что является высоким показателем для физико-химических методов анализа. При проведении люминесцентного анализа можно использовать как специальные приборы (люминесцентный микроскоп, спектрофотометры различного спектра излучения), так и источники излучения, не снабженные измерительными модулями (ртутные лампу, дейтериевые лампу и пр.)» [1].

«По характеру люминесцентного излучения можно судить о состоянии многих тканей организма таких как: ногти, зубы, волосы, хрусталик глаза, роговая оболочка. Это связано с различным цветом излучения при норме и патологии ткани под действием УФ- излучения» [1].

«В медицинских и биологических исследованиях явление люминесценции и люминесцентный анализ играет существенную роль» [1].

«Явление люминесценция широко применяется для диагностики кожных заболеваний в частности грибка и лишаев, так как они обладают характерным свечением» [1].

«Применение люминесцентного анализа в медицинских и фармацевтических исследованиях » [1]:

«В дерматологии – для диагностики некоторых грибковых заболеваний кожи и волос, используется УФ спектр излучения и соответствующая лампа, чаще всего ртутная» [1].

«В онкологии – для обнаружения границ роста злокачественной опухоли во время оперативных вмешательств и диагностики, так как клетки опухоли накапливают вещества дающие излучение при возбуждении в УФ области спектра» [1].

«В микробиологии – для обнаружения бактерий туберкулеза, также применяется в вирусологии. Люминесцентные материалы применяют для окраски препаратов для микроскопии, с последующим применением люминесцентных ламп» [1].

«В биохимических исследованиях – для определения содержания витаминов в биологических объектах» [1].

«В клинике глазных болезней – для обнаружения язв и прочих заболеваний роговицы» [1].

«В клинике внутренних болезней – для определения скорости кровотока.

«В судебной медицине и токсикологии – для обнаружения следов токсичных веществ, наркотических веществ (атропин люминесцирует синим светом, стрихнин – зеленым и т.д.), биологических жидкостей, анализа крови и т.д.» [1].

«В фармацевтическом анализе» [1]:

«Для анализа лекарственных веществ и их изменений при порче и длительности хранения (Например, при гидролизе аспирина образуется салициловая кислота, проявляющая яркую флуоресценцией синего цвета, в то время как сам аспирин не проявляет люминесцентных свойств. Поэтому по флуоресценции твердых кристаллических препаратов аспирина можно судить о степени их разложения)» [1];

«Для установления подлинности лекарственных веществ» [1];

«Для обнаружения и исследования алкалоидов: морфина, кокаина и др., обладающих люминесцентными свойствами» [1].

«Возможно также сочетание люминесцентного анализа с другими видами физико-химических методов, например с хроматографией или электрохимическими методами анализа. Применение люминесцентных индикаторов возможно использовать в титриметрических методах анализа» [1].

«Из всего вышесказанного можно сделать вывод,  что явление люминесценции является перспективным для применения в медицинских и фармацевтических исследованиях» [1].

## **1.2 Люминесцентные наночастицы**

«В фотобиологии и биомедицине наночастицы на основе NaYF4 с добавкой других редкоземельных ионов широко используются благодаря низкой токсичности материала и его способности проникать в ткани за счет маленького размера (единицы и десятки нанометров). Так, гадолиний с его магнитными свойствами уже сейчас применяется для создания контрастных веществ при проведении магнитно-резонансной томографии. А люминесцентные наночастицы, содержащие ионы европия, используются как маркеры различных заболеваний для исследования биологических тканей с помощью люминесцентной микроскопии. Новейшие люминофоры, дают возможность проводить сразу два вида диагностики — с помощью света и магнитного поля, что значительно увеличивает эффективность поиска заболеваний» [1].

«Можно взять образцы клеток или тканей и исследовать их с помощью люминесцентного микроскопа: европий подсветит те места, куда пришли "настроенные" на поиск раковой опухоли частицы. А использование МРТ позволит в живом организме выявить опухоль, поскольку в ней будут скапливаться наночастицы, содержащие магнитные ионы гадолиния» [1].

«В перспективе часть ионов в составе соединения можно заменить на радиоактивные и применять в лечении онкологических заболеваний. Кроме медицины данные материалы можно использовать при создании люминесцентных красок для защиты документов, денег и ценных бумаг, где требуется особенно яркое свечение» [1].

# 2. КРАТКОЕ ОПИСАНИЕ ТЕХНОЛОГИЙ ПРОЕКТА

В проекте использовался комплекс программного обеспечения для обработки изображений образцов раковых клеток разных возрастов с целью нахождения и визуализации пиков, нахождения уникальных пиков(которых нету в других точках для конкретного образца раковых клеток), сохранение информации о пиках пяти точек для каждого образца раковых клеток разных возрастов в базу данных, реализации статистики для сравнения образцов раковых клеток разных возрастов. Технологии связанные с созданием веб-приложения: JavaScript, React, Python, Flask, MySQL.

## **2.1 Язык программирования JavaScript**

«JavaScript (аббр. JS) — мультипарадигменный язык программирования. Поддерживает объектно-ориентированный, императивный и функциональный стили. Является реализацией языка ECMAScript (стандарт ECMA-262)» [2].

«JavaScript обычно используется как встраиваемый язык для программного доступа к объектам приложений. Наиболее широкое применение находит в браузерах как язык сценариев для придания интерактивности веб-страницам» [2].

«Основные архитектурные черты: динамическая типизация, слабая типизация, автоматическое управление памятью, прототипное программирование, функции как объекты первого класса» [2].

«JavaScript является объектно-ориентированным языком, но используемое в языке прототипирование обуславливает отличия в работе с объектами по сравнению с традиционными класс-ориентированными языками. Кроме того, JavaScript имеет ряд свойств, присущих функциональным языкам — функции как объекты первого класса, объекты как списки, карринг, анонимные функции, замыкания — что придаёт языку дополнительную гибкость [2].

## **2.2 Язык программирования Python**

«Python (в русском языке встречается как питон или пайтон) — [высокоуровневый язык программирования](https://ru.wikipedia.org/wiki/%D0%92%D1%8B%D1%81%D0%BE%D0%BA%D0%BE%D1%83%D1%80%D0%BE%D0%B2%D0%BD%D0%B5%D0%B2%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F) общего назначения с [динамической](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D0%BD%D0%B0%D0%BC%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%82%D0%B8%D0%BF%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F) [строгой](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D0%BE%D0%B3%D0%B0%D1%8F_%D1%82%D0%B8%D0%BF%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F) типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости [кода](https://ru.wikipedia.org/wiki/%D0%98%D1%81%D1%85%D0%BE%D0%B4%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BE%D0%B4) и его качества, а также на обеспечение переносимости написанных на нём программ. Язык является полностью [объектно-ориентированным](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F) в том плане, что всё является [объектами](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)). [Синтаксис](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BD%D1%82%D0%B0%D0%BA%D1%81%D0%B8%D1%81_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) ядра языка минималистичен. Сам же язык известен как [интерпретируемый](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%BF%D1%80%D0%B5%D1%82%D0%B8%D1%80%D1%83%D0%B5%D0%BC%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F) и используется в том числе для написания [скриптов](https://ru.wikipedia.org/wiki/%D0%A1%D0%BA%D1%80%D0%B8%D0%BF%D1%82)» [3].

«Python является мультипарадигменным языком программирования, в котором используются такие подходы как структурный, объектно-ориентированный, функциональный, императивный и аспектно-ориентированный» [3].

«Эталонной реализацией Python является интерпретатор CPython, поддерживающий большинство активно используемых платформ. Он распространяется под свободной лицензией Python Software Foundation License, позволяющей использовать его без ограничений в любых приложениях, включая проприетарные» [3].

«Python — активно развивающийся язык программирования, новые версии (с добавлением/изменением языковых свойств) выходят примерно раз в два с половиной года. Вследствие этого и некоторых других причин на Python отсутствуют стандарт ANSI, ISO или другие официальные стандарты, их роль выполняет CPython» [3].

«[Разработка языка Python была начата в конце 1980-х годов сотрудником голландского института CWI Гвидо ван Россумом. Для распределенной ОС Amoeba требовался расширяемый скриптовый язык, и Гвидо начал писать Python на досуге, позаимствовав некоторые наработки для языка ABC (Гвидо участвовал в разработке этого языка, ориентированного на обучение программированию). В феврале 1991 года Гвидо опубликовал исходный текст в группе новостей alt.sources. С самого начала Python проектировался как объектно-ориентированный язык](https://ru.wikipedia.org/wiki/Python#cite_note-18)» [3].

«[Наличие дружелюбного, отзывчивого сообщества пользователей считается наряду с дизайнерской интуицией Гвидо одним из факторов успеха Python. Развитие языка происходит согласно четко регламентированному процессу создания, обсуждения, отбора и реализации документов PEP (англ. Python Enhancement Proposal) — предложений по развитию Python](https://ru.wikipedia.org/wiki/Python#cite_note-18)» [3].

## **2.3 React**

React — это JavaScript библиотека с открытым исходным кодом для построения пользовательских интерфейсов.

React может использоваться при разработке одностраничных веб-приложений и мобильных приложений. React нацелен в первую очередь на обеспечение высокой скорости разработки, простоты и расширяемости.

React был разработан Джорданом Уолком (Jordan Walke), программистом компании Facebook. Впервые React был использован в 2011 году в новостной ленте Facebook и впоследствии на сайте Instagram в 2012. Его исходный код был открыт в мае 2013 года [4].

## **2.4 Flask**

«[Модуль Flask](https://docs-python.ru/packages/veb-frejmvork-flask-python/) представляет собой микро-фреймворк, но слово "микро" не означает, что веб-приложение, построенное на Flask, должно помещаться в один файл с кодом на Python, хотя, это не запрещено. Или Flask испытывает недостаток в функциональности. Нет, приставка "микро" означает, что Flask придерживаться простого, но расширяемого ядра. Это означает, что Flask не решает за программиста что и как ему делать. При проектировании веб-приложений на Flask программист все решения принимает сам: например, какие паттерны использовать, какую базу данных подключить, использовать ли ORM или работать на чистом SQL. Flask не содержит больших слоев абстракции и поэтому быстр. В общем Flask - это всё, что нужно, и ничего лишнего» [5].

«Фреймворк Flask поддерживает много расширений для добавления различной функциональности, которые обеспечивают интеграцию с базами данных, валидацию форм, различные технологии аутентификации и так далее. Flask имеет множество параметров конфигурации с разумными значениями по умолчанию, и мало предварительных соглашений. По соглашениям Flask, шаблоны и статические файлы хранятся в поддиректориях внутри дерева исходных кодов Python, с названиями templates и static соответственно, а в качестве движка шаблонов рекомендовано использовать [модуль Jinja2](https://docs-python.ru/packages/modul-jinja2-python/). Эти соглашения изменяемы, но изменять их не стоит, особенно тем, кто только начинает знакомится с Flask» [5].

## **2.5 MySQL**

«MySQL — [свободная](https://ru.wikipedia.org/wiki/%D0%A1%D0%B2%D0%BE%D0%B1%D0%BE%D0%B4%D0%BD%D0%BE%D0%B5_%D0%9F%D0%9E) [реляционная система управления базами данных](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BB%D1%8F%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D0%A1%D0%A3%D0%91%D0%94). Разработку и поддержку MySQL осуществляет корпорация [Oracle](https://ru.wikipedia.org/wiki/Oracle), получившая права на торговую марку вместе с поглощённой [Sun Microsystems](https://ru.wikipedia.org/wiki/Sun_Microsystems), которая ранее приобрела шведскую компанию [MySQL AB](https://ru.wikipedia.org/wiki/MySQL_AB). Продукт распространяется как под [GNU General Public License](https://ru.wikipedia.org/wiki/GNU_General_Public_License), так и под собственной коммерческой лицензией. Помимо этого, разработчики создают функциональность по заказу лицензионных пользователей. Именно благодаря такому заказу почти в самых ранних версиях появился механизм [репликации](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BF%D0%BB%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F_(%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D1%82%D0%B5%D1%85%D0%BD%D0%B8%D0%BA%D0%B0))» [6].

«MySQL является решением для малых и средних приложений. Входит в состав серверов [WAMP](https://ru.wikipedia.org/wiki/WAMP), [AppServ](https://ru.wikipedia.org/w/index.php?title=AppServ&action=edit&redlink=1), [LAMP](https://ru.wikipedia.org/wiki/LAMP) и в портативные сборки серверов [Денвер](https://ru.wikipedia.org/wiki/%D0%94%D0%B5%D0%BD%D0%B2%D0%B5%D1%80_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0)), [XAMPP](https://ru.wikipedia.org/wiki/XAMPP), [VertrigoServ](https://ru.wikipedia.org/w/index.php?title=VertrigoServ&action=edit&redlink=1). Обычно MySQL используется в качестве сервера, к которому обращаются локальные или удалённые клиенты, однако в дистрибутив входит библиотека внутреннего сервера, позволяющая включать MySQL в автономные программы» [6].

«Гибкость СУБД MySQL обеспечивается поддержкой большого количества типов таблиц: пользователи могут выбрать как таблицы типа [MyISAM](https://ru.wikipedia.org/wiki/MyISAM), поддерживающие полнотекстовый поиск, так и таблицы [InnoDB](https://ru.wikipedia.org/wiki/InnoDB), поддерживающие транзакции на уровне отдельных записей. Более того, СУБД MySQL поставляется со специальным типом таблиц EXAMPLE, демонстрирующим принципы создания новых типов таблиц. Благодаря открытой архитектуре и GPL-лицензированию, в СУБД MySQL постоянно появляются новые типы таблиц» [6].

## **2.6 Используемое программное обеспечение Python**

Решение разрабатывается на языке Python и дополнительных библиотек для работы с данными и изображениями с использованием стандартного интерпретатора Python и среды Jupyter-Notebook.

Используемые Python библиотеки:

Pillow — библиотека языка [Python](https://ru.wikipedia.org/wiki/Python), предназначенная для работы с [растровой графикой](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D1%81%D1%82%D1%80%D0%BE%D0%B2%D0%B0%D1%8F_%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D0%BA%D0%B0), которая добавляет вашему коду поддержку открытия, изменения и сохранения изображений в различных расширениях [7].

«OpenCV (англ. Open Source Computer Vision Library - библиотека компьютерного зрения с открытым исходным кодом) — библиотека алгоритмов компьютерного зрения, обработки изображений и численных алгоритмов общего назначения с открытым кодом. Реализована на C/C++, также разрабатывается для Python, Java, Ruby, Matlab, Lua и других языков. Может свободно использоваться в академических и коммерческих целях — распространяется в условиях лицензии BSD» [8].

«Второе крупное обновление OpenCV было выпущено в октябре 2009 года. OpenCV2 включает в себя серьезные изменения в интерфейсе C++, направленные на упрощение, улучшение безопасности, введение новых функций и увеличение производительности (особенно для многоядерных систем). Официальные релизы теперь выпускаются каждые шесть месяцев, а разработка ведется независимой российской командой при поддержке коммерческих корпораций» [8].

«Tesseract — [свободная компьютерная программа](https://ru.wikipedia.org/wiki/%D0%A1%D0%B2%D0%BE%D0%B1%D0%BE%D0%B4%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5) для [распознавания текстов](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%BE%D0%B5_%D1%80%D0%B0%D1%81%D0%BF%D0%BE%D0%B7%D0%BD%D0%B0%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5_%D1%81%D0%B8%D0%BC%D0%B2%D0%BE%D0%BB%D0%BE%D0%B2), разрабатывавшаяся [Hewlett-Packard](https://ru.wikipedia.org/wiki/Hewlett-Packard) с середины 1980-х по середину 1990-х, а затем 10 лет «пролежавшая на полке». В августе 2006 г. [Google](https://ru.wikipedia.org/wiki/Google_(%D0%BA%D0%BE%D0%BC%D0%BF%D0%B0%D0%BD%D0%B8%D1%8F)) купил её и открыл исходные тексты под [лицензией Apache 2.0](https://ru.wikipedia.org/wiki/Apache_Software_License) для продолжения разработки. В настоящий момент программа уже работает с UTF-8, поддержка языков (включая русский с версии 3.0) осуществляется с помощью дополнительных модулей» [9].

«Программа Tesseract используется менеджером загрузок [Tucan Manager](https://ru.wikipedia.org/wiki/Tucan_Manager) для распознавания текстов в тестах [CAPTCHA](https://ru.wikipedia.org/wiki/CAPTCHA)» [9].

«NumPy (сокращенно от Numerical Python)— библиотека с открытым исходным кодом для языка программирования Python. Возможности» [10]:

* «поддержка многомерных массивов (включая матрицы)» [10];
* «поддержка высокоуровневых математических функций, предназначенных для работы с многомерными массивами» [10].

«Математические алгоритмы, реализованные на интерпретируемых языках (например, Python), часто работают гораздо медленнее тех же алгоритмов, реализованных на компилируемых языках (например, Фортран, Си, Java). Библиотека NumPy предоставляет реализации вычислительных алгоритмов (в виде функций и операторов), оптимизированные для работы с многомерными массивами. В результате любой алгоритм, который может быть выражен в виде последовательности операций над массивами (матрицами) и реализованный с использованием NumPy, работает так же быстро, как эквивалентный код, выполняемый в MATLAB» [10].

«Matplotlib — библиотека на языке программирования Python для визуализации данных двумерной (2D) графикой (3D графика также поддерживается). Получаемые изображения могут быть использованы в качестве иллюстраций в публикациях» [11].

«Matplotlib написан и поддерживался в основном Джоном Хантером (англ. John Hunter) и распространяется на условиях BSD-подобной лицензии. Генерируемые в различных форматах изображения могут быть использованы в интерактивной графике, в научных публикациях, графическом интерфейсе пользователя, веб-приложениях, где требуется построение диаграмм (англ. plotting). В документации автор признаётся, что Matplotlib начинался с подражания графическим командам MATLAB, но является независимым от него проектом» [11].

«Версия 2.1.1 — последняя стабильная — требует Python версии 2.7 или от 3.4 и выше и версию NumPy от 1.7.1 и выше» [11].

«Библиотека Matplotlib построена на принципах ООП, но имеет процедурный интерфейс pylab, который предоставляет аналоги команд MATLAB» [11].

«Matplotlib является гибким, легко конфигурируемым пакетом, который вместе с NumPy, SciPy и IPython предоставляет возможности, подобные MATLAB. В настоящее время пакет работает с несколькими графическими библиотеками, включая wxWindows и PyGTK» [11].

«Пакет поддерживает многие виды графиков и диаграмм» [11]:

* «Графики (line plot)» [11]
* «Диаграммы разброса (scatter plot)» [11]
* «Столбчатые диаграммы (bar chart) и гистограммы (histogram)» [11]
* «Круговые диаграммы (pie chart)» [11]
* «Ствол-лист диаграммы (stem plot)» [11]
* «Контурные графики (contour plot)» [11]
* «Поля градиентов (quiver)» [11]
* «Спектральные диаграммы (spectrogram)» [11]

«Пользователь может указать оси координат, решетку, добавить надписи и пояснения, использовать логарифмическую шкалу или полярные координаты» [11].

