МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра програмної інженерії

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему: **«РОЗРОБКА СИСТЕМИ АНАЛІЗУ МЕДИЧНИХ ЗОБРАЖЕНЬ»**

Виконав: студент 4 курсу, групи 6.1219-1пі
спеціальності 121 інженерія програмного забезпечення
(шифр і назва спеціальності)
освітньої програми програмна інженерія
(назва освітньої програми)
А.С. Нехороший
(ініціали та прізвище)

Керівник доцент кафедри програмної інженерії, доцент, к.ф.-м.н. Кудін О.В.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Рецензент завідувач кафедри фундаментальної та прикладної математики, професор, д.т.н. Гребенюк С.М.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя – 2023

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математичний

Кафедра програмної інженерії

Рівень вищої освіти бакалавр

Спеціальність 121 інженерія програмного забезпечення
(шифр і назва)

Освітня програма програмна інженерія

ЗАТВЕРДЖУЮ
Завідувач кафедри програмної інженерії, к.ф.-м.н., доцент

_____ Лісняк А.О.
(підпис)

“ 07 ” 02 2023р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Нехорошому Андрію Сергійовичу
(прізвище, ім’я та по-батькові)

1. Тема роботи Розробка системи аналізу медичних зображень

керівник роботи Кудін Олексій Володимирович, к.ф.-м.н, доцент
(прізвище, ім’я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 26 » січня 2023 року № 102-с

2. Строк подання студентом роботи 07.06.2023 р.

3. Вихідні дані до роботи 1. Постановка задачі.
2. Перелік літератури.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Постановка задачі, аналіз предметної області.
2. Проєктування.
3. Реалізація та тестування.

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)
презентація за темою докладу

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 07.02.2023 р.

К А Л Е Н Д А Р Н И Й П Л А Н

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи.	08.02.2023	
2.	Збір вихідних даних.	17.02.2023	
3.	Обробка методичних та теоретичних	16.03.2023	
	джерел.		
4.	Розробка першого та другого розділу.	17.04.2023	
5.	Розробка третього розділу.	15.05.2023	
6.	Оформлення та нормоконтроль	01.06.2023	
	кваліфікаційної роботи бакалавра.		
7.	Захист кваліфікаційної роботи.	21.06.2023	

Студент _____ А.С. Нехороший
(підпис) (ініціали та прізвище)

Керівник роботи _____ О.В. Кудін
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ А.В. Столярова
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Кваліфікаційна робота бакалавра «Розробка системи аналізу медичних зображень»: 61 с., 14 рис., 1 табл., 19 джерел, 1 додаток.

МЕДИЧНЕ ЗОБРАЖЕННЯ, МЕДИЧНІ ЗНІМКИ, ОБРОБКА ЗОБРАЖЕНЬ, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, PYTHON.

Об’єкт дослідження – методи роботи з медичними знімками, їх обробка.

Мета роботи: розробити програмне забезпечення для роботи з медичними знімками.

Метод дослідження – аналітичний.

В контексті виконання роботи мною був отриманий результат у вигляді мобільного додатку, а саме середовище для простого аналізу медичних знімків. Також було проведено дослідження методів для подальшого вдосконалення програмного забезпечення.

Розроблений додаток може бути задіяний користувачем для виявлення патологій на медичних знімках.

SUMMARY

Bachelor's qualifying paper "Development of a System for Medical Images Analysis": 61 pages, 14 figures, 1 table, 19 references, 1 supplement.

MEDICAL IMAGE, MEDICAL IMAGING, IMAGE PROCESSING, SOFTWARE, PYTHON.

The object of the study is methods for working with medical images and their processing.

The aim of the study is to develop software for medical images processing.

The method of research is analytical.

In the context of this work, we have achieved a result in the form of a mobile application, specifically an environment for simple analysis of medical images. Research on methods for further improvement of the software has also been conducted.

The developed application can be utilized by users to detect pathologies in medical images.

ЗМІСТ

ВСТУП

За останні роки розвиток сучасних технологій зображувальної медицини спричинив значний приріст обсягу медичних зображень, що зберігаються переважно у форматі DICOM (Digital Imaging and Communications in Medicine). Цей формат дозволяє зберігати, передавати та обмінюватися медичними зображеннями з високою якістю і стандартизованим підходом. Застосування зображень у форматі DICOM стало ключовим елементом у багатьох аспектах медичної практики, включаючи діагностику, лікування та дослідження.

Метою даної кваліфікаційної роботи є розробка програмного забезпечення, яке сприятиме ефективній роботі з медичними зображеннями у форматі DICOM. Основний акцент роботи полягатиме на створенні інструментів та функціоналу, які допоможуть лікарям та медичному персоналу здійснювати швидку, точну та зручну обробку, перегляд, аналіз та архівацію медичних знімків у форматі DICOM.