«Несложные трёхмерные графики можно строить с помощью набора инструментов (toolkit) mplot3d. Есть и другие наборы инструментов: для картографии, для работы с Excel, утилиты для GTK и другие» [11].

«С помощью Matplotlib можно делать и анимированные изображения» [11].

«SciPy — библиотека для языка программирования Python с открытым исходным кодом, предназначенная для выполнения научных и инженерных расчётов. Пакет signal содержит набор инструментов обработки сигналов в настоящее время содержит некоторые функции фильтрации, ограниченный набор инструментов проектирования фильтров и несколько алгоритмов интерполяции B-сплайнов для 1- и 2-D данных. Пакет stats предоставляет некоторое количество статистических методов, которые рассматриваются в курсах статистики» [12].

# 3. ПРАКТИЧЕСКАЯ ЧАСТЬ

## **3.1 Общая структура**

Проект назван Cancer cell statistics и разделен на две функционально независимые части – клиент и сервер.

На клиенте в свою очередь пользователь сможет загружать образцы изображений раковых клеток разных возрастов и отправлять их в качестве запроса на сервер.

На сервере эти изображения будут вытаскиваться из запроса и обрабатываться с целью распознавания, учёта, анализа, визуализации пиков и реализации статистики для сравнения образцов раковых клеток разных возрастов. В конечном итоге сервер будет записывать информацию о пиках на образцах раковых клеток разных возрастов в БД и возращать эту информацию с графиками визуализации пиков клиенту в качестве ответа.

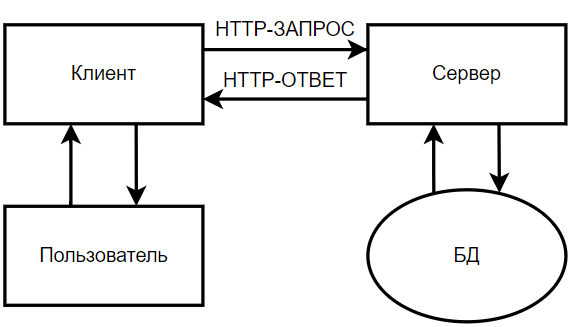


Рисунок 3.1 – Схема взаимодействия между частями приложения

REST API (Representational State Transfer) — это API, которое использует HTTP-запросы для обмена данными.

REST API должны соответствовать определенным критериям:

Архитектура клиент-сервер: пользователь взаимодействует с пользовательским интерфейсом, а сервер — с бэкендом и БД. Клиент и сервер независимы, любой из них может быть заменен отдельно от другого.

Stateless — никакие клиентские данные не сохраняются на сервере. Состояние сеанса хранится на стороне клиента.

Кэшируемость — клиенты могут кэшировать ответы сервера для улучшения общей производительности.

Что такое CRUD?

CRUD — концепция программирования, которая описывает четыре базовых действия (create, read, update и delete).

В REST API типы запросов и методы запроса отвечают за такие действия, как post, get, put, delete.

Теперь, когда мы разобрались с базовыми терминами, можно приступить к созданию API.

## **3.2 Реализация клиентской части**

Задачи клиентской части:

1) Удобный и понятный интерфейс;

2) Отправка HTPP запросов на сервер и ожидание ответа;

3) Рендеринг полученных данных от сервера

На рисунке 3.2 представлена блок-схема работы клиентской части системы.

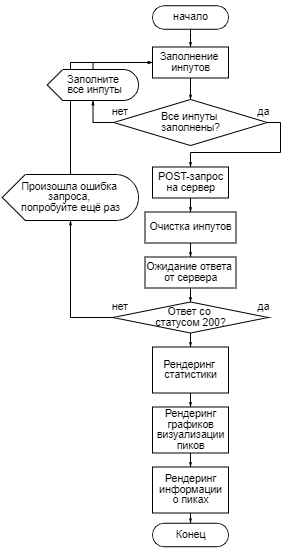


Рисунок 3.2 – Блок-схема работы клиентской части

### **3.3.1 Начало работы с React**

Приложение [Create React App](https://github.com/facebook/create-react-app) - лучший способ начать создание нового [одностраничного приложения](https://reactjs.org/docs/glossary.html#single-page-application) на React [13].

Оно настраивает вашу среду разработки, так что вы можете использовать новейшие функции JavaScript, обеспечивает удобство разработки и оптимизирует ваше приложение для продакшен [13].

Чтобы создать проект, запустим в корневой папке проекта команду через терминал [13]:

npx create-react-app client [13]

На рисунке 3.3 мы получим следующую начальную сруктуру файлов и папок клиентской части.

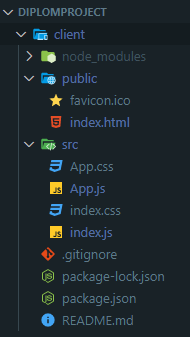


Рисунок 3.3 – Начальная структура файлов и папок клиентской части

### **3.3.2 Рендеринг элементов**

Элементы — мельчайшие кирпичики React-приложений [14].

Элемент описывает то, что вы хотите увидеть на экране [14]:

const element = <h1>Hello, world</h1> [14];

В отличие от DOM-элементов, элементы React — это простые объекты, не отнимающие много ресурсов. React DOM обновляет DOM, чтобы он соответствовал переданным React-элементам [14].

Перейдём по пути проекта /client/public/ и откроем файл index.html.

В нашем HTML-файле есть <div> [14]:

<div id="root"></div>

Мы назовём его «корневым» узлом DOM, так как React DOM будет управлять его содержимым [14].

Обычно в приложениях, написанных полностью на React, есть только один корневой элемент. При встраивании React в существующее приложение вы можете рендерить во столько независимых корневых элементов, во сколько посчитаете нужным [14].

Перейдём по пути проекта /client/scr/ и откроем файл index.js.

Он будет являться отправной точкой нашего приложения.

Для рендеринга элементов в React вначале необходимо определить корневой элемент, в который будет производиться рендеринг. Для этого применяется метод ReactDOM.createRoot(). В него передается элемент веб-страницы, в котором будет производиться рендеринг [15].

const root = ReactDOM.createRoot(document.getElementById('root'));

Здесь представляет некоторый элемент html-страницы с id="root". И в данном случае именно в этот элемент будет производиться рендеринг приложения React. Результатом метода ReactDOM.createRoot является объект ReactDOMRoot (в данном случае он представлен константой root), который представляет указатель на структуру данных, которую инфраструктура React использует для отслеживания дерева элементов для рендеринга [15].

Непосредственно для рендеринга у объекта ReactDOMRoot вызывается метод render(). В этот метод передается компонент, который мы хотим отобразить на веб-странице. Например, рендеринг компонента App [15].

root.render(<App />);

Компонент App имеет в свою очередь следующий код:

import './App.css';

function App() {

    return (

        <div className="App">

            Hello React

        </div>

    );

}

export default App;

### **3.3.2 Главный компонент App**

Компоненты позволяют разбить интерфейс на независимые части, про которые легко думать в отдельности. Их можно складывать вместе и использовать несколько раз [16].

Во многом компоненты ведут себя как обычные функции JavaScript. Они принимают произвольные входные данные (так называемые «пропсы») и возвращают React-элементы, описывающие, что мы хотим увидеть на экране [16].

Объект state описывает внутреннее состояние компонента, он похож на props за тем исключением, что состояние определяется внутри компонента и доступно только из компонента [17].

Если props представляет входные данные, которые передаются в компонент извне, то состояние хранит такие объекты, которые создаются в компоненте и полностью зависят от компонента [17].

Также в отличие от props значения в state можно изменять [17]. Именно поэтому мы и будем использовать стейты для записи данных из элементов управления и сохранения информации для последующего рендеринга.

Для получения данных от пользователя будем использовать элементы управления input. Для записи текста input с атрибутом type и значением "text", а для файлов со значением "file".

Нам нужны стейты firstAgeCells, firstFiles для записи данных для 1-го объекта(раковых клеток) и states secondAgeCells, secondFiles для 2-го объекта(раковых клеток). Эти states будут содержать информацию о возрасте раковых клеток и образцах изображений раковых клеток для конкретного объекта (Рисунок - 3.4, Рисунок - 3.5).

Имеются образцы тканей раковых клеток продолжительностью < 5 лет.

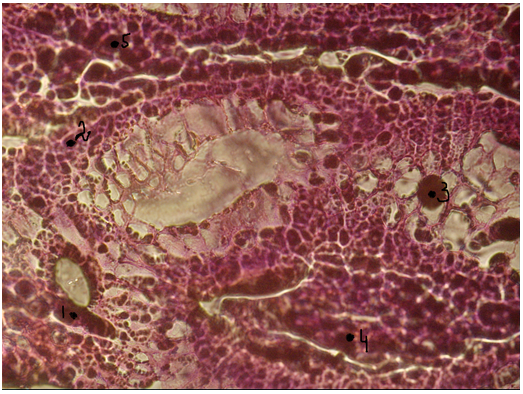


Рисунок - 3.4 Образец раковых клеток

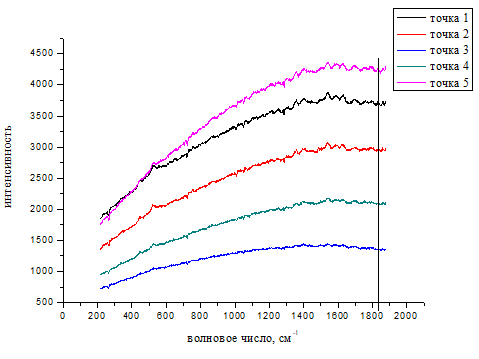


Рисунок - 3.5 Графики зависимости образца, загружаемые в input

Стейты для записи данных с input-элементов:

Для 1-го объекта(раковых клеток):

const [firstAgeCells, setFirstAgeCells] = useState('');

const [firstFiles, setFirstFiles] = useState([]);

Для 2-го объекта(раковых клеток):

const [secondAgeCells, setSecondAgeCells] = useState('');

const [secondFiles, setSecondFiles] = useState([]);

Добавим два input-элемента (Рисунок - 3.6), с атрибутом type и значением "text":

<input

type="text"

value={firstAgeCells}

onInput={(e) => setFirstAgeCells(e.target.value)}

placeholder="Введите возраст раковых клеток"

className="input\_\_text"

/>

<input

      type="text"

      value={secondAgeCells}

      onInput={(e) => setSecondAgeCells(e.target.value)}

      placeholder="Введите возраст раковых клеток"

className="input\_\_text" />

Добавим два input-элемента (Рисунок - 3.6), с атрибутом type и значением "file":

      <input

      type="file"

      multiple="multiple"

      onChange={e => setFirstFiles(e.target.files)}

      id="firstFiles"

      className="input-file\_\_real"

    />

      <input

      type="file"

      multiple="multiple"

      onChange={e => setSecondFiles(e.target.files)}

      id="secondFiles"

      className="input-file\_\_real"

      />

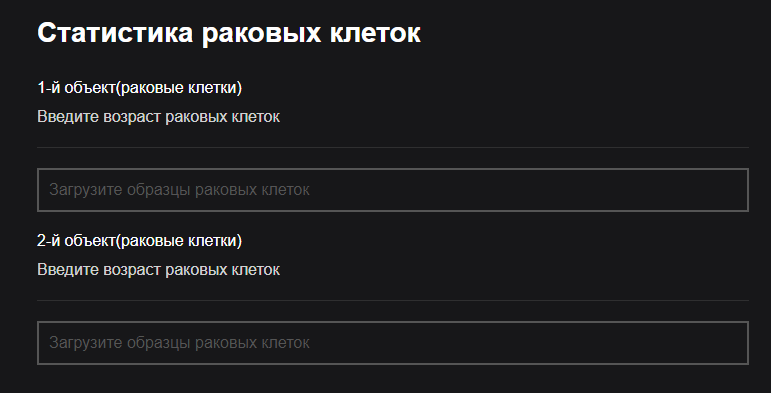


Рисунок 3.6 – Элементы управления input

Создаём события onInput, onChange и к ним функции-обработчика событий в виде анонимных функций.

Обновление стейтов происходит тогда, когда срабатывают события onInput(при введении в input-элемент возраста раковых клеток для конкретного объекта) и onChange(при загрузке файлов образцов изображений раковых клеток для конкретного объекта), вызывается анонимная функция в качестве аргумента прокидывается объект event с данными, в функцию, которая обновляет конкретный стейт.

Когда мы заполнили input-элементы данные для 1-го и 2-го объекта(раковых клеток) можно отправлять данные на сервер.

Для отправки данных на сервер добавим button-элемент (Рисунок - 3.7), создаём событие onClick и функцию-обработчика события sendFiles:

<button

      onClick={sendFiles}

  className="btn btn--yellow"

disabled={isLoading}

>

Сделать статистику

</button>

Код функции-обработчика события sendFiles:

const sendFiles = (e) => {

if(firstAgeCells && secondAgeCells.length && firstFiles.length && secondFiles.length) {

e.preventDefault();

let formData = new FormData();

            formData.append('firstAgeCells', firstAgeCells);

            formData.append('secondAgeCells', secondAgeCells);

            Array.from(firstFiles).forEach(file => {

            formData.append('firstFiles', file);

        });

            Array.from(secondFiles).forEach(file => {

            formData.append('secondFiles', file);

          });

            setIsLoading(true);

            setValidateText('');

            axios.post('/files', formData, {headers: {"Content-Type":"multipart/form-data"}})

            .then(({data}) => {

                setImages(data['arrayImages']);

                    setStatistics(data.statistics);

                    setCharacteristics(data.characteristics);

                    setIsLoading(false);

                    setPrevFirstAgeCells(firstAgeCells);

                    setPrevSecondAgeCells(secondAgeCells);

                    setFirstAgeCells('');

                    setSecondAgeCells('');

                    setFirstFiles([]);

                    setSecondFiles([]);

                })

    } else {

    setValidateText('Заполните все инпуты');

    }

};

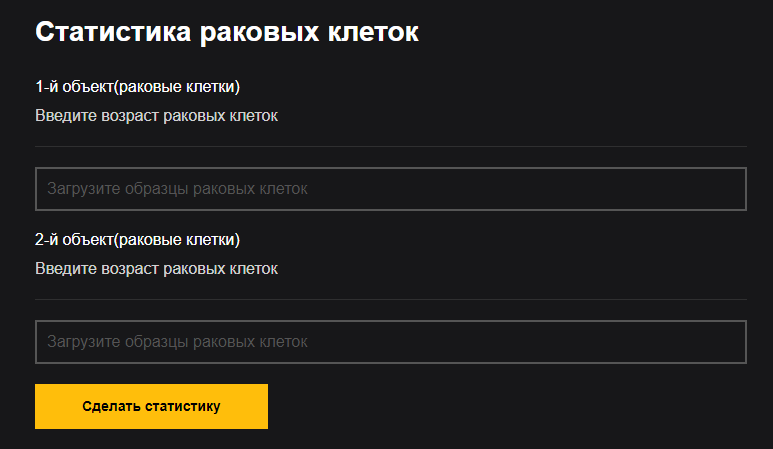


Рисунок 3.7 – Элемент управления button

Добавим новый стейт для вывода ошибки, если пользователь случайно захочет отправить пустые input-элементы:

const [validateText, setValidateText] = useState('');

Функция sendFiles проверяет input-элементы, если они не заполнены, то в state validateText записывается значение "Заполните все инпуты" (Рисунок - 3.8), иначе записывает данные в объект formData и отправляем post-запрос по адресу http://localhost:3000/files.

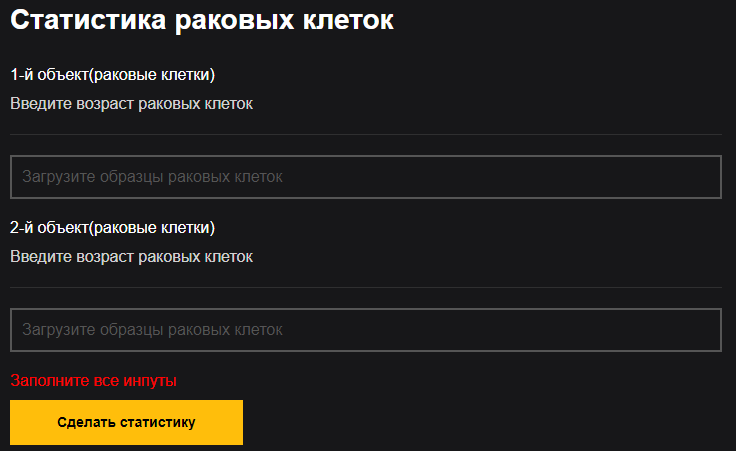


Рисунок 3.8 – Рендеринг state validateText

Чтобы посмотреть всю информацию об отправленном запросе, нажимаем в браузере F12 и открываем devTools, и видим: в Payload есть данные с input-элементов (Рисунок - 3.9), Headers status code = 200 – показал что запрос выполнился успешно (Рисунок - 3.10), и клиент в качестве Response получил данные (Рисунок - 3.11).

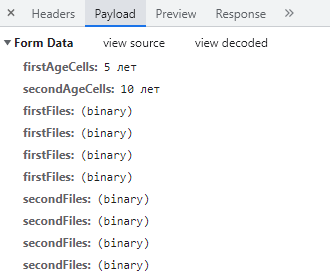


Рисунок 3.9 – Информация о post-запросе, Payload

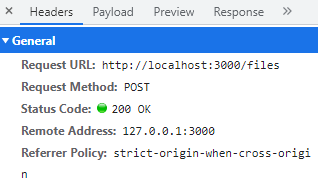


Рисунок 3.10 – Информация о post-запросе, Headers

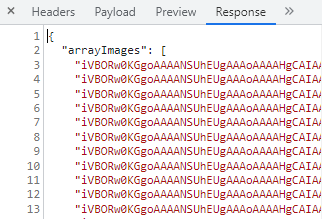


Рисунок 3.11 – Информация о post-запросе, Response

Создадим стейты для хранения данных от сервера для последующего рендеринга:

  const [images, setImages] = useState([]);

    const [statistics, setStatistics] = useState('');

    const [characteristics, setCharacteristics] = useState([]);

Стейт images – графики визуализации пиков.

Стейт statistics – статистика.

Стейт characteristics – информация о пиках.

Так как axios.post() возращает promise(обещание) мы можем этот promise обработать с помощью конструкции .then() и получить обработанные данные от сервера(data) и записать эти данные в стейты images, statistics, characteristics. После записи в states, очищаем значения стейтов input-элементов.

axios.post('/files', formData, {headers: {"Content-Type":"multipart/form-data"}})

.then(({data}) => {

      setImages(data['arrayImages']);

      setStatistics(data.statistics);

      setCharacteristics(data.characteristics);

      setIsLoading(false);

setPrevFirstAgeCells(firstAgeCells);

  setPrevSecondAgeCells(secondAgeCells);

    setFirstAgeCells('');

    setSecondAgeCells('');

      setFirstFiles([]);

    setSecondFiles([]);

          })

Для того что бы пользователь понял что данные, отправленные им обрабатываются на сервере и что нужно дождаться ответа от сервера, добавим компонент Loader (Рисунок - 3.12). и новый стейт isLoading, который будет отвечать за условный рендеринг компонентов.

const [isLoading, setIsLoading] = useState(false);

Код компонента Loader.js:

import React from 'react';

function Loader() {

return (

<div class="loader">

<div></div>

     <div></div>

     <div></div>

  </div>

);

}

export default Loader;

Когда пользователь нажмёт на кнопку отправки и отправит данные на сервер, стейт isLoading примет значение true, а когда от сервера придёт ответ, то будет иметь значение false.

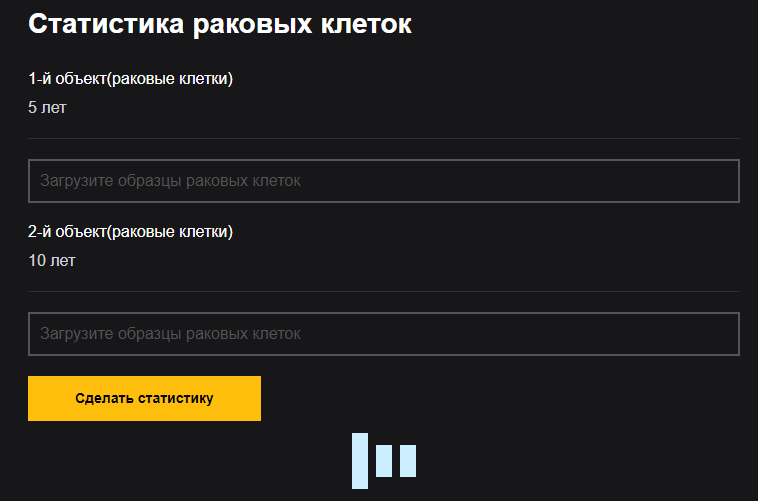


Рисунок 3.12 – Рендеринг Loader

На основании полученных от сервера данных в компоненте App можно произвести рендеринг статистики, графиков визуализации пиков.

Но для лучшего рендеринга визуализации пиков разбиваем стейт images (одномерный массив) на подмассивы, на выходе получаем два массива firstImages и secondImages для 1-го и 2-го объекта(раковых клеток).

function sliceIntoChunks(arr, chunkSize) {

      const res = [];

      for (let i = 0; i < arr.length; i += chunkSize) {

      const chunk = arr.slice(i, i + chunkSize);

    res.push(chunk);

      }

      return res;

    };

    function cutArray(arr) {

      const firstImages = arr.slice(0, arr.length / 2);

      const secondImages = arr.slice(arr.length / 2, arr.length);

        return { firstImages, secondImages };

    };

    const newImages = sliceIntoChunks(images, 5);

    const { firstImages, secondImages } = cutArray(newImages);

Если стейт isLoading примет значение true, то отрендерится компонент Loader (Рисунок - 3.12), если стейт images пустой массив, то ничего не отрендерится, в противном случае отрендерится статистика (Рисунок - 3.13), графики визуализации пиков (Рисунок - 3.14).

{isLoading ?

    <Loader /> :

      !images.length ?

      '' :

      <>

      <div className="statistics">

<div>p: {statistics.p}</div>

      <div>{statistics.textStatistics}</div>

    </div>

        <div className="images-list">

          {firstImages.map((images, index) =>

          <div>

          {images.map((img, i) =>

                    <>

<div>1-й объект(раковые клетки с возрастом {prevFirstAgeCells})</div>

                          <div>{index + 1}-й образец</div>

                          <div>{i + 1}-я точка</div>

<img src={`data:image/jpeg;base64,

${img}`} alt="img\_data" />

                       </>

              )}

              </div>

            )}

             {secondImages.map((images, index) =>

              <div>

                {images.map((img, i) =>

                    <>

                  <div>2-й объект(раковые клетки с возрастом

{prevSecondAgeCells})</div>

                      <div>{index + 1}-й образец</div>

                        <div>{i + 1}-я точка</div>

                        <img src={`data:image/jpeg;base64,${img}`} alt="img\_data"

/>

                    </>

              )}

                </div>

            )}

         </div>

    </>}

Рисунок 3.13 – Рендеринг статистики

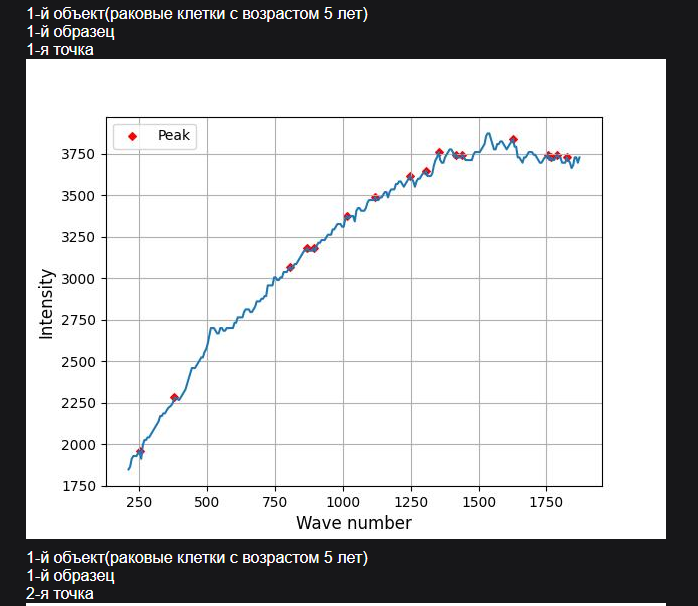


Рисунок 3.14 – Рендеринг графиков визуализации пиков

Отрендерим информацию о пиках в новом компоненте Modal, создадим новый state, который будет отвечать за то открыто/закрыто ли модальное окно, изначально оно будет закрыто.

const [modalIsOpen, setModalIsOpen] = useState(false);

Код компонента Modal.js:

import React from 'react';

import Loader from './Loader';

function Modal({ characteristics, setModalIsOpen, isLoading }) {

    return (

        <div className="modal" onClick={() => setModalIsOpen(false)}>

            <div className="modal\_\_dialog">

                <div className="modal\_\_content" onClick={(e) => e.stopPropagation()}>

                    <span className="modal\_\_close" onClick={() => setModalIsOpen(false)}>X</span>

                    {isLoading ?

                        <Loader /> :

                    !characteristics.length ?

                        'База данных пустая' :

                        <>

                            <table>

                                <thead>

                                    <tr>

                                        <th></th>

                                        <th>Волн. число</th>

                                        <th>Интенсивность</th>

                                        <th>Уник. значение</th>

                                    </tr>

                                </thead>

                                <tbody>

                                    {characteristics.length ? characteristics.map(characteristic =>

                                            <tr>

                                                {characteristic['textInfo'] ?

<th>{characteristic['textInfo']}</th> : <th></th>}

                                                {characteristic['waveNumber'] ?

<td>{characteristic['waveNumber']}</td> : <td></td>}

                                                {characteristic['intensity'] ?

<td>{characteristic['intensity']}</td> : <td></td>}

                                                {characteristic['uniq'] !== null ?

<td>{characteristic['uniq'] ? 'Правда' : 'Ложь'}</td> : <td></td>}

                                            </tr>

                                        ) : 'no characteristics'}

                                </tbody>

                            </table>

                        </>}

                    </div>

            </div>

        </div>

    )

}

export default Modal;

Рендеринг компонент Modal в компоненте App, передаём props в компонент Modal.

{modalIsOpen &&

      <Modal

              characteristics={characteristics}

                  setModalIsOpen={setModalIsOpen}

                  isLoading={isLoading}

              />}

Добавляем в компонент App, button-элемент (Рисунок - 3.15), и создаём событие onClick и функцию-обработчика события getCharacteristicsFromDB.

<button

              className="btn btn--primary control-ui\_\_btn"

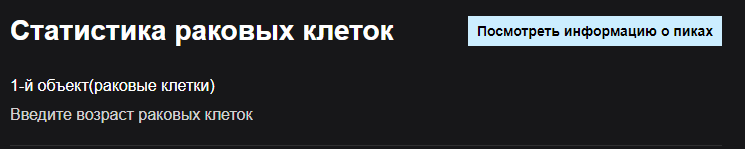
                  onClick={getCharacteristicsFromDB}

                  disabled={isLoading}

              >

              Посмотреть информацию о пиках

              </button>

Рисунок 3.15 – Button-элемент для рендеринга информации о пиках

По клику на button-элемент выполняется функция getCharacteristicsFromDB.

В функции getCharacteristicsFromDB, мы открываем компонент Modal (Рисунок - 3.16) и если state characteristics пустой, отправляем get-запрос по url <http://localhost:3000/data> получаем ответ с информацией о раковых клетках(data), помещаем в state characteristics и рендерим в компоненте Modal информацию о пиках.

Функция getCharacteristicsFromDB:

const getCharacteristicsFromDB = () => {

        setModalIsOpen(true);

        if(!characteristics.length) {

            setIsLoading(true);

            axios.get('/data')

                .then(({data}) => {

                    setCharacteristics(data.characteristics);

                    setIsLoading(false);

                })

        }

    };

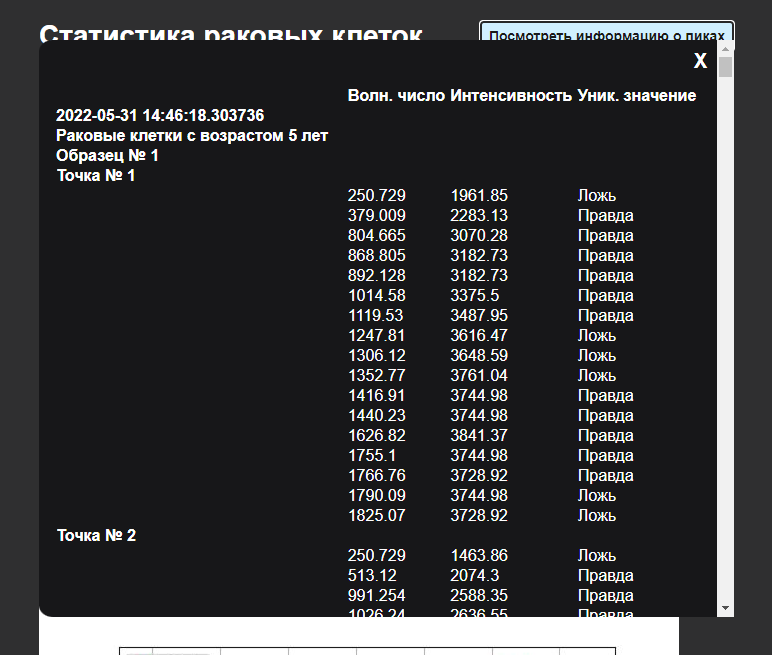


Рисунок 3.16 – Открытие модального окна и рендеринг

информации о пиках

## **3.3 Реализация серверной части**

Задачи серверной части:

1) Получить из тела запроса возраст раковых клеток и образцы изображений раковых клеток для 1-го и 2-го объекта(раковых клеток);

2) Обработка изображения, нахождение и визуализация пиков, нахождение уникальных пиков и сохранение информации о пиках для каждого объекта(раковых клеток) в базу данных

3) Реализация статистики для сравнения объектов(раковых клеток)

4) Возращение в качестве ответа графиков визуализации пиков, результат статистики и информацию о пиках объектов(раковых клеток)



Рисунок 3.17 – Блок-схема работы серверной части

### **3.3.1 Создание Flask сервера**

Создаём папку server в корневой директории что и client и файл api.py, который будет точкой входа в запуске нашего сервера.

Устанавливаем пакет flask в терминале в папке server:

pip3 install flask

Импортируем пакет flask. Создав экземпляр app, мы начинаем использовать его для обработки поступающих веб-запросов и отправки ответов пользователю. @app.route​​​ — это [декоратор](https://en.wikipedia.org/wiki/Python_syntax_and_semantics#Decorators), который превращает стандартную функцию Python в функцию просмотра Flask, конвертирующую возвращаемое значение функции в ответ HTTP, который отображается клиентом HTTP, например веб-браузером. Простейший сервер готов.

from flask import Flask

app = Flask(\_\_name\_\_)

@app.route('/')

def hello():

return 'Hello, World!'

if \_\_name\_\_ == "\_\_main\_\_":

  app.run(debug=True, port=5000);

Пропишем в cmd проекта:

Flask run

Сервер запущен на 5000 порту, остаётся только написать в адресной строке браузера <http://localhost:5000/> и получаем ответ от сервера 'Hello, World!', теперь мы можем отправлять запрос с клиента на наш роут.

Импортируем в api.py необходимые пакеты и модули:

from flask import Flask

from flask import jsonify

from flask import request

import os

import shutil

import base64

from PIL import Image

import io

import pymysql

import datetime

from modules.main import main

Подключаемся к базе данных, указываем: имя хоста, имя пользователя, название базы данных, пароль. Создаём курсор, он используется для перемещения записей из набора результатов.

myDb = pymysql.connect(

    host='127.0.0.1',

    user='root',

    database='db',

    passwd='root'

)

myCursor = myDb.cursor()

Создадим маршрут /files и сопоставим его с методом data\_processing в api.py.

app.route('/files', methods=['POST'])

def data\_processing():

БЛОК КОДА

return ''

На маршрут /files с клиента будут отправляться данные из input-элементов, так что мы можем их вытащить из объекта request в методе data\_processing. Данные содержащие информацию о возрасте раковых клеток и образцах изображений раковых клеток для 1-го и 2-го объекта вытаскиваем в данные переменные.

    firstFiles = request.files.getlist('firstFiles')

    secondFiles = request.files.getlist('secondFiles')

    firstAgeCells = request.values['firstAgeCells']

    secondAgeCells = request.values['secondAgeCells']

Создадим функцию clearImgDirectory, которая будет очищать папку с озображениями. Если такая папка существует, нужно удалисть и создать новую. В качестве параметра path функция принимает путь до папки.

def clearImgDirectory(path):

    dirSamples = path

    if(os.path.isdir(dirSamples)):

      shutil.rmtree(dirSamples)

    os.mkdir(dirSamples)

Вызовы функции clearImgDirectory:

clearImgDirectory('D:/RROJECT/DiplomProject/server/img/samples/')

    clearImgDirectory('D:/RROJECT/DiplomProject/server/img/visualization\_graphs/')

Обьявим переменные в которые будем записывать пути до файлов образцов раковых клеток для 1-го и 2-го объекта.

firstPathFiles = []

secondPathFiles = []

Создадим функцию saveFilesInDirectory, которая будет записывать файлы образцов раковых клеток для 1-го и 2-го объекта в папку. В качестве параметров принимает массив файлов образцов раковых клеток и массив куда будем записывать пути до файлов.

def saveFilesInDirectory(files, arrayToSave):

    for file in files:

      file.save(os.path.join('D:/RROJECT/DiplomProject/server/img/samples/',

file.filename))

      arrayToSave.append(f'img/samples/{file.filename}')

Вызовы функции clearImgDirectory:

saveFilesInDirectory(firstFiles, firstPathFiles)

    saveFilesInDirectory(secondFiles, secondPathFiles)

Для обработки образцов раковых клеток для 1-го и 2-го объекта и для реализации статистики создадим функцию main в которую в качестве аргументов будем прокидывать, к её объявлению вернёмся попозже:

firstAgeCells - возраст образцов раковых клеток для 1-го объекта,

secondAgeCells - возраст образцов раковых клеток для 2-го объекта,

firstPathFiles – пути до файлов образцов раковых клеток для 1-го объекта,

secondPathFiles – пути до файлов образцов раковых клеток для 2-го объекта,

myCursor и myDb – объекты для работы с БД.

Вызов функции main, которая возращает 2 значения статистики:

p, textStatistics = main(firstAgeCells, secondAgeCells, firstPathFiles, secondPathFiles, myCursor, myDb)

Объявим переменную в которую будем записывать файлы визуализации найденых пиков на образцах раковых клеток в бинарном формате.

arrayImages = []

Переведём файлы визуализации образцов раковых клеток в бинарный формат:

for filename in os.listdir('img/visualization\_graphs/'):

      pil\_img = Image.open(f"img/visualization\_graphs/{filename}", mode='r')

      byte\_arr = io.BytesIO()

      pil\_img.save(byte\_arr, format='PNG')

      encoded\_img = base64.encodebytes(byte\_arr.getvalue()).decode('ascii')

      arrayImages.append(encoded\_img)

Создадим функцию get\_characteristicsFromDB, которая будет возращать информацию о найденных пика.

def get\_characteristicsFromDB():

    myCursor.execute("SELECT \* FROM characteristics;")

    rows = myCursor.fetchall()

    characteristics = []

    for row in rows:

        characteristics.append({'idCharacteristic': row[0], 'textInfo': row[1], 'waveNumber': row[2], 'intensity': row[3], 'uniq': row[4]})

    return characteristics

Вывов функции get\_characteristicsFromDB():

characteristics = get\_characteristicsFromDB()

Когда отработает маршрут /files и его с метод data\_processing в api.py, то мы мы будем возращать в качестве ответа данные.

return jsonify({'arrayImages': arrayImages, 'statistics': {'p': p, 'textStatistics': textStatistics}, 'characteristics': characteristics})

Эти данные содержат:

arrayImages – визуализации образцов раковых клеток,

statistics – результат статистики,

characteristics – информацию о найденных пика.

### **3.3.2 Методы нахождения пиков**

1) Попиксельное сравнение пиков

Мы могли бы получить пиксели графика для каждой точки образца. И с помощью цикла for пробежаться по ним и сравнить значения x и y текущего пикселя с предыдущим и последующим пикселем.

pixelsArray = [{'x': 5, 'y': 10}, {'x': 6, 'y': 12}, {'x': 7, 'y': 8},   
{'x': 8, 'y': 10}, {'x': 9, 'y': 9}]

peaks = []

for i, pixel in enumerate(pixelsArray):

if i == 0 or i == len(pixelsArray) - 1:

continue

if(pixelsArray[i]['x'] > pixelsArray[i - 1]['x'] and pixelsArray[i]['x'] < pixelsArray[i + 1]['x']) and (pixelsArray[i]['y'] >

pixelsArray[i - 1]['y'] and pixelsArray[i]['y'] >

pixelsArray[i + 1]['y']):

peaks.append(pixelsArray[i])

print(peaks)

Получим:

[{'x': 6, 'y': 12}, {'x': 8, 'y': 10}]

2) Использование функции find\_peaks, которая принадлежит к пакету signal из библиотеки Scipy.