Програмне забезпечення, розроблене у рамках даної кваліфікаційної роботи, буде включати такі функції:

- завантаження медичних зображень у форматі DICOM та їх зберігання у відповідній базі даних;
- візуалізація медичних зображень з можливістю масштабування, ротації та розгортання;
- розробка інструментів для вимірювання розмірів, площі, об'єму та інших параметрів на зображеннях;
- підтримка роботи з різними типами медичних знімків, включаючи рентгенограми, КТ, МРТ та інші.

Результати даної роботи матимуть практичне застосування в медичних закладах, сприяючи поліпшенню якості надання медичних послуг, зменшенню часу обробки зображень та спрощенню робочого процесу медичного персоналу. Використання програмного забезпечення, розробленого у рамках

даної дипломної роботи, дозволить покращити доступність та ефективність аналізу медичних зображень у форматі DICOM, сприяючи підвищенню рівня медичної діагностики та лікування.

1 АНАЛІЗ ВИМОГ

1.1 Актуальність розробки ПЗ для аналізу медичних зображень

Розробка програмного забезпечення для аналізу медичних знімків є актуальною та важливою задачею в сучасному медичному середовищі. Зростання обсягу медичних даних, швидкі темпи технологічного розвитку та потреба у швидкій та точній діагностиці ставлять перед медичною спільнотою складні завдання. У такому контексті програмне забезпечення, яке може ефективно обробляти та аналізувати медичні зображення, є великим кроком у напрямку поліпшення якості медичної діагностики та лікування.

Однією з ключових переваг програмного забезпечення для аналізу медичних знімків є можливість автоматизації процесу обробки та інтерпретації зображень. Традиційно, медичний персонал проводить ручний аналіз зображень, що може бути часо- та працезатратним завданням. Застосування програмного забезпечення дозволяє автоматизувати деякі етапи аналізу, забезпечуючи швидкість та консистентність результатів. Наприклад, алгоритми обробки зображень можуть бути використані для виявлення патологій, сегментації структур або вимірювання параметрів зображень [1].

Крім того, програмне забезпечення для аналізу медичних знімків може допомогти у покращенні взаємодії між медичним персоналом та пацієнтами. Завдяки цифровим зображенням та системам зберігання даних, медичні знімки можуть бути легкодоступні для спільного перегляду та консультацій, навіть на відстані. Це забезпечує більш ефективну комунікацію між медичними фахівцями, сприяє швидшій діагностиці та прийняттю рішень [1].

Важливим аспектом розробки програмного забезпечення для аналізу медичних знімків є забезпечення конфіденційності та безпеки медичних даних. У зв'язку зі зростаючими загрозами кібербезпеки, необхідно розробляти рішення, які забезпечують захист приватності пацієнтів та безпеку

медичних записів. Розробники програмного забезпечення повинні дотримуватись відповідних стандартів та протоколів для забезпечення безпеки даних, що включає шифрування, автентифікацію та контроль доступу [1].

Важливо враховувати перспективи майбутнього розвитку програмного забезпечення для аналізу медичних знімків. Застосування штучного інтелекту, машинного навчання та глибокого навчання в цій області відкриває нові можливості для автоматизації аналізу зображень та покращення точності діагностики. Додаткові дослідження та розробка нових алгоритмів можуть привести до подальшого покращення продуктивності та ефективності програмного забезпечення для аналізу медичних знімків [1].

Отже, розробка програмного забезпечення для аналізу медичних знімків має великий потенціал у покращенні медичної діагностики, ефективності та доступності медичних послуг. Застосування автоматизованих методів аналізу та обробки зображень, спільної роботи між медичним персоналом та використання сучасних технологій допомагає забезпечити швидку, точну та ефективну діагностику, а також поліпшити якість лікування. Розвиток програмного забезпечення для аналізу медичних знімків є важливим напрямом досліджень, що сприяє прогресу в медицині та допомагає зберегти життя [1].

1.2 Формування завдань

Відповідно до мети, задача кваліфікаційної роботи – створення програмного забезпечення для роботи з медичними знімками, його подальше вдосконалення. Основне призначення ПЗ полягає у створенні інструменту для швидкої роботи з медичними знімками формату DICOM. Користувач зможе відкрити медичний знімок та швидко провести аналіз [2].

Таким чином планується полегшити роботу, перейти від традиційного аналізу знімку під світлом лампи, до сучасного – цифрового [2].