Эта функция принимает одномерный массив и находит все локальные максимумы путем простого сравнения соседних значений. Необязательно, подмножество этих пиков может быть выбрано путем указания условий для свойств пика.

scipy.signal.find\_peaks(x, height=None, threshold=None, distance=None, prominence=None, width=None, wlen=None, rel\_height=0.5, plateau\_size=None)

Основные параметры:

*Height*: номер высоты или последовательность, необязательно

Требуемая высота пиков. Либо число, либо None, массив, соответствующий x, либо последовательность из 2 элементов первого. Первый элемент всегда интерпретируется как минимальная, а второй, если он указан, как максимальная требуемая высота.

*Threshold*: номер высоты или последовательность, необязательно

Требуемый порог пиков, расстояние по вертикали до соседних образцов. Либо число, либо None, массив, соответствующий x, либо последовательность из 2 элементов первого. Первый элемент всегда интерпретируется как минимальный, а второй, если он указан, как максимальный требуемый порог.

*Distance*: номер, необязательный

Требуемое минимальное горизонтальное расстояние (>= 1) в образцах между соседними пиками. Меньшие пики удаляются первыми до тех пор, пока не будет выполнено условие для всех оставшихся пиков.

Основные выходные данные:

*Peaks*: последовательность

Индексы пиков в x, удовлетворяющие всем заданным условиям.

*Peak\_heights: последовательность*

Если задана высота, то высота каждого пика в x.

В ходе разработки программного кода для нахождения пиков я буду использовать функцию find\_peaks, которая принимает некоторые параметры свойств пиков и возращает индексы самих пиков и свойств пиков.

### **3.3.3 Критерий Крускала-Уоллиса**

«Критерий Крускала-Уоллиса - это непараметрическая альтернатива одномерному (межгрупповому) дисперсионному анализу. Он используется для сравнения трех или более выборок, и проверяет нулевые гипотезы, согласно которым различные выборки были взяты из одного и того же распределения, или из распределений с одинаковыми медианами» [18].

«Таким образом, интерпретация критерия Крускала-Уоллиса в основном сходна с параметрическим одномерным дисперсионным анализом, за исключением того, что этот критерий основан скорее на рангах, чем на средних. Для дополнительных деталей, см. Siegel & Castellan, 1988» [18].

«Этот непараметрический критерий — расширение двухвыборочного критерия Вилкоксона ранговых сумм. При нулевой гипотезе отсутствия различий в распределениях между группами суммы рангов в каждой из *k* групп должны быть сравнимы после учета любых различий в размере выборки» [18].

1. «Определить нулевую и альтернативную гипотезы» [18].

«*H*0: каждая группа имеет одинаковое распределение величин в популяции» [18].

«*H*1: каждая группа не имеет одинакового распределения величин в популяции» [18].

1. «Отобрать необходимые данные из двух взаимосвязанных выборок» [18].
2. «Вычислить величину статистики критерия, отвечающую *H0*,

«Проранжируйте все n значений и рассчитайте сумму рангов в каждой из групп: эти суммы — *R*1…*R*k. Статистика критерия (которая должна быть модифицирована, если имеется много связанных значений) выражается формулой» [18]:

1. «Сравнить значение статистики F-критерия со значением из известного распределения вероятности» [18].
2. «Интерпретировать величину р и результаты» [18].

«Интерпретируйте величину р, и если результат статистически значим, используйте двухвыборочные непараметрические критерии, корректируя их для множественного тестирования. Рассчитайте ДИ для медианы в каждой группе. Однофакторный ANOVA применяют тогда, когда группы соотносятся с одним фактором и независимы. Можно использовать другие виды ANOVA, если план исследования более сложен» [18].

### **3.3.4 Структура БД**

Будем использовать язык запросов MySQL и программу MySQL Workbench для визуального проектирования БД.

Создадим новое соединение (Рисунок - 3.18), введём данные connection name, host, port, username, password для дальнейшей связи с python.

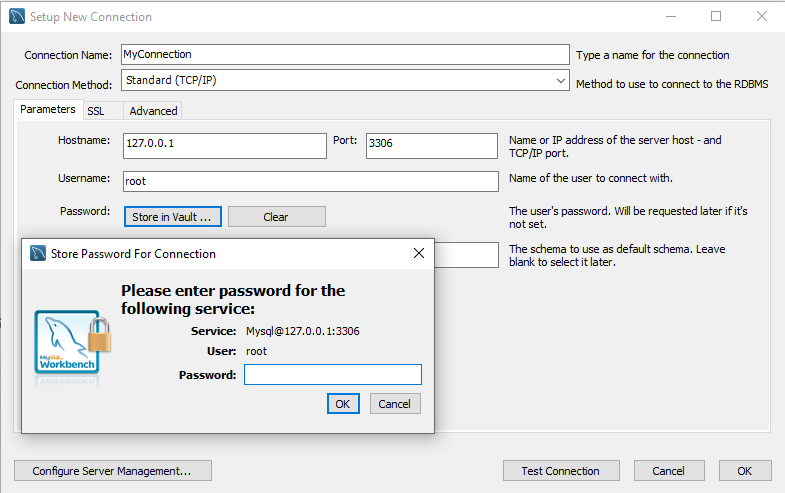


Рисунок 3.18 – Создание нового соединения MyConnection

Создадим новую базу данных с названием db (Рисунок - 3.19).

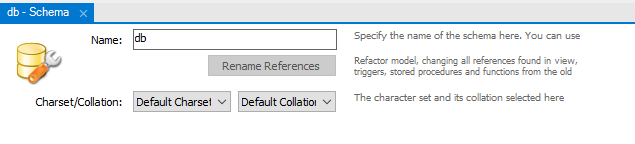


Рисунок 3.19 – Создание новой базы db

Создадим таблицу c названием characteristics (Рисунок - 3.20).

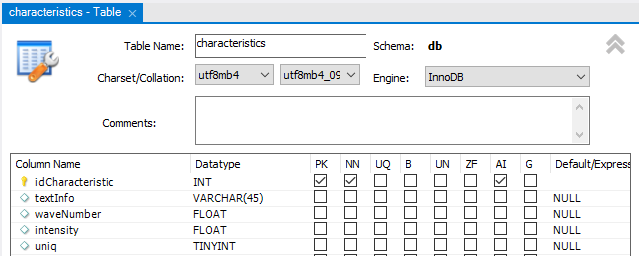


Рисунок 3.20 – Создание новой таблицы characteristics

Столбцы таблицы:

idCharacteristic(INT – целочисленный тип) – id характеристик пика;

textInfo(VARCHAR – символьный тип) – информация о дате и времени статистики, объекте, образце и возрасте раковых клеток;

wavenumber(FLOAT – числовой тип с плавающей точкой) – значение волнового числа пика;

insensity(FLOAT – числовой тип с плавающей точкой) – значение интенсивности пика;

uniq(TINYINT – булевый тип 0 или 1) – значение уникальности пика, среди 5 точек образца раковых клеток.

Примерный вид таблицы characteristics (Рисунок - 3.21).

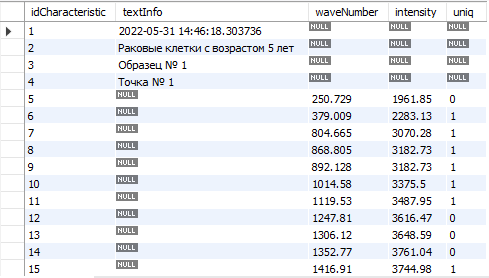


Рисунок 3.21 – Создание новой таблицы characteristics

### **3.3.5 Реализация модулей**

Модуль- это файл, содержащий определения и операторы Python.

Все модули будут находиться в папке modules из которой мы их будем вытаскивать где нужно.

Функцию main вынесем модуль main.py и подключим её в api.py:

from modules.main import main

Создадим функцию imgsProcessing в main модуле, которая будет перебирать образцы раковых клеток для конкретного объекта и прокидывать каждый образец в функцию getResultsFromImg:

def imgsProcessing(imgArray, ageCells, myCursor, myDb):

waveNumberArray = []

    myCursor.execute(f"INSERT INTO characteristics(textInfo)

VALUES('Раковые клетки с возрастом {ageCells}');")

    myDb.commit()

    for i, img in enumerate(imgArray):

      getResultsFromImg(img, i + 1, waveNumberArray, myCursor, myDb)

    return waveNumberArray

Функция main:

def main(firstAgeCells, secondAgeCells, firstPathFiles, secondPathFiles, myCursor, myDb):

  firstWaveNumberArray = imgsProcessing(firstPathFiles, firstAgeCells, myCursor, myDb)

secondWaveNumberArray = imgsProcessing(secondPathFiles, secondAgeCells, myCursor, myDb)

    stat, p = kruskal(firstWaveNumberArray, secondWaveNumberArray)

    textStatistics = ''

  alpha = 0.05

    if p > alpha:

        print('Примеры распределений равны')

        textStatistics = 'Примеры распределений равны'

    else:

        print('Примеры распределений не равны')

        textStatistics = 'Примеры распределений не равны'

    return p, textStatistics

Для реализации статистики будем использовать пакет scipy.stats и метод kruskal, подключим его:

from scipy.stats import kruskal

В качестве аргументов в метод kruskal могут быть переданы два или более массива это будут массивы firstWaveNumberArray и secondWaveNumberArray они будут содержать значения волнового числа пиков для 1-го и 2-го объекта раковых клеток.

На выходе получим p - значение:

\_, p = kruskal(firstWaveNumberArray, secondWaveNumberArray)

В качестве критического значения в большинстве случаев берется 0.05. При p-значении меньше 0.05 мы вынуждены отклонить нулевую гипотезу.

textStatistics = ''

    alpha = 0.05

    if p > alpha:

      print('Примеры распределений равны')

      textStatistics = 'Примеры распределений равны'

    else:

      print('Примеры распределений не равны')

      textStatistics = 'Примеры распределений не равны'

Функцию getResultsFromImg вынесем в модуль и подключим её main.py:

from modules.functions import getResultsFromImg

Для обработки образцов раковых клеток (Рисунок - 3.22) будем использовать функцию getResultsFromImg, вынесенную в отдельный модуль functions.py.

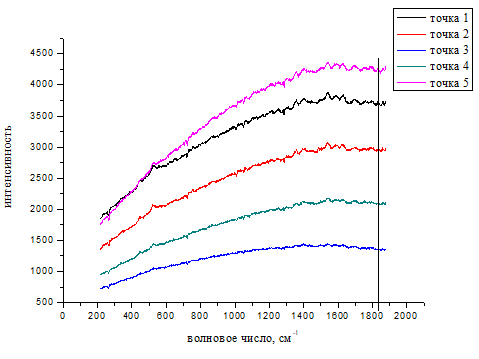


Рисунок 3.22 – Графики зависимости образца

Основная задача функции getResultsFromImg – обработка изображения, нахождение и визуализация пиков, нахождение уникальных пиков и сохранение информации о пиках в БД.

Импортирование зависимостей:

from PIL import Image

import cv2

import pytesseract

import numpy as np

import matplotlib.pyplot as plt

from scipy.signal import find\_peaks

from itertools import groupby

import uuid

import re

Для каждого образца раковых клеток вызываем функцию getResultsFromImg и прокидываем путь до изображения, номер образца, массив waveNumberArray для записи значений волнового числа каждого пика на образцах раковых клеток и объекты для работы с БД.

В данной модуле происходит выполнение вспомогательных функций и инструкций:

1) Считывание картинки и возращение преобразованной копии этого изображения.

    im = Image.open(imgPath)

    im = im.convert('RGB')

    col\_black = (0, 0, 0)

    col\_white = (255, 255, 255)

2) Получение всех чёрных пикселей изображения.

black\_pixels\_X = []

    black\_pixels\_Y = []

    black\_pixels = []

    for y in range(im.size[1]):

      for x in range(im.size[0]):

      if im.getpixel((x, y)) == col\_black:

      black\_pixels\_X.append(x)

  black\_pixels\_Y.append(y)

      black\_pixels.append({'x': x, 'y': y})

3) Нахождение постоянного значения x на котором находится ось Y и y на котором находится ось X и удаление чёрной вертикальной линии, если она присутствует на образце.

axis\_y\_const\_x = findAxisConstValue(black\_pixels\_X, 'x', black\_pixels, im, col\_white)

    axis\_x\_const\_y = findAxisConstValue(black\_pixels\_Y, 'y', black\_pixels, im, col\_white)

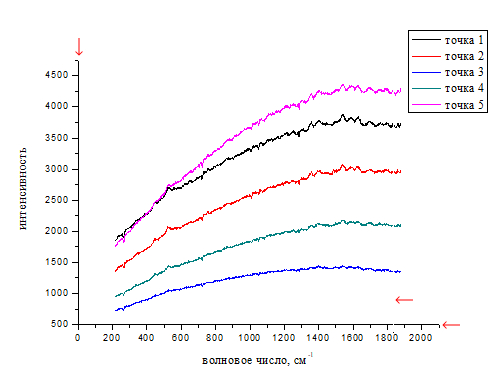


Рисунок - 3.23 Результат функции findAxisConstValue на образце

4) Нахождение всех чёрных пикселей в столбце в котором находятся ось Y и ось X.

    axisYCoords = []

    for pixel in black\_pixels:

      if axis\_y\_const\_x == pixel['x']:

      axisYCoords.append({'x': pixel['x'], 'y': pixel['y']})

    axisXCoords = []

    for pixel in black\_pixels:

    if axis\_x\_const\_y == pixel['y']:

      axisXCoords.append({'x': pixel['x'], 'y': pixel['y']})

5) Удаление лишних чёрных пикселей и получение только пикселей

принадлежащих оси Y и оси X.

axisYCoords = removeUnnecessaryPixels(axisYCoords, 'y')

    axisXCoords = removeUnnecessaryPixels(axisXCoords, 'x')

6) Нахождение начало координат, пересечение оси Y и оси X.

beginning\_of\_coord = 0

    for axisYCoord in axisYCoords:

      for axisXcoord in axisXCoords:

      if axisYCoord['x'] == axisXcoord['x'] and axisYCoord['y'] ==

axisXcoord['y']:

      beginning\_of\_coord = axisXcoord

      break

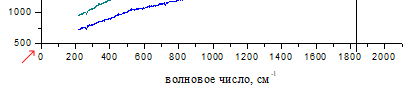


Рисунок – 3.24 Результат функции findAxisConstValue на образце

7) Нахождение порога у маленьких и больших штрихов на оси X и оси Y.

thresholdsX = thresholdStroke('x', im, axis\_x\_const\_y, axisXCoords, col\_black, axis\_y\_const\_x, axisYCoords)

thresholdsY = thresholdStroke('y', im, axis\_x\_const\_y, axisXCoords, col\_black, axis\_y\_const\_x, axisYCoords)

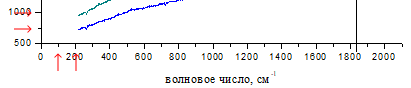


Рисунок – 3.25 Результат функции thresholdStroke на образце

8) Нахождение первого, предпоследнего и последнего большого больших штрихов на оси X и оси Y и предпоследнего маленького штриха на оси X.

firstBigStrokeX, secondLastBigStrokeX, lastBigStrokeX = findStrokes(axisXCoords, 'x', thresholdsX['bigThresholsX'], im, col\_black)

firstBigStrokeY, secondLastBigStrokeY, lastBigStrokeY = findStrokes(axisYCoords, 'y', thresholdsY['bigThresholsY'], im, col\_black)

\_, secondSmallStrokeX, \_ = findStrokes(axisXCoords, 'x', thresholdsX['smallThresholsX'], im, col\_black)

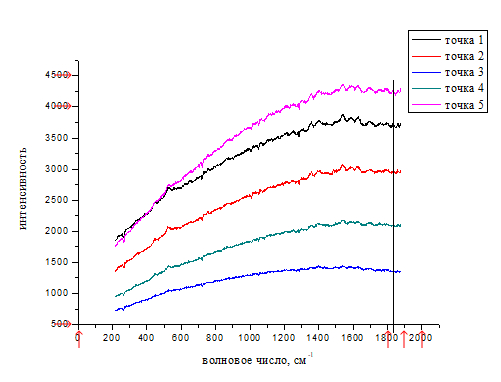


Рисунок - 3.26 Результат функции findStrokes на образце

9) Распознавание максимального и минимального значения интенсивности на оси Y и волнового числа на оси X.

diffBigStrokeY = int(abs((lastBigStrokeY['y'] - secondLastBigStrokeY['y'])   
/ 2))

im\_cropMaxY = im.crop((lastBigStrokeY['x'] - thresholdsX['bigThresholsX'] - 40, lastBigStrokeY['y'] - diffBigStrokeY, lastBigStrokeY['x'] - thresholdsY['bigThresholsY'], lastBigStrokeY['y'] + diffBigStrokeY))

    im\_cropMaxY.save('img/crops/cropMaxY.png')

    diffBigStrokeX = int(abs((lastBigStrokeX['x'] - secondLastBigStrokeX['x'])

/ 2))

im\_cropMaxX = im.crop((lastBigStrokeX['x'] - diffBigStrokeX, lastBigStrokeX['y'] + thresholdsX['bigThresholsX'] + 1, lastBigStrokeX['x'] + diffBigStrokeX, lastBigStrokeX['y'] + 25))

    im\_cropMaxX.save('img/crops/cropMaxX.png')

maxValueY = recognizeValue(cv2.imread("img/crops/cropMaxY.png", cv2.IMREAD\_GRAYSCALE))

maxValueX = recognizeValue(cv2.imread("img/crops/cropMaxX.png", cv2.IMREAD\_GRAYSCALE))

im\_cropMinY = im.crop((firstBigStrokeY['x'] - thresholdsY['bigThresholsY'] - 40, firstBigStrokeY['y'] - diffBigStrokeY, firstBigStrokeY['x'] - thresholdsY['bigThresholsY'], firstBigStrokeY['y'] + diffBigStrokeY))

    im\_cropMinY.save('img/crops/cropMinY.png')

im\_cropMinX = im.crop((firstBigStrokeX['x'] - diffBigStrokeX, firstBigStrokeX['y'] + thresholdsX['bigThresholsX'] + 1, firstBigStrokeX['x'] + diffBigStrokeX, firstBigStrokeX['y'] + 25)) im\_cropMinX.save('img/crops/cropMinX.png')

minValueY = recognizeValue(cv2.imread("img/crops/cropMinY.png", cv2.IMREAD\_GRAYSCALE))

minValueX = recognizeValue(cv2.imread("img/crops/cropMinX.png", cv2.IMREAD\_GRAYSCALE))

10) Рассчитываем коэффициенты для интенсивности и волнового числа(сколько в 1 пикселе значения интенсивности и волнового числа)

img\_1px\_equel\_func\_valueX = (maxValueX - minValueX) / (lastBigStrokeX['x'] - beginning\_of\_coord['x'])

img\_1px\_equel\_func\_valueY = (maxValueY - minValueY) / (firstBigStrokeY['y'] - lastBigStrokeY['y'])

11) Вырезаем картинку с графиками

cropIm = im.crop((lastBigStrokeY['x'] + 1, 0, secondSmallStrokeX['x'], secondSmallStrokeX['y']))

    cropIm.save("img/crops/cropIm.bmp")

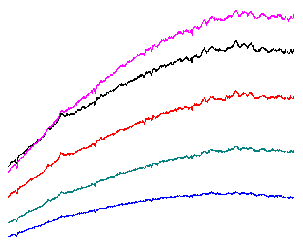


Рисунок - 3.27 Результат вырезания на образце

11) Нахождение пикселей по цвету для каждой точки данного образца

    firstPointArray = findPixelsForPoint((0, 0, 0), cropIm)

    secondPointArray = findPixelsForPoint((255, 0, 0), cropIm)

    thirdPointArray = findPixelsForPoint((0, 0, 255), cropIm)

    fourthPointArray = findPixelsForPoint((0, 128, 128), cropIm)

    fifthPointArray = findPixelsForPoint((255, 0, 255), cropIm)

12) Получаем пики для каждой точки образца и визуализируем их

firstPointArrayPeaks, pointNumberImg = getPeaks(firstPointArray, img\_1px\_equel\_func\_valueX, firstBigStrokeY, beginning\_of\_coord, cropIm, img\_1px\_equel\_func\_valueY, minValueY, imgPath, pointNumberImg)

secondPointArrayPeaks, pointNumberImg = getPeaks(secondPointArray, img\_1px\_equel\_func\_valueX, firstBigStrokeY, beginning\_of\_coord, cropIm, img\_1px\_equel\_func\_valueY, minValueY, imgPath, pointNumberImg)

thirdPointArrayPeaks, pointNumberImg = getPeaks(thirdPointArray, img\_1px\_equel\_func\_valueX, firstBigStrokeY, beginning\_of\_coord, cropIm, img\_1px\_equel\_func\_valueY, minValueY, imgPath, pointNumberImg)

fourthPointArrayPeaks, pointNumberImg = getPeaks(fourthPointArray, img\_1px\_equel\_func\_valueX, firstBigStrokeY, beginning\_of\_coord, cropIm, img\_1px\_equel\_func\_valueY, minValueY, imgPath, pointNumberImg)

fifthPointArrayPeaks, pointNumberImg = getPeaks(fifthPointArray, img\_1px\_equel\_func\_valueX, firstBigStrokeY, beginning\_of\_coord, cropIm, img\_1px\_equel\_func\_valueY, minValueY, imgPath, pointNumberImg)

13) Нахождение уникальных пиков(пики, которые не повторяются в других точках данного образца)

firstPointArrayPeaks = setUniqPeaks(firstPointArrayPeaks,

[secondPointArrayPeaks, thirdPointArrayPeaks, fourthPointArrayPeaks,

      fifthPointArrayPeaks])

    secondPointArrayPeaks = setUniqPeaks(secondPointArrayPeaks,

      [firstPointArrayPeaks, thirdPointArrayPeaks, fourthPointArrayPeaks,

      fifthPointArrayPeaks])

thirdPointArrayPeaks = setUniqPeaks(thirdPointArrayPeaks,

      firstPointArrayPeaks, secondPointArrayPeaks, fourthPointArrayPeaks,

      fifthPointArrayPeaks])

    fourthPointArrayPeaks = setUniqPeaks(fourthPointArrayPeaks,

      [firstPointArrayPeaks, secondPointArrayPeaks, thirdPointArrayPeaks,

      fifthPointArrayPeaks])

    fifthPointArrayPeaks = setUniqPeaks(fifthPointArrayPeaks,

      [firstPointArrayPeaks, secondPointArrayPeaks, thirdPointArrayPeaks,

      fourthPointArrayPeaks])

14) Добавляем в waveNumberArray значения волнового числа пиков, этот массив нужен для реализации статистики

writeToWaveNumberArray(waveNumberArray, firstPointArrayPeaks)

    writeToWaveNumberArray(waveNumberArray, secondPointArrayPeaks)

    writeToWaveNumberArray(waveNumberArray, thirdPointArrayPeaks)

    writeToWaveNumberArray(waveNumberArray, fourthPointArrayPeaks)

    writeToWaveNumberArray(waveNumberArray, fifthPointArrayPeaks)

15) Запись информации о пиках в БД

pointNumber = 1

myCursor.execute(f"INSERT INTO characteristics(textInfo) VALUES('Образец № {sampleNumber}');")

    myDb.commit()

    pointNumber = writeToDB(firstPointArrayPeaks, pointNumber, myCursor, myDb)

    pointNumber = writeToDB(secondPointArrayPeaks, pointNumber, myCursor, myDb)

    pointNumber = writeToDB(thirdPointArrayPeaks, pointNumber, myCursor, myDb)

    pointNumber = writeToDB(fourthPointArrayPeaks, pointNumber, myCursor, myDb)

    pointNumber = writeToDB(fifthPointArrayPeaks, pointNumber, myCursor, myDb)

Серверная часть готова, были реализованы эндпоинты /files и /data на которые с клиента можно делать запросы, а также были выполнены задачи серверной части.

# 4. ТЕХНИКА БЕЗОПАСНОСТИ ПРИ РАБОТЕ ЗА ПЕРСОНАЛЬНЫМ КОМПЬЮТЕРОМ

Персональный компьютер — электроприбор. От прочих электроприборов он отличается тем, что для него предусмотрена возможность длительной эксплуатации без отключения от электрической сети. Кроме обычного режима работы компьютер может находиться в режиме работы с пониженным электропотреблением или в дежурном режиме ожидания запроса. В связи с возможностью продолжительной работы компьютера без отключения от электросети следует уделить особое внимание качеству организации электропитания.

## **4.1 Требования по электрической безопасности**

Недопустимо использование некачественных и изношенных компонентов в системе электроснабжения, а также их суррогатных заменителей: розеток, удлинителей, переходников, тройников. Недопустимо самостоятельно модифицировать розетки для подключения вилок, соответствующих иным стандартам. Электрические контакты розеток не должны испытывать механических нагрузок, связанных с подключением массивных компонентов (адаптеров, тройников и т. п.).

Все питающие кабели и провода должны располагаться с задней стороны компьютера и периферийных устройств. Их размещение в рабочей зоне пользователя недопустимо.

Запрещается производить какие-либо операции, связанные с подключением, отключением или перемещением компонентов компьютерной системы без предварительного отключения питания.

Компьютер не следует устанавливать вблизи электронагревательных приборов и систем отопления.

Недопустимо размещать на системном блоке, мониторе и периферийных устройствах посторонние предметы: книги, листы бумаги, салфетки, чехлы для защиты от пыли. Это приводит к постоянному или временному перекрытию вентиляционных отверстий.

Запрещается внедрять посторонние предметы в эксплуатационные или вентиляционные отверстия компонентов компьютерной системы.

Особенности электропитания монитора. Монитор имеет элементы, способные сохранять высокое напряжение в течение длительного времени после отключения от сети. Вскрытие монитора пользователем недопустимо ни при каких условиях. Это не только опасно для жизни, но и технически бесполезно, так как внутри монитора нет никаких органов, регулировкой или настройкой которых пользователь мог бы улучшить его работу. Вскрытие и обслуживание мониторов может производиться только в специальных мастерских.

## **4.2 Особенности электропитания системного блока**

Все компоненты системного блока получают электроэнергию от блока питания. Блок питания ПК — это автономный узел, находящийся в верхней части системного блока. Правила техники безопасности не запрещают вскрывать системный блок, например при установке дополнительных внутренних устройств или их модернизации, но это не относится к блоку питания. Блок питания компьютера — источник повышенной пожаро-опасности, поэтому вскрытию и ремонту он подлежит только в специализированных мастерских.

Блок питания имеет встроенный вентилятор и вентиляционные отверстия. В связи с этим в нем неминуемо накапливается пыль, которая может вызвать короткое замыкание. Рекомендуется периодически (один - два раза в год) с помощью пылесоса удалять пыль из блока питания через вентиляционные отверстия без вскрытия системного блока. Особенно важно производить эту операцию перед каждой транспортировкой или наклоном системного блока.

## **4.3 Система гигиенических требований**

Длительная работа с компьютером может приводить к расстройствам состояния здоровья. Кратковременная работа с компьютером, установленным с грубыми нарушениям гигиенических норм и правил, приводит к повышенному утомлению. Вредное воздействие компьютерной системы на организм человека является комплексным. Параметры монитора оказывают влияние на органы зрения. Оборудование рабочего места влияет на органы опорно-двигательной системы. Характер расположения оборудования в компьютерном классе и режим его использования влияет как на общее психофизиологическое состояние организма, так и им органы зрения.

В прошлом монитор рассматривали в основном как источник вредных излучений, воздействующих прежде всего на глаза. Сегодня такой подход считается недостаточным. Кроме вредных электромагнитных излучений (которые на современных мониторах понижены до сравнительно безопасного уровня) должны учитываться параметры качества изображения, а они определяются не только монитором, но и видеоадаптером, то есть всей видеосистемы в целом.

Монитор компьютера должен удовлетворять следующим между народным стандартам безопасности:

* по уровню электромагнитных излучений — ТСО 95
* по параметрам качества изображения (яркость, контрастность, мерцание, антибликовые свойства и др. ) — ТСО 99

Узнать о соответствии конкретной модели данным стандартам можно в сопроводительной документации. Для работы с мониторами, удовлетворяющими данным стандартам, специальные защитные экраны не требуется.

На рабочем месте монитор должен устанавливаться таким образом, чтобы исключить возможность отражения от его экрана в сторону пользователя источников общего освещения помещения.

Расстояние от экрана монитора до глаз пользователя должно составлять от 50 до 70 см. Оптимально размещение монитора на расстоянии 1,5 D от глаз пользователя, где D — размер экрана монитора, измеренный по диагонали. Завыщенное расстояния от глаз до монитора приводит к дополнительному напряжению органов зрения, сказывается на затруднении перехода от работы с монитором к работе с книгой и проявляется в преждевременном развитии дальнозоркости.

Важным параметром является частота кадров, которая зависит от свойств монитора, видеоадаптера и программных настроек видеосистемы. Для работы с текстами минимально допустима частота кадров 72 Гц. Для работы с графикой рекомендуется частота кадров от 85 Гц и выше.

## **4.4 Требования к рабочему месту**

В требования к рабочему месту входят требования к рабочему столу, посадочному месту (стулу, креслу), Подставкам для рук и ног. Несмотря на кажущуюся простоту, обеспечить правильное размещение элементов компьютерной системы и правильную посадку пользователя чрезвычайно трудно. Полное решение проблемы требует дополнительных затрат, сопоставимых по величине со стоимостью отдельных узлов компьютерной системы, поэтому и биту и на производстве этими требованиями часто пренебрегают.

Монитор должен быть установлен прямо перед пользователем и не требовать поворота головы или корпуса тела.

Рабочий стол и посадочное место должны иметь такую высоту, чтобы уровень глаз пользователя находился чуть выше центра монитора. На экран монитора следует смотреть сверху вниз, а не наоборот. Даже кратковременная работа с монитором, установленным слишком высоко, приводит к утомлению шейных отделов позвоночника.

Если при правильной установке монитора относительно уровня глаз выясняется, что ноги пользователя не могут свободно покоиться на полу, следует установить подставку для ног, желательно наклонную. Если ноги не имеют надежной опоры, это непременно ведет к нарушению осанки и утомлению позвоночника. Удобно, когда компьютерная мебель (стол и рабочее кресло) имеют средства для регулировки по высоте. В этом случае проще добиться оптимального положения.

Клавиатура должна быть расположена на такой высоте, чтобы пальцы рук располагались на ней свободно, без напряжения, а угол между плечом и предплечьем составлял 100° — 110°. При использовании обычных школьно-письменных столов добиться одновременно правильного положения и монитора, и клавиатуры практически невозможно. Для работы рекомендуется использовать специальные компьютерные столы, имеющие выдвижные полочки для клавиатуры.

При длительной работе с клавиатурой возможно утомление сухожилий кистевого сустава. Известно тяжелое профессиональное заболевание — кистевой туннельный синдром, связанное с неправильным положением рук на клавиатуре. Во избежание чрезмерных нагрузок на кисть желательно предоставить рабочее кресло с подлокотниками, уровень высоты которых, замеренный от пола, совпадает с уровнем высоты расположения клавиатуры.

При работе с мышью рука не должна находиться на весу. Локоть руки или хотя бы запястье должны иметь твердую опору. Если предусмотреть необходимое расположение рабочего стола и кресла затруднительно, рекомендуется применить коврик для мыши, имеющий специальный опорный валик. Нередки случаи, когда в поисках опоры для руки (обычно правой) располагают монитор сбоку от пользователя (соответственно, слева), чтобы он работал вполоборота, опирая локоть или запястье правой руки о стол. Этот прием недопустим. Монитор должен обязательно находиться прямо перед пользователем.

## **4.5 Требования к организации пространства**

Экран монитора — не единственный источник вредных электромагнитных излучений. Разработчики мониторов достаточно давно и успешно занимаются их преодолением. Меньше внимания уделяется вредным побочным излучениям, возникающим со стороны боковых и задней стенок оборудования. В современных компьютерных системах эти зоны наиболее опасны.

Монитор компьютера следует располагать так, чтобы задней стенкой он был обращен не к людям, а к стене помещения. В компьютерных помещениях, имеющих несколько компьютеров, рабочие места должны располагаться по периферии помещения, оставляя свободным центр. При этом дополнительно необходимо проверить каждое из рабочих мест на отсутствие прямого отражения внешних источников освещения. Как правило, добиться этого для всех рабочих мест одновременно достаточно трудно. Возможное решение состоит в использовании штор на окнах и продуманном размещении искусственных источников общего и местного освещения.

Сильными источниками электромагнитных излучений являются устройства бесперебойного питания. Располагать их следует как можно дальше от посадочных мест пользователей.

# 5. Энергосбережение

Энергосбережение — реализация правовых, организационных, научных, производственных, технических и экономических мер, направленных на эффективное (рациональное) использование (и экономное расходование) топливно-энергетических ресурсов и на вовлечение в хозяйственный оборот возобновляемых источников энергии. При разработке ПО самые большие траты приходятся на электроэнергию, потребляемую компьютерами.

## **5.1** **Основные способы экономии энергии**

Ключевыми мероприятиями оптимизации потребления электроэнергии на освещение являются:

* максимальное использование дневного света (повышение прозрачности и увеличение площади окон, дополнительные окна);
* повышение отражающей способности (белые стены и потолок);
* оптимальное размещение световых источников (местное освещение, направленное освещение);
* использование осветительных приборов только по необходимости;
* повышение светоотдачи существующих источников (замена люстр, плафонов, удаление грязи с плафонов, применение более эффективных отражателей);
* замена ламп накаливания на энергосберегающие (люминесцентные, компактные люминесцентные, светодиодные);
* применение устройств управления освещением (датчики движения и акустические датчики, датчики освещенности, таймеры, системы дистанционного управления);
* внедрение автоматизированной системы диспетчерского управления наружным освещением (АСДУ НО);
* установка интеллектуальных распределённых систем управления освещением (минимизирующих затраты на электроэнергию для данного объекта).

## **5.2 Энергосберегающие режимы работы компьютера**

Спящий режим — это режим пониженного потребления электроэнергии, который позволяет быстро возобновить работу в режиме обычного потребления энергии (обычно в течение нескольких секунд) по требованию пользователя.

Перевод компьютера в спящий режим напоминает нажатие кнопки "Пауза" на проигрывателе DVD — компьютер немедленно останавливает все операции и готов к возврату в рабочий режим при необходимости.

Режим гибернации — это режим пониженного потребления электроэнергии, разработанный в первую очередь для ноутбуков. При переходе в спящий режим все открытые документы и параметры сохраняются в памяти и компьютер переходит в режим пониженного потребления электроэнергии, а при переходе в режим гибернации все открытые документы и программы сохраняются на жестком диске и затем компьютер выключается. Из всех энергосберегающих режимов, используемых в ОС Windows, для поддержания режима гибернации требуется наименьшее количество электроэнергии. Если в течение длительного промежутка времени не планируется использовать ноутбук и нет возможности подзарядить батарею, рекомендуется перевести ноутбук в режим гибернации.

Гибридный спящий режим — это режим, который разработан преимущественно для настольных компьютеров. Гибридный спящий режим сочетает в себе спящий режим и режим гибернации, поскольку все открытые документы и программы сохраняются в памяти и на жестком диске и компьютер переводится в режим пониженного потребления электроэнергии. При неожиданном сбое питания операционная система Windows может легко восстановить данные с диска. Если гибридный спящий режим включен, переход в спящий режим автоматически переводит компьютер в гибридный спящий режим. На настольных компьютерах гибридный спящий режим обычно включен по умолчанию.

Таким образом, наиболее эффективное энергосбережение достигается при применении спящего режима в сочетании с режимом гибернации, либо при применении гибридного спящего режима.

## **5.3 Энергоэффективность в серверных помещениях**

Обслуживание касается как серверного оборудования, так и системы кондиционирования. Нужно поддерживать правильные настройки. Например, не нужно задавать на кондиционере слишком низкую температуру, которая не нужна для работы серверов. Нужно следить за расположением коммутационных кабелей в серверных шкафах. Обычно со временем клубок кабелей копится, потоки воздуха начинают испытывать излишнее сопротивление. Пустые юниты в стойках стоит закрывать панелями-заглушками, чтобы холодный воздух не перемешивался с горячим. Нужно вовремя чистить и мыть оборудование, т.к. грязный наружный блок кондиционера или грязные воздушные фильтры во внутренних блоках существенно снижают теплообмен. Если не следить за всем выше перечисленным, существенно увеличиваются энергозатраты (компрессор и вентиляторы вынуждены работать более производительно).

## **5.4 Стандарты энергосберегающих технологий**

В начале 90-х годов компания EPA (Environmental Protection Agency — Агентство по защите окружающей среды) начало проводить кампанию по сертификации энергосберегающих персональных компьютеров и периферийного оборудования. Компьютер или монитор во время продолжительного простоя должен снизить энергопотребление до 30 Вт и более. Система, удовлетворяющая этим требованиям, может получить сертификат Energy Star.