Опираючись на інформацію вище, для вирішення поставленої задачі були застосовані такі кроки:

а) вибір інструментів для реалізації розробки продукту;

б) формування функціональних вимог;

в) визначення логіки взаємодії між системою та користувачем [2].

1.3 Загальні положення

Зображення можна уявити собі як картину, де кожній точці на площині (представленій координатами x і y) відповідає певне значення, яке називається інтенсивністю або яскравістю зображення в цій точці. Якщо ці значення обмежені і представлені дискретними числами, то ми говоримо про цифрове зображення [2].

Для обробки цифрових зображень використовують комп'ютери. Це означає, що за допомогою програмного забезпечення та обчислювальних операцій ми можемо виконувати різні маніпуляції з цими зображеннями: змінювати їх, виділяти особливості та отримувати бажані результати. Така обробка зображень дозволяє нам аналізувати та розуміти їх зміст, а також виявляти корисну інформацію, яка може бути недоступна на перший погляд [3].

1.4 Типи цифрових зображень

Цифрове зображення складається з пікселів – найменших елементів зображення, кожен з яких розташований у конкретному місці та приймає певне значення. Залежно від способу опису зображення бувають наступних типів: дворівневе, напівтонове, кольорове, безперервно тонове, дискретно-тонове, а

також зображення, подібне до мультфільмів [2].

Дворівневе (монохроматичне) зображення має лише два можливих значення для кожного пікселя: 1 і 0, що відповідають чорному та білому кольорам відповідно. Кожен піксель представляється одним бітом. Напівтонове зображення дозволяє кожному пікселю мати значення від 0 до 2^n-1, де n-кількість градацій сірого кольору.

Кольорове зображення може бути представлене кількома методами, але в кожному використовуються три або чотири параметри. Зазвичай кожен кольоровий піксель складається з трьох байтів (по одному байту на кожен параметр) [4].

Безперервно-тонове зображення може мати безліч схожих кольорів або півтонів. В таких зображеннях можуть бути області, де кольори здаються безперервно змінюватися. Піксель в цьому типі зображення може представлятися як більше число (у напівтоновому випадку) або трьома компонентами (у випадку кольорового зображення). Цей тип зображення зазвичай використовується для природних зображень, таких як фотографії природи або рентгенограми [5].

Дискретно-тонове зображення (синтетичне) – це зазвичай зображення виходить штучним шляхом. У ньому може бути все кілька кольорів або багато кольорів, але в ньому немає шумів чи плям природного зображення. Приклади: фото штучних об'єктів, скан сторінки тексту, карти, томограми. Об'єкти на фоні дискретно-тонових зображеннях різко контрастують [5].

Зображення подібне до мультфільмів використовується для анімаційних фільмів або комп'ютерних ігор. Воно може мати гладкі контури, насичені кольори та відсутність деталей, що характерно для реальних об'єктів [6].

Кожен тип зображень має свої особливості і використовується в різних сферах, включаючи фотографію, медицину, комп'ютерну графіку, телекомунікації та інші. Коректне представлення зображень дозволяє зберігати, передавати та обробляти візуальну інформацію з високою якістю та точністю [6].

1.5 Аналіз проблеми обробки медичних зображень

У сучасній медицині все більше залежить від використання медичних зображень для діагностики та лікування пацієнтів. Ці зображення містять значний обсяг інформації про стан пацієнта та його захворювання. Медичні зображення надають лікарям необхідні дані для аналізу та прийняття рішень [1].

Проте, іноді захворювання не можна однозначно визначити лише за допомогою зображення. Це пов'язано з тим, що вимірювання медичних параметрів можуть відхилятися від реальних значень через похибки вимірювання, наявність перешкод або неправильне налаштування обладнання [1].

Крім того, медичні знімки можуть бути піддані впливу навколишнього середовища, що може негативно вплинути на їх якість та призвести до появи артефактів. Якість медичної інформації безпосередньо впливає на точність діагностики та подальше лікування. Тому проводиться вторинна обробка медичних даних з метою покращення їх якості [3].

Обробка медичних зображень є технологією, яка дозволяє виявляти приховані елементи на зображенні, які практично невидимі без спеціальних маніпуляцій. Головна мета цієї процедури – зміна зображення таким чином, щоб не спотворювати початкові дані, але водночас виділяти тонкі структури органів під час різних типів досліджень. Це дозволяє забезпечити високу якість діагностики та точність визначення патологій [3].

1.6 Особливості обробки медичних зображень

Цифрова обробка медичних зображень зацікавлює дослідників та медичні установи через її потенціал у двох важливих областях застосування. По-перше, це підвищення якості медичних знімків з метою поліпшення їх

візуального сприйняття людиною. Завдяки цифровій обробці можна зменшити шуми та артефакти на зображеннях