### **5.4.1 Стандарт усовершенствованной системы управления**

### **питанием (Advanced Power Management — APM)**

Разработан фирмой Intel совместно с Microsoft и определяет ряд интерфейсов между аппаратными средствами управления питанием и операционной системой компьютера. Полностью реализованный стандарт APM позволяет автоматически переключать компьютер между пятью состояниями в зависимости от текущего состояния системы. Каждое последующее состояние в приведенном ниже списке характеризуется уменьшением потребления энергии.

* Full On

Система полностью включена.

* APM Enabled

Система работает, некоторые устройства являются объектами управления для системы управления питанием. Неиспользуемые устройства могут быть выключены, может быть также остановлена или замедлена (т.е. снижена тактовая частота) работа тактового генератора центрального процессора.

* APM Standby (резервный режим)

Система не работает, большинство устройств находятся в состоянии потребления малой мощности. Работа тактового генератора центрального процессора может быть замедлена или остановлена, но необходимые параметры функционирования хранятся в памяти. Пользователь или операционная система могут запустить компьютер из этого состояния почти мгновенно.

* APM Suspend (режим приостановки).

Система не работает, большинство устройств пассивны. Тактовый генератор центрального процессора остановлен, а параметры функционирования хранятся на диске и при необходимости могут быть считаны в память для восстановления работы системы. Чтобы запустить систему из этого состояния, требуется некоторое время.

* Off (система отключена).

Система не работает. Источник питания выключен.

Для реализации режимов APM требуются аппаратные средства и программное обеспечение. Источниками питания ATX можно управлять с помощью сигнала Power\_On и факультативного разъема питания с шестью контактами. (Необходимые для этого команды выдаются программой.) Изготовители также встраивают подобные устройства управления в другие элементы системы, например в системные платы, мониторы и дисководы. Операционные системы (такие как Windows), которые поддерживают APM, при наступлении соответствующих событий запускают программы управления питанием, наблюдая за действиями пользователя и прикладных программ. Однако операционная система непосредственно не посылает сигналы управления питанием аппаратным средствам. Система может иметь множество различных аппаратных устройств и программных функций, используемых при выполнении функций APM. Чтобы разрешить проблему сопряжения этих средств в операционной системе и аппаратных средствах предусмотрен специальный абстрактный уровень, который облегчает связь между различными элементами архитектуры

При запуске операционной системы загружается программа — драйвер APM, который связывается с различными прикладными программами и программными функциями. Именно они запускают действия управления питанием, причем все аппаратные средства, совместимые с APM, связываются с системной BIOS. Драйвер APM и BIOS связаны напрямую; именно эту связь использует операционная система для управления режимами аппаратных средств.

Таким образом, чтобы функционировали средства APM, необходим стандарт, поддерживаемый схемами, встроенными в конкретные аппаратные устройства системы, системная BIOS и операционная система с драйвером APM. Если хотя бы один из этих компонентов отсутствует, APM работать не будет.

### **5.4.2 Стандарт усовершенствованной системы управления**

### **питанием (Advanced Power Management—APM)**

Усовершенствованная конфигурация и интерфейс питания (Advanced Configuration and Power Interface— ACPI) впервые реализованы в современных BIOS и операционных системах Windows 98 и более поздних. Если BIOS компьютера поддерживает систему ACPI, то все управление питанием передается операционной системе. Это упрощает конфигурирование параметров, все они находятся в одном месте — в операционной системе.

## **5.5 Энергосберегающие технологии в мобильных ПК**

Для решения проблемы энергопотребления в корпорации Intel были созданы специальные версии мобильных процессоров, например Intel Pentium III-M, Intel Pentium 4-M и процессор Intel Pentium M для мобильных ПК с поддержкой технологии Intel Centrino. Они отличаются от своих собратьев для стационарных ПК (исключение составляет процессор Intel Pentium M, не имеющий аналога для стационарного ПК) средствами управления энергопотреблением, позволяющими увеличить продолжительность автономной работы ноутбука на мобильном процессоре. К таким средствам относится:

* технология Enhanced Intel SpeedStep;
* режимы ожидания Deep Sleep и Deeper Sleep;
* технология Intel Mobile Voltage Positioning (IMVP).

Использование мобильных версий процессоров позволяет отчасти решить проблему производительности в совокупности с увеличением времени работы ноутбука от батареи.

### **5.5.1 Технология Enhanced Intel SpeedStep**

Улучшенная технология SpeedStep (Enhanced Intel SpeedStep) дает пользователям возможность увеличить время автономной работы от батареи за счет динамического изменения напряжения ядра процессора и его тактовой частоты. Изменение условий работы процессора зависит от его загрузки (степени утилизации), от температурного режима, а также от установленных пользователем предпочтений через задания схемы энергопотребления (Power Schemes) в настройках операционной системы.

В отличие от предыдущей версии технологии Intel SpeedStep, предусматривающей возможность работы мобильного процессора лишь на двух тактовых частотах, улучшенная технология Enhanced Intel SpeedStep определяет использование нескольких возможных напряжений питания и частот (в совокупности — рабочих точек), что позволяет достичь лучшего соотношения «напряжение/частота» и более эффективного режима функционирования, когда производительность согласуется с рабочей нагрузкой.

Крайние рабочие точки процессора задаются аппаратно, а промежуточные точки устанавливаются программно. Управление переходами между различными рабочими точками выполняется только самим процессором и блоком регулятора напряжения (VRM).

Для установки требуемого напряжения процессор Intel Pentium M посылает служебные VID-последовательности непосредственно в VRM-модуль. При этом не используются никакие другие компоненты системы при осуществлении перехода между рабочими состояниями процессора.

Переход между различными рабочими точками процессора, характеризующимися напряжением и частотой, происходит таким образом, чтобы обеспечивать работоспособность процессора в процессе самого перехода (который не может осуществляться мгновенно). Для того чтобы осуществить переход на более высокую тактовую частоту, сначала до требуемого уровня меняется напряжение процессора. Процесс изменения напряжения длится порядка 100 мкс, то есть является достаточно длительным. Чтобы сохранить работоспособность процессора при изменении напряжения, частота процессора при этом не меняется. Когда же напряжение изменится и достигнет требуемого уровня, происходит скачкообразное увеличение частоты процессора, которое длится порядка 10 мкс. Если требуется осуществить переход к меньшей частоте, сначала происходит практически мгновенное изменение частоты (в течение 10 мкс), а после этого постепенно уменьшается напряжение самого процессора — уже при неизменной частоте.

Всего в технологии Enhanced Intel SpeedStep рассматриваются четыре схемы энергопотребления:

* Maximum Performance Mode;
* Automatic Mode;
* Battery-Optimized Performance Mode;
* Maximum Battery Mode.

Схема Maximum Performance Mode — это режим по умолчанию работы ноутбука в случае питания от сети (внешнего источника питания). В этом режиме процессор работает на максимальной тактовой частоте, что обеспечивает максимальную производительность.

Схема Automatic Mode является схемой по умолчанию при автономной работе ноутбука от аккумуляторной батареи. В данном режиме средствами операционной системы определяется степень загруженности процессора и в зависимости от полученного значения динамически устанавливаются требуемые значения тактовой частоты и напряжения ядра процессора. Тем самым режим Automatic Mode обеспечивает баланс между производительностью ноутбука и временем автономной работы от батареи. Отметим также, что режим Automatic Mode автоматически устанавливается и при выборе схемы Battery-Optimized Performance Mode, если температура процессора превышает допустимый уровень, заданный в настройках BIOS.

Battery-Optimized Performance Mode — это режим работы ноутбука, устанавливаемый программным способом средствами операционной системы (Windows XP/Me/2000) через настройки схемы энергопотребления (Power Schemes). В данном режиме работы тактовая частота и напряжение процессора при выполнении им многих нересурсоемких задач понижаются до минимального значения, что позволяет существенно снизить энергопотребление (и соответственно увеличить время автономной работы от батареи) по сравнению с режимом работы процессора на номинальной тактовой частоте.

Maximum Battery Mode. Данный режим, так же как и режим Battery-Optimized Performance Mode, устанавливается программным способом. При его выборе тактовая частота и напряжение процессора понижаются до минимального значения, что позволяет значительно снизить энергопотребление. Следует отметить, что в данном режиме процессор работает на пониженной тактовой частоте при любой степени загрузки. В результате за счет снижения производительности достигается максимально возможное время автономной работы от аккумуляторной батареи. Данный режим предназначен для тех случаев, когда для пользователей наиболее критично именно время автономной работы от батареи, даже в ущерб производительности ноутбука.

Автоматическое переключение между различными схемами энергопотребления (например, при отключении внешнего питания) происходит незаметно для пользователя, так как для этого требуется менее 0,001 с. Естественно, сам процесс переключения не нарушает режима работы всех запущенных приложений. Кроме того, переключение между различными режимами работы возможно и вручную. При использовании операционной системы Windows XP установка требуемого режима работы осуществляется через настройки схемы питания (Power Schemes) в диалоговом окне Power Options.

### **5.5.2 Режимы Deep Sleep и Deeper Sleep**

Другими средствами энергосбережения, реализованными в мобильных процессорах, являются технологии Deep Sleep (глубокий сон) и ее усовершенствованная версия Deeper Sleep (еще более глубокий сон). Технология Deep Sleep известна также как режим C3 ACPI, а технология Deeper Sleep — как режим C4 ACPI.

Эти технологии позволяют процессору динамически переключаться в режим минимально возможного энергопотребления. Так, для процессоров семейства Intel Pentium M в режиме Deeper Sleep напряжение питания ядра составляет от 0,705 до 0,785 В.

Переход в состояние Deeper Sleep происходит каждый раз, когда регулятор напряжения понижает напряжение ядра процессора по сигналу, получаемому от хаба ввода-вывода (I/O hub).

Несмотря на то что режим Deeper Sleep позволяет снизить общее энергопотребление компьютера, он никак не отражается на его производительности. Дело в том, что динамическое переключение в режим «спячки» происходит только в том случае, если система неактивна. К примеру, когда пользователь набирает текст, то в промежутках между нажатием клавиш (для компьютера это очень большие периоды времени) ноутбук неактивен и может динамически переключаться в режим Deeper Sleep. Вообще, переключение в режим Deeper Sleep происходит каждый раз, когда система неактивна менее 1 мс. Обратный переход из режима Deeper Sleep происходит практически мгновенно, как только система начинает проявлять активность.

В принципе режим Deeper Sleep полностью идентичен режиму Deep Sleep — за тем лишь исключением, что в режиме Deeper Sleep напряжение питания процессора снижается на 30% больше, чем в режиме Deep Sleep.

### **5.5.3 Технология Intel Mobile Voltage Positioning (IMVP)**

Intel Mobile Voltage Positioning (IMVP) — это технология интеллектуального регулирования напряжения (smart voltage regulation), позволяющая снижать напряжение ядра процессора при одновременном повышении питающего тока, что дает возможность поддерживать требуемый уровень производительности при одновременном снижении энергопотребления, а также обеспечивать условия, необходимые для режимов Deep Sleep и Deeper Sleep.

Технология IMVP также оказывает влияние на тепловыделение процессора (Thermal Design Power, TDP), которое должно поддерживаться в заданных пределах. Снижение TDP позволяет производителям ноутбуков использовать более мощные процессоры в тонких и компактных ноутбуках.

Не так давно корпорация Intel представила улучшенную версию технологии IMVP, которая называется IMVP-IV. В технологии IMVP-IV применяются инновационные методы, позволяющие еще больше снизить требования по энергопотреблению и TDP процессоров. Новую технологию поддерживают мобильные процессоры Intel Pentium 4-M и Intel Pentium M.

# 6. ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ

В разделе представлена информация, из которой выводится целесообразность (или нецелесообразность) создания ПО. Содержит анализ затрат и результатов проекта. Позволяет определить, стоит ли вкладывать деньги в предлагаемый проект.

## **6.1 Оптимальные характеристики оборудования**

Как инструмент разработки программ использовался ноутбук Dell Inspiron 15-3567 со следующими ключевыми техническими характеристиками:

* Процессор Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz
* Встроенный дисплей с разрешением 1920x1080
* ОЗУ DDR4 SODIMM 8 Гбайт

Отметим пункты для значительного увеличения скорости разработки программ:

* Наличие дополнительного монитора
* Мониторы с высоким разрешением экрана
* Высокопроизводительный многоядерный процессор
* Большой размер ОЗУ (16Гбайт - 32Гбайт)
* Наличие SSD в качестве основного дискового пространства (для операционной системы и запуска программ)

Указанные пункты помогут уменьшить время разработки программ за счет увеличения скорости работы сборщика проектов, возможности одновременного доступа к нескольким источникам информации без необходимости переключения внимания и скорости проведения тестов разработанного ПО за счет возможности параллельной работы сразу с несколькими эмуляторами вместо последовательной в случае работы с устаревшим оборудованием.

Минимальные системные требования:

* Процессор 2ГГц
* Разрешение экрана 1024х768
* ОЗУ 2Гбайта
* Наличие монитора, клавиатуры и мыши

## **6.2 Расчет себестоимости программного обеспечения**

При расчете себестоимости необходимо учитывать целый ряд технико-экономических показателей, к которым относятся время разработки, размер и объем программного кода.

Совокупные затраты на создание скрипта состоят из нескольких компонентов. При расчете все эти компоненты необходимо приводить к одной единице измерения с целью обеспечения их однородности и удобства сопоставления. В качестве такой единицы удобнее всего рассматривать денежные выражения либо человеко-часы.

При разработке программных комплексов наибольшее значение в составе затрат имеют следующие составляющие:

1. затраты на подготовку и применение технологии и программных средств автоматизации разработки программ;
2. затраты на проектирование, программирование, отладку и испытания программ согласно требований пользователя или заказчика;
3. затраты на ЭВМ, используемые при разработке данного программного продукта;
4. затраты на изготовление опытного образца программного продукта как продукции производственно–технического назначения;
5. затраты на подготовку и повышение квалификации специалистов–разработчиков.

Затраты типа 1 и 4 можно исключить, учитывая условия и характер разработки данного программного продукта. Затраты типа 5 тоже не учитываются, так как их наиболее трудно учитывать и формализовать в конкретной работе. Имеются и дополнительные составляющие стоимости, которые не учитываются исходя из условий работы. К этим составляющим относятся: заработная плата, мебель, коммунальные услуги, охрана, противопожарные мероприятия, налоги и т.д.

Для определения составляющих затрат труда воспользуемся расчетом условного числа выполняемых строк программного кода в программе:

, (1)

где *с* = 1,2— коэффициент увеличения затрат, характеризует увеличение затрат труда вследствие недостаточно полного описания задачи, уточнений и некоторой доработки. Этот коэффициент может принимать значения от 1,2 до 5;

*p* = 0.08 — коэффициент коррекции программы в ходе ее разработки, принимает значения от 0.05 до 0.1;

*q* = 801— число строк программного кода.

В итоге получаем, что условное число операторов в программе принимает значение:



Важной составляющей при расчете себестоимости программного продукта является коэффициент квалификации *k*. Коэффициент квалификации берется из таблицы коэффициентов, приведенной ниже.

Таблица 1 Коэффициенты квалификации

|  |  |
| --- | --- |
| Опыт работы | Коэффициент |
| До 2–х лет | 0,8 |
| 2 — 3 года | 1,0 |
| 3 — 5 лет | 1,1–1,2 |
| 5 — 7 лет | 1,3–1,4 |
| Больше 7 лет | 1,5–1,6 |

Согласно данным таблицы, необходимый коэффициент квалификации принимаем равным *k =* 1,2*.*

Для определения трудоемкости разработки приложения необходимо рассчитать следующие виды затрат:

* затраты труда на подготовку описания задачи;
* затраты труда на разработку блок–схемы приложения;
* затраты труда на программирование по блок–схеме;
* затраты труда на отладку кода;
* затраты труда на подготовку документации/описания.

Затраты труда на подготовку описания задачи рассчитываются по следующей формуле:

, (2)

где  — минимально возможная трудоемкость выполнения работы;

 — наиболее вероятная трудоемкость;

 — максимально возможная трудоемкость.

По формуле 2 получаем, что затраты составляют

 (чел/час).

Затраты труда на разработку блок–схемы алгоритма:

, (3)

где *B* — коэффициент увеличения затрат в зависимости от качества постановки задачи [1,2...5];

*k* — коэффициент квалификации берется из диапазона [75…85], (примем его равным 80):

 (чел/час).

Затраты труда на программирование по блок–схеме рассчитывается по формуле (коэффициент квалификации берется из диапазона [60…75], примем его равным 70):

 (4)

(чел/час).

Затраты труда на отладку кода (коэффициент квалификации берется из диапазона [40…50], примем его равным 40):

 (5)

(чел/час).

Затраты труда на подготовку документации:

(чел/час), (6)

где *t*пр — затраты труда на подготовку

*t*оф — время на оформление документов

,

(чел/час),



В итоге получаем:

(чел/час).

Тогда итоговая трудоемкость разработки данного программного обеспечения равна:

(чел/час),

(чел/час).

Это время приблизительно равняется одному месяцу при длине рабочего дня в 8 часов, что соответствует реальному времени разработки данного программного продукта.

При расчете затрат на разработку программного обеспечения необходимо разделить на составляющие: заработная плата (основная и дополнительная), отчисления на социальные нужды, эксплуатационные затраты (электроэнергия, техническое обслуживание и текущий ремонт), накладные расходы, материалы и комплектующие.

Основная заработная плата рассчитывается по формуле:

(руб.), (7)

где  – суммарные затраты труда, 148(чел/час);

*t*ср – среднее число рабочих дней в месяце: (365–107)/12 = 21,5 дней;

Тарифная ставка (ТС) представляет собой минимальный размер оплаты труда (МРОТ), увеличенный в зависимости от тарифного коэффициента *k*т, соответствующего данному разряду работ. Для 13–ого разряда работ, который соответствует работе программиста, тарифный коэффициент равен 3,04.

ТС = МРОТ.*k*т =480 ⋅ 3,04 = 1459,2 (руб/мес),

(руб.),

(мес.),

где *t*з - время, затраченное на создание программного обеспечения (в месяцах).

Надбавка за стаж составляет 5% от основной заработной платы

(руб.)

Премия 30 % от основной заработной платы за успешно выполненное задание в срок

(руб.).

Отчисление на социальное страхование составляют 35% от всей заработной платы:

(руб.)

Эксплуатационные затраты возникают при эксплуатации ЭВМ:

Стоимость электроэнергии:

, (8)

где *М* — мощность ЭВМ (0,06 кВт);

*к*з — коэффициент загрузки (0,7);

*С*кВт.ч. — стоимость 1 кВт час электроэнергии (0,165 руб.);

*F*эф — эффективный фонд, рассчитывается по формуле:

, (9)

где  — номинальное число рабочих дней в году (255);

 — планируемый процент времени на ремонт ЭВМ (2%);

тогда

(час.),

тогда стоимость электроэнергии составит:

(руб.).

Это стоимость электроэнергии за год, а за период разработки программного обеспечения стоимость электроэнергии составит:

, (10)

где  — суммарные затраты труда, 1273 (чел/час);

 — номинальное число рабочих дней в году (255);

 — стоимость электроэнергии за год.

Тогда стоимость электроэнергии за период разработки программного обеспечения составит:

(руб.).

Предположительная стоимость компьютера с необходимыми параметрами будет составлять 2000 (руб.).

Техническое обслуживание и ремонт составляют 2.5% от стоимости компьютера, тогда получаем:

(руб.).

Это стоимость и обслуживание технического ремонта за год, а за период разработки стоимость технического обслуживания и текущего ремонта составит:

(руб.),

где  — суммарные затраты труда, 1273 (чел.час);

 — номинальное число рабочих дней в году (255).

Накладные расходы составляют 10% от основной заработной платы:

(руб.).

Материалы и комплектующие составляют 1.5% от стоимости оборудования:

(руб.)

Таким образом, суммарные расходы на разработку программного обеспечения составили:

 (руб.) (11)

 (руб.)

Таблица 2 Смета затрат на разработку

|  |  |
| --- | --- |
| Составляющие себестоимости | Сумма, BYN |
| Зарплата | 653,54 |
| Социальное страхование | 308,79 |
| Эксплуатационные расходы | 0,52 |
| Накладные расходы | 63,35 |
| Материалы и комплектующие | 30 |
| Техническое обслуживание | 0,94 |
| **Итого** | 1057,14 |

Согласно данным таблицы 10 себестоимость разработанного программного продукта составляет 1057,14 (руб.).

# ЗАКЛЮЧЕНИЕ

Таким образом в ходе выполнения дипломного проекта было создано ПО(веб-приложения), были поставлены и выполнены следующие задачи:

1. Выбор языков программирования
2. Выбор программного обеспечения
3. Описать методы для нахождения пиков и выбрать подходящий
4. Написание приложения для распознавания, учёта, анализа, визуализации пиков
5. Реализация статистики для сравнения образцов раковых клеток разных возрастов

Взаимодействие между клиентской и серверной частью программного комплекса организовано по протоколу HTTP.

Приложение также записывает информацию в БД о пиках объектов раковых клеток разных возрастов.

# СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. А.А. Ищенко, М.А. Гольдштрах ЛЮМИНЕСЦЕНТНЫЙ АНАЛИЗ. Учебное пособие. М.: МИТХТ им. М.В.Ломоносова, 2009 — 36 с. ил.
2. JavaScript — Википедия [Электронный ресурс] / Википедия — свободная энциклопедя. – Режим доступа: https://ru.wikipedia.org/wiki/JavaScript. Дата доступа: 02.04.2022.
3. Python — Википедия [Электронный ресурс] / Википедия — свободная энциклопедия. – Режим доступа: https://ru.wikipedia.org/wiki/Python. Дата доступа: 04.09.2021.
4. React — Википедия [Электронный ресурс] / Википедия — свободная энциклопедия. – Режим доступа: <https://ru.wikipedia.org/wiki/React>. Дата доступа: 03.04.2022.
5. Flask — Docs-python [Электронный ресурс] / Docs-python — документация по языку Python3. – Режим доступа: https://docs-python.ru/packages/veb-frejmvork-flask-python. Дата доступа: 01.05.2022
6. MySQL — Википедия [Электронный ресурс] / Википедия — свободная энциклопедия. – Режим доступа: https://ru.wikipedia.org/wiki/MySQL. Дата доступа: 01.05.2022.
7. Pillow — Википедия [Электронный ресурс] / Википедия — свободная энциклопедия. – Режим доступа: https://ru.wikipedia.org/wiki/Python\_Imaging\_Library. Дата доступа: 05.10.2021.
8. OpenCV — Википедия [Электронный ресурс] / Википедия — свободная энциклопедия. – Режим доступа: https://ru.wikipedia.org/wiki/OpenCV. Дата доступа: 13.10.2021.
9. Tesseract — Википедия [Электронный ресурс] / Википедия — свободная энциклопедия. – Режим доступа: https://ru.wikipedia.org/wiki/Tesseract. Дата доступа: 15.10.2021.
10. Numpy — Википедия [Электронный ресурс] / Википедия — свободная энциклопедия. – Режим доступа: https://ru.wikipedia.org/wiki/numpy. Дата доступа: 16.10.2021.
11. Matplotlib — Википедия [Электронный ресурс] / Википедия — свободная энциклопедия. – Режим доступа: https://ru.wikipedia.org/wiki/Matplotlib. Дата доступа: 20.10.2021.
12. SciPy — Википедия [Электронный ресурс] / Википедия — свободная энциклопедия. – Режим https://ru.wikipedia.org/wiki/SciPy. Дата доступа: 22.10.2021.
13. Создайте новое приложение React — Ru.react.js [Электронный ресурс] / Ru.react.js — React. – Режим доступа: https://ru.react.js.org/docs/create-a-new-react-app.html. Дата доступа: 01.05.2022
14. Рендеринг элементов — Ru.react.js [Электронный ресурс] / Ru.react.js — React. – Режим доступа: https://ru.reactjs.org/docs/rendering-elements.html. Дата доступа: 01.05.2022
15. Рендеринг элементов — Metanit [Электронный ресурс] / Metanit — сайт о программировании. – Режим доступа: https://metanit.com/web/react/2.1.php. Дата доступа: 18.04.2022
16. Компоненты и пропсы — Ru.react.js [Электронный ресурс] / Ru.react.js — React. – Режим доступа: https://ru.reactjs.org/docs/components-and-props.html. Дата доступа: 01.05.2022
17. Состояние. Управление компонентами-классами — Metanit [Электронный ресурс] / Metanit — сайт о программировании. – Режим доступа: https://metanit.com/web/react/2.4.php. Дата доступа: 18.04.2022
18. Критерий Крускала-Уоллиса — Statistica [Электронный ресурс] / Metanit — Интеллектуальный портал. – Режим доступа: http://statistica.ru/local-portals/medicine/kriteriy-kraskela-uollisa/. Дата доступа: 13.05.2022

# ПРИЛОЖЕНИЕ

App.js

import React, { useState } from 'react';

import axios from 'axios';

import Modal from './components/Modal';

import Loader from './components/Loader';

import './sass/styles.sass';

function App() {

    const [firstAgeCells, setFirstAgeCells] = useState('');

    const [secondAgeCells, setSecondAgeCells] = useState('');

    const [firstFiles, setFirstFiles] = useState([]);

    const [secondFiles, setSecondFiles] = useState([]);

    const [images, setImages] = useState([]);

    const [statistics, setStatistics] = useState('');

    const [characteristics, setCharacteristics] = useState([]);

  const [isLoading, setIsLoading] = useState(false);

    const [validateText, setValidateText] = useState('');

    const [modalIsOpen, setModalIsOpen] = useState(false);

    const [prevFirstAgeCells, setPrevFirstAgeCells] = useState('');

    const [prevSecondAgeCells, setPrevSecondAgeCells] = useState('');

    const sendFiles = (e) => {

      if(firstAgeCells && secondAgeCells.length && firstFiles.length &&

secondFiles.length) {

e.preventDefault();

  let formData = new FormData();

  formData.append('firstAgeCells', firstAgeCells);

  formData.append('secondAgeCells', secondAgeCells);

Array.from(firstFiles).forEach(file => {

  formData.append('firstFiles', file);

    });

  Array.from(secondFiles).forEach(file => {

  formData.append('secondFiles', file);

});

setIsLoading(true);

    setValidateText('');

    axios.post('/files', formData, {headers: {"Content-Type":

"multipart/form-data"}})

.then(({data}) => {

  setImages(data['arrayImages']);

  setStatistics(data.statistics);

  setCharacteristics(data.characteristics);

  setIsLoading(false);

      setPrevFirstAgeCells(firstAgeCells);

    setPrevSecondAgeCells(secondAgeCells);

    setFirstAgeCells('');

    setSecondAgeCells('');

setFirstFiles([]);

    setSecondFiles([]);

  })

  } else {

  setValidateText('Заполните все инпуты');

  }

    };

    const getCharacteristicsFromDB = () => {

    setModalIsOpen(true);

      if(!characteristics.length) {

      setIsLoading(true);

  axios.get('/data')

    .then(({data}) => {

    setCharacteristics(data.characteristics);

setIsLoading(false);

})

    }

  };

    function sliceIntoChunks(arr, chunkSize) {

const res = [];

      for (let i = 0; i < arr.length; i += chunkSize) {

      const chunk = arr.slice(i, i + chunkSize);

  res.push(chunk);

    }

      return res;

    };

    function cutArray(arr) {

      const firstImages = arr.slice(0, arr.length / 2);

      const secondImages = arr.slice(arr.length / 2, arr.length);

      return { firstImages, secondImages };

    };

    const newImages = sliceIntoChunks(images, 5);

    const { firstImages, secondImages } = cutArray(newImages);

    return (

      <div className="app">

      <div className="control-ui">

    <div className="control-ui\_\_top">

    <div className="control-ui\_\_main-title">

Статистика раковых клеток

</div>

      <button

      className="btn btn--primary control-ui\_\_btn"

      onClick={getCharacteristicsFromDB}

      disabled={isLoading}

>

      Посмотреть информацию о пиках

    </button>

</div>

<div className="control-ui\_\_title">1-й объект(раковые клетки)</div>

  <input

type="text"

value={firstAgeCells}

onInput={(e) => setFirstAgeCells(e.target.value)}

  placeholder="Введите возраст раковых клеток"

className="input\_\_text"

/>

<div className="input-file\_\_wrapper">

  <input

type="file"

multiple="multiple"

onChange={e => setFirstFiles(e.target.files)}

id="firstFiles"

className="input-file\_\_real"

/>

<label htmlFor="firstFiles" className="input-file\_\_fake">

<span>Загрузите образцы раковых клеток</span>

</label>

</div>

<div className="control-ui\_\_title">2-й объект(раковые клетки)</div>

<input

type="text"

value={secondAgeCells}

onInput={(e) => setSecondAgeCells(e.target.value)}

placeholder="Введите возраст раковых клеток"

className="input\_\_text"

/>

<div className="input-file\_\_wrapper">

<input

type="file"

multiple="multiple"

onChange={e => setSecondFiles(e.target.files)}

id="secondFiles"

className="input-file\_\_real"

/>

<label htmlFor="secondFiles" className="input-file\_\_fake">

<span>Загрузите образцы раковых клеток</span>

  </label>

</div>

<div className="validate\_\_text">{validateText}</div>

<button

onClick={sendFiles}

className="btn btn--yellow"

disabled={isLoading}

>

Сделать статистику

</button>

</div>

{isLoading ?

<Loader /> :

!images.length ?

'' :

<>

<div className="statistics">

<div>p: {statistics.p}</div>

<div>{statistics.textStatistics}</div>

</div>

<div className="images-list">

{firstImages.map((images, index) =>

<div>

{images.map((img, i) =>

<>

<div>1-й объект(раковые клетки с возрастом

{prevFirstAgeCells})</div>

<div>{index + 1}-й образец</div>

<div>{i + 1}-я точка</div>

<img src={`data:image/jpeg;base64,${img}`}

alt="img\_data" />

</>

)}

</div>

)}

{secondImages.map((images, index) =>

  <div>

{images.map((img, i) =>

<>

<div>2-й объект(раковые клетки с возрастом

{prevSecondAgeCells})</div>

<div>{index + 1}-й образец</div>

<div>{i + 1}-я точка</div>

<img src={`data:image/jpeg;base64,${img}`}

alt="img\_data" />

</>

)}

</div>

)}

</div>

</>}

{modalIsOpen &&

<Modal

characteristics={characteristics}

setModalIsOpen={setModalIsOpen}

isLoading={isLoading}

/>}

</div>

);

}

export default App;

Loader.js

import React from 'react';

function Loader() {

    return (

        <div class="loader">

            <div></div>

            <div></div>

            <div></div>

        </div>

    );

}

export default Loader;

Modal.js

import React from 'react';

import Loader from './Loader';

function Modal({ characteristics, setModalIsOpen, isLoading }) {

return (

<div className="modal" onClick={() => setModalIsOpen(false)}>

<div className="modal\_\_dialog">

<div className="modal\_\_content" onClick={(e) =>

e.stopPropagation()}>

<span className="modal\_\_close" onClick={() =>

setModalIsOpen(false)}>X</span>

{isLoading ?

<Loader /> :

!characteristics.length ?

'База данных пустая' :

<>

<table>

<thead>

<tr>

<th></th>

<th>Волн. число</th>

<th>Интенсивность</th>

<th>Уник. значение</th>

</tr>

</thead>

<tbody>

{characteristics.length ?

characteristics.map(characteristic =>

<tr>

{characteristic['textInfo'] ?

<th>{characteristic['textInfo']}

</th> : <th></th>}

{characteristic['waveNumber'] ?

<td>{characteristic['waveNumber']}</td> :

<td></td>}

{characteristic['intensity'] ?

<td>{characteristic['intensity']}</td> :

<td></td>}

{characteristic['uniq'] !== null ?

<td>{characteristic['uniq'] ? 'Правда' :

'Ложь'}</td> : <td></td>}

</tr>

) : 'no characteristics'

}

</tbody>

</table>

</>}

</div>

</div>

</div>

)

}

export default Modal;

api.py

from flask import Flask

from flask import jsonify

from flask import request

import os

import shutil

import base64

from PIL import Image

import io

import pymysql

import datetime

from modules.main import main

app = Flask(\_\_name\_\_)

myDb = pymysql.connect(

    host='127.0.0.1',

    user='root',

    database='db',

    passwd='root'

)

myCursor = myDb.cursor()

def clearImgDirectory(path):

    dirSamples = path

    if(os.path.isdir(dirSamples)):

    shutil.rmtree(dirSamples)

os.mkdir(dirSamples)

def get\_characteristicsFromDB():

myCursor.execute("SELECT \* FROM characteristics;")

    rows = myCursor.fetchall()

    characteristics = []

    for row in rows:

characteristics.append({'idCharacteristic': row[0], 'textInfo': row[1],

'waveNumber': row[2], 'intensity': row[3], 'uniq': row[4]})

return characteristics

def saveFilesInDirectory(files, arrayToSave):

for file in files:

file.save(os.path.join('D:/RROJECT/DiplomProject/server/img/samples/',

file.filename))

arrayToSave.append(f'img/samples/{file.filename}')

@app.route('/files', methods=['POST'])

def data\_processing():

    firstPathFiles = []

    secondPathFiles = []

    firstFiles = request.files.getlist('firstFiles')

    secondFiles = request.files.getlist('secondFiles')

    firstAgeCells = request.values['firstAgeCells']

    secondAgeCells = request.values['secondAgeCells']

    clearImgDirectory('D:/RROJECT/DiplomProject/server/img/samples/')

clearImgDirectory('D:/RROJECT/DiplomProject/server/img/

visualization\_graphs/')

saveFilesInDirectory(firstFiles, firstPathFiles)

saveFilesInDirectory(secondFiles, secondPathFiles)

myCursor.execute(f"INSERT INTO characteristics(textInfo)

VALUES('{datetime.datetime.now()}');")

myDb.commit()

p, textStatistics = main(firstAgeCells, secondAgeCells, firstPathFiles,

secondPathFiles, myCursor, myDb)

    arrayImages = []

    for filename in os.listdir('img/visualization\_graphs/'):

pil\_img = Image.open(f"img/visualization\_graphs/{filename}",

mode='r')

byte\_arr = io.BytesIO()

pil\_img.save(byte\_arr, format='PNG')

encoded\_img = base64.encodebytes(byte\_arr.getvalue()).decode('ascii')

arrayImages.append(encoded\_img)

characteristics = get\_characteristicsFromDB()

return jsonify({'arrayImages': arrayImages, 'statistics': {'p': p,

'textStatistics': textStatistics}, 'characteristics': characteristics})

@app.route('/data')

def get\_data():

    characteristics = get\_characteristicsFromDB()

    return jsonify({'characteristics': characteristics})

if \_\_name\_\_ == "\_\_main\_\_":

    app.run(debug=True, port=5000);

main.py

from modules.functions import getResultsFromImg

from scipy.stats import kruskal

def imgsProcessing(imgArray, ageCells, myCursor, myDb):

    waveNumberArray = []

myCursor.execute(f"INSERT INTO characteristics(textInfo) VALUES('Раковые

клетки с возрастом {ageCells}');")

    myDb.commit()

    for i, img in enumerate(imgArray):

getResultsFromImg(img, i + 1, waveNumberArray, myCursor, myDb)

return waveNumberArray

def main(firstAgeCells, secondAgeCells, firstPathFiles, secondPathFiles,

myCursor, myDb):

firstWaveNumberArray = imgsProcessing(firstPathFiles, firstAgeCells,

myCursor, myDb)

secondWaveNumberArray = imgsProcessing(secondPathFiles, secondAgeCells,

myCursor, myDb)

    \_, p = kruskal(firstWaveNumberArray, secondWaveNumberArray)

    textStatistics = ''

    alpha = 0.05

    if p > alpha:

    print('Примеры распределений равны')

textStatistics = 'Примеры распределений равны'

else:

print('Примеры распределений не равны')

textStatistics = 'Примеры распределений не равны'

return p, textStatistics

functions.py

from PIL import Image

import cv2

import pytesseract

import numpy as np

import matplotlib.pyplot as plt

from scipy.signal import find\_peaks

from itertools import groupby

import uuid

import re

# ------------ФУНКЦИИ------------

def findAxisConstValue(array, coordParam, black\_pixels, im, col\_white):

    black\_pixels\_X\_with\_count = [{coordParam: i, 'count': array.count((i))} for

i in array]

    black\_pixels\_X\_with\_count\_sort\_by\_count = sorted(black\_pixels\_X\_with\_count,

key=lambda x: x['count'], reverse=True)

withoutDuplicates = [el for el, \_ in groupby(black\_pixels\_X\_with\_count\_sort\_by\_count)]

    axisConstValue = withoutDuplicates[0][coordParam]

    if coordParam == 'x' and withoutDuplicates[0]['count'] –

withoutDuplicates[1]['count'] <= 100:

      vertLineX = withoutDuplicates[1]['x']

vertLineCoords = []

for pixel in black\_pixels:

if vertLineX == pixel['x']:

vertLineCoords.append({'x': pixel['x'], 'y': pixel['y']})

print('vertLineCoords: ', vertLineCoords)

for x in range(im.size[0]):

for y in range(im.size[1]):

if x == vertLineX and (y >= vertLineCoords[0]['y'] or y <=

vertLineCoords[len(vertLineCoords) - 1]['y']):

im.putpixel((x, y), col\_white)

return axisConstValue

def removeUnnecessaryPixels(array, coordParam):

tempArray = []

for i in range(len(array)):

    if array[i][coordParam] == array[i - 1][coordParam] + 1 or i == 0:

      tempArray.append({'x': array[i]['x'], 'y': array[i]['y']})

else:

    break

return tempArray

def thresholdStroke(axis, im, axis\_x\_const\_y, axisXCoords, col\_black,

axis\_y\_const\_x, axisYCoords):

    threshols = []

    if axis == 'x':

      for x in range(im.size[0]):

k = 0

for y in range(im.size[1]):

if y >= axis\_x\_const\_y and x >= axisXCoords[0]['x'] and x <=

axisXCoords[len(axisXCoords) - 1]['x'] and

im.getpixel((x, axis\_x\_const\_y + k + 1)) == col\_black:

k = k + 1

if y == im.size[1] - 1 and k != 0:

threshols.append(k)

threshols = list(set(threshols))

return {'smallThresholsX': threshols[0], 'bigThresholsX': threshols[1]}

else:

for y in range(im.size[1]):

k = 0

for x in reversed(range(im.size[0])):

if x <= axis\_y\_const\_x and y >= axisYCoords[0]['y'] and y <=

axisYCoords[len(axisYCoords) - 1]['y'] and

im.getpixel((axis\_y\_const\_x - k - 1, y)) == col\_black:

k = k + 1

if x == 0 and k != 0:

threshols.append(k)

threshols = list(set(threshols))

return {'smallThresholsY': threshols[0], 'bigThresholsY': threshols[1]}

def findStrokes(array, axis, threshold, im, col\_black):

    strokeArray = []

    if axis == 'x':

      for y in range(im.size[1]):

      for x in range(im.size[0]):

if im.getpixel((x, y)) == col\_black:

elem = list(filter(lambda el: el['x'] == x and el['y'] == y,

array))

if len(elem) != 0 and im.getpixel((x, y + threshold)) ==

col\_black:

strokeArray.append({'x': x, 'y': y})

return [strokeArray[0], strokeArray[len(strokeArray) - 2],

strokeArray[len(strokeArray) - 1]]

    else:

for y in range(im.size[1]):

for x in range(im.size[0]):

if im.getpixel((x, y)) == col\_black:

elem = list(filter(lambda el: el['x'] == x and el['y'] == y,

array))

if len(elem) != 0 and im.getpixel((x - threshold, y)) ==

col\_black:

strokeArray.append({'x': x, 'y': y})

return [strokeArray[len(strokeArray) - 1], strokeArray[1],

strokeArray[0]]

def recognizeValue(img):

im\_tesseract = cv2.resize(img, None, fx=9, fy=9)

pytesseract.pytesseract.tesseract\_cmd = r"C:\Program Files\Tesseract-

OCR\tesseract.exe"

config = r'--oem 3 --psm 6 outputbase digits'

string = pytesseract.image\_to\_string(im\_tesseract, config=config)

value = int(string.split()[0]) if len(string) > 4 else int(string)

return value

def findPixelsForPoint(color, cropIm):

    pixelsArray = []

    arrayX = []

    arrayY = []

    for x in range(cropIm.size[0]):

k = 0

for y in range(cropIm.size[1]):

if k == 1:

k == 0

break

if cropIm.getpixel((x, y)) == color:

k = k + 1

pixelsArray.append({'x': x, 'y': y})

arrayX.append(x)

arrayY.append(y)

listX = list(range(pixelsArray[0]['x'], pixelsArray[len(pixelsArray) –

1]['x'] + 1))

    interpArrayY = np.interp(listX, arrayX, arrayY)

    newArray = []

    for i, y in enumerate(interpArrayY):

  newArray.append({'x': listX[i], 'y': y})

    return newArray

def getPeaks(pointArray, img\_1px\_equel\_func\_valueX, firstBigStrokeY,

beginning\_of\_coord, cropIm, img\_1px\_equel\_func\_valueY, minValueY, imgPath,

pointNumberImg):

    arrayX = []

    arrayY = []

for pixel in pointArray:

arrayX.append(pixel['x'])

arrayY.append(pixel['y'])

    intensityArray = []

waveNumberArray = []

for i, x in enumerate(arrayX):

waveNumberArray.append(x \* img\_1px\_equel\_func\_valueX)

for i, y in enumerate(arrayY):

if firstBigStrokeY['y'] == beginning\_of\_coord['y']:

intensityArray.append((cropIm.size[1] - y) \* i

mg\_1px\_equel\_func\_valueY + minValueY)

      else:

diffStrokeY = beginning\_of\_coord['y'] - firstBigStrokeY['y']

if y < firstBigStrokeY['y']:

intensityArray.append((firstBigStrokeY['y'] - y) \*

img\_1px\_equel\_func\_valueY + minValueY)

else:

intensityArray.append(((cropIm.size[1] - y) \* ((minValueY - 0) /

diffStrokeY)))

    intensityArray = np.array(intensityArray)

    waveNumberArray = np.array(waveNumberArray)

    peaks = find\_peaks(intensityArray, height=1, threshold=1, distance=1)

    height = peaks[1]['peak\_heights']

    peak\_pos = waveNumberArray[peaks[0]]

    plt.plot(waveNumberArray, intensityArray)

    plt.scatter(peak\_pos, height, color='r', s=15, marker='D', label='Peak')

    plt.legend()

    plt.grid()

    plt.xlabel('Wave number', fontsize=12)

    plt.ylabel('Intensity', fontsize=12)

    newStr = imgPath.replace("samples", "visualization\_graphs").replace("bmp",

"jpg").replace(".", f'({pointNumberImg}).')

    plt.savefig(f'{newStr}')

    plt.clf()

    peaksArray = []

    for i in peaks[0]:

peaksArray.append({'waveNumber': waveNumberArray[i], 'intensity':

intensityArray[i]})

pointNumberImg = pointNumberImg + 1

return peaksArray, pointNumberImg

def setUniqPeaks(currPeaksArray, peaksArrays):

    for peaksArray in peaksArrays:

for currPeak in currPeaksArray:

for peak in peaksArray:

if currPeak['waveNumber'] == peak['waveNumber']:

currPeak['noUniq'] = True

peak['noUniq'] = True

return currPeaksArray

def writeToDB(array, pointNumber, myCursor, myDb):

myCursor.execute(f"INSERT INTO characteristics(textInfo) VALUES('Точка №

{pointNumber}');")

    myDb.commit()

    for i in range(len(array)):

myCursor.execute(f"INSERT INTO characteristics(waveNumber, intensity,

uniq) VALUES({array[i]['waveNumber']}, {array[i]['intensity']}, {False

if 'noUniq' in array[i] else True})")

myDb.commit()

pointNumber = pointNumber + 1

return pointNumber

def writeToWaveNumberArray(waveNumberArray, pointArrayPeaks):

for peak in pointArrayPeaks:

waveNumberArray.append(peak['waveNumber'])

# ------------ФУНКЦИИ------------

def getResultsFromImg(imgPath, sampleNumber, waveNumberArray, myCursor, myDb):

    im = Image.open(imgPath)

    im = im.convert('RGB')

    col\_black = (0, 0, 0)

    col\_white = (255, 255, 255)

    # массивы чёрных пикселей x, y, xy

    black\_pixels\_X = []

    black\_pixels\_Y = []

    black\_pixels = []

    for y in range(im.size[1]):

for x in range(im.size[0]):

if im.getpixel((x, y)) == col\_black:

black\_pixels\_X.append(x)

black\_pixels\_Y.append(y)

black\_pixels.append({'x': x, 'y': y})

# нахождения постоянного значения x оY и y oX и удаление чёрной вертикальной линии

axis\_y\_const\_x = findAxisConstValue(black\_pixels\_X, 'x', black\_pixels, im,

col\_white)

axis\_x\_const\_y = findAxisConstValue(black\_pixels\_Y, 'y', black\_pixels, im,

col\_white)

# нахождение всех чёрных пикселей в столбце в котором находится ось Y

axisYCoords = []

for pixel in black\_pixels:

if axis\_y\_const\_x == pixel['x']:

axisYCoords.append({'x': pixel['x'], 'y': pixel['y']})

    # нахождение всех чёрных пикселей в столбце в котором находится ось X

axisXCoords = []

for pixel in black\_pixels:

if axis\_x\_const\_y == pixel['y']:

axisXCoords.append({'x': pixel['x'], 'y': pixel['y']})

# удаления лишних пикселей в столбце и строке в котором есть координаты оY

и oX

    axisYCoords = removeUnnecessaryPixels(axisYCoords, 'y')

    axisXCoords = removeUnnecessaryPixels(axisXCoords, 'x')

    print('axisYCoords: ', axisYCoords)

    print('axisXCoords: ', axisXCoords)

    # находим начало координат(пересечение oY и oX)

    beginning\_of\_coord = 0

    for axisYCoord in axisYCoords:

for axisXcoord in axisXCoords:

if axisYCoord['x'] == axisXcoord['x'] and axisYCoord['y'] ==

axisXcoord['y']:

beginning\_of\_coord = axisXcoord

break

print('beginning\_of\_coord: ', beginning\_of\_coord)

# нахождение порога у мал и бол штрихов на оX и oY

thresholdsX = thresholdStroke('x', im, axis\_x\_const\_y, axisXCoords,

col\_black, axis\_y\_const\_x, axisYCoords)

thresholdsY = thresholdStroke('y', im, axis\_x\_const\_y, axisXCoords,

col\_black, axis\_y\_const\_x, axisYCoords)

print('thresholdX: ', thresholdsX)

print('thresholdY: ', thresholdsY)

# нахождение штрихов по порогу на оX и oY

firstBigStrokeX, secondLastBigStrokeX, lastBigStrokeX =

findStrokes(axisXCoords, 'x', thresholdsX['bigThresholsX'], im, col\_black)

firstBigStrokeY, secondLastBigStrokeY, lastBigStrokeY =

findStrokes(axisYCoords, 'y', thresholdsY['bigThresholsY'], im, col\_black)

    \_, secondSmallStrokeX, \_ = findStrokes(axisXCoords, 'x',

thresholdsX['smallThresholsX'], im, col\_black)

print('firstBigStrokeX: ', firstBigStrokeX, 'lastBigStrokeX: ',

lastBigStrokeX)

print('firstBigStrokeY: ', firstBigStrokeY, 'lastBigStrokeY: ',

lastBigStrokeY)

print('secondSmallStrokeX: ', secondSmallStrokeX)

# распознование максимального значения интенсивности на oY и волнового

числа на oX с картинки

diffBigStrokeY = int(abs((lastBigStrokeY['y'] - secondLastBigStrokeY['y'])

/ 2))

im\_cropMaxY = im.crop((lastBigStrokeY['x'] - thresholdsX['bigThresholsX'] –

40, lastBigStrokeY['y'] - diffBigStrokeY, lastBigStrokeY['x'] –

thresholdsY['bigThresholsY'], lastBigStrokeY['y'] + diffBigStrokeY))

    im\_cropMaxY.save('img/crops/cropMaxY.png')

    diffBigStrokeX = int(abs((lastBigStrokeX['x'] - secondLastBigStrokeX['x'])

/ 2))

im\_cropMaxX = im.crop((lastBigStrokeX['x'] - diffBigStrokeX,

lastBigStrokeX['y'] + thresholdsX['bigThresholsX'] + 1,

lastBigStrokeX['x'] + diffBigStrokeX, lastBigStrokeX['y'] + 25))

im\_cropMaxX.save('img/crops/cropMaxX.png')

maxValueY = recognizeValue(cv2.imread("img/crops/cropMaxY.png",

cv2.IMREAD\_GRAYSCALE))

maxValueX = recognizeValue(cv2.imread("img/crops/cropMaxX.png",

cv2.IMREAD\_GRAYSCALE))

    print('maxValueY: ', maxValueY)

    print('maxValueX: ', maxValueX)

    # распознование минимального значения интенсивности на oY и волного числа

oX с картинки

    im\_cropMinY = im.crop((firstBigStrokeY['x'] - thresholdsY['bigThresholsY']

- 40, firstBigStrokeY['y'] - diffBigStrokeY,

firstBigStrokeY['x'] - thresholdsY['bigThresholsY'], firstBigStrokeY['y'] +

diffBigStrokeY))

im\_cropMinY.save('img/crops/cropMinY.png')

im\_cropMinX = im.crop((firstBigStrokeX['x'] - diffBigStrokeX,

firstBigStrokeX['y'] + thresholdsX['bigThresholsX'] + 1,

firstBigStrokeX['x'] + diffBigStrokeX, firstBigStrokeX['y'] + 25))

im\_cropMinX.save('img/crops/cropMinX.png')

minValueY = recognizeValue(cv2.imread("img/crops/cropMinY.png",

cv2.IMREAD\_GRAYSCALE))

minValueX = recognizeValue(cv2.imread("img/crops/cropMinX.png",

cv2.IMREAD\_GRAYSCALE))

    print('minValueY: ', minValueY)

    print('minValueX: ', minValueX)

# сколько 1px занимает значений интенсивности и волнового числа

img\_1px\_equel\_func\_valueX = (maxValueX - minValueX) / (lastBigStrokeX['x']

- beginning\_of\_coord['x'])

img\_1px\_equel\_func\_valueY = (maxValueY - minValueY) / (firstBigStrokeY['y']

- lastBigStrokeY['y'])

print('img\_1px\_equel\_func\_valueX: ', img\_1px\_equel\_func\_valueX)

print('img\_1px\_equel\_func\_valueY: ', img\_1px\_equel\_func\_valueY)

# вырезаем графики

cropIm = im.crop((lastBigStrokeY['x'] + 1, 0, secondSmallStrokeX['x'],

secondSmallStrokeX['y']))

cropIm.save("img/crops/cropIm.bmp")

# нахождение пикселей для каждой точки образца

firstPointArray = findPixelsForPoint((0, 0, 0), cropIm)

secondPointArray = findPixelsForPoint((255, 0, 0), cropIm)

thirdPointArray = findPixelsForPoint((0, 0, 255), cropIm)

fourthPointArray = findPixelsForPoint((0, 128, 128), cropIm)

fifthPointArray = findPixelsForPoint((255, 0, 255), cropIm)

print('firstPointArray: ', firstPointArray)

print('secondPointArray: ', secondPointArray)

print('thirdPointArray: ', thirdPointArray)

print('fourthPointArray: ', fourthPointArray)

print('fifthPointArray: ', fifthPointArray)

# получаем и визуализируем пики

pointNumberImg = 1

firstPointArrayPeaks, pointNumberImg = getPeaks(firstPointArray,

img\_1px\_equel\_func\_valueX, firstBigStrokeY, beginning\_of\_coord, cropIm,

img\_1px\_equel\_func\_valueY, minValueY, imgPath, pointNumberImg)

    secondPointArrayPeaks, pointNumberImg = getPeaks(secondPointArray,

img\_1px\_equel\_func\_valueX, firstBigStrokeY, beginning\_of\_coord, cropIm,

img\_1px\_equel\_func\_valueY, minValueY, imgPath, pointNumberImg)

thirdPointArrayPeaks, pointNumberImg = getPeaks(thirdPointArray,

img\_1px\_equel\_func\_valueX, firstBigStrokeY, beginning\_of\_coord, cropIm, img\_1px\_equel\_func\_valueY, minValueY, imgPath, pointNumberImg)

fourthPointArrayPeaks, pointNumberImg = getPeaks(fourthPointArray,

img\_1px\_equel\_func\_valueX, firstBigStrokeY, beginning\_of\_coord, cropIm, img\_1px\_equel\_func\_valueY, minValueY, imgPath, pointNumberImg)

fifthPointArrayPeaks, pointNumberImg = getPeaks(fifthPointArray, img\_1px\_equel\_func\_valueX, firstBigStrokeY, beginning\_of\_coord, cropIm, img\_1px\_equel\_func\_valueY, minValueY, imgPath, pointNumberImg)

    print('firstPointArrayPeaks: ', firstPointArrayPeaks)

    print('secondPointArrayPeaks: ', secondPointArrayPeaks)

    print('thirdPointArrayPeaks: ', thirdPointArrayPeaks)

    print('fourthPointArrayPeaks: ', fourthPointArrayPeaks)

    print('fifthPointArrayPeaks: ', fifthPointArrayPeaks)

# нахождение уникальных пиков

firstPointArrayPeaks = setUniqPeaks(firstPointArrayPeaks,

[secondPointArrayPeaks, thirdPointArrayPeaks, fourthPointArrayPeaks,

fifthPointArrayPeaks])

secondPointArrayPeaks = setUniqPeaks(secondPointArrayPeaks,

[firstPointArrayPeaks, thirdPointArrayPeaks, fourthPointArrayPeaks,

fifthPointArrayPeaks])

thirdPointArrayPeaks = setUniqPeaks(thirdPointArrayPeaks,

[firstPointArrayPeaks, secondPointArrayPeaks, fourthPointArrayPeaks,

fifthPointArrayPeaks])

fourthPointArrayPeaks = setUniqPeaks(fourthPointArrayPeaks,

[firstPointArrayPeaks, secondPointArrayPeaks, thirdPointArrayPeaks,

fifthPointArrayPeaks])

fifthPointArrayPeaks = setUniqPeaks(fifthPointArrayPeaks,

[firstPointArrayPeaks, secondPointArrayPeaks, thirdPointArrayPeaks,

fourthPointArrayPeaks])

    print('firstPointArrayPeaks: ', firstPointArrayPeaks)

    print('secondPointArrayPeaks: ', secondPointArrayPeaks)

    print('thirdPointArrayPeaks: ', thirdPointArrayPeaks)

    print('fourthPointArrayPeaks: ', fourthPointArrayPeaks)

    print('fifthPointArrayPeaks: ', fifthPointArrayPeaks)

    writeToWaveNumberArray(waveNumberArray, firstPointArrayPeaks)

    writeToWaveNumberArray(waveNumberArray, secondPointArrayPeaks)

    writeToWaveNumberArray(waveNumberArray, thirdPointArrayPeaks)

    writeToWaveNumberArray(waveNumberArray, fourthPointArrayPeaks)

    writeToWaveNumberArray(waveNumberArray, fifthPointArrayPeaks)

    pointNumber = 1

    myCursor.execute(f"INSERT INTO characteristics(textInfo) VALUES('Образец №

{sampleNumber}');")

    myDb.commit()

    # запись значений пиков в Microsoft Excel

    pointNumber = writeToDB(firstPointArrayPeaks, pointNumber, myCursor, myDb)

    pointNumber = writeToDB(secondPointArrayPeaks, pointNumber, myCursor, myDb)

    pointNumber = writeToDB(thirdPointArrayPeaks, pointNumber, myCursor, myDb)

    pointNumber = writeToDB(fourthPointArrayPeaks, pointNumber, myCursor, myDb)

    pointNumber = writeToDB(fifthPointArrayPeaks, pointNumber, myCursor, myDb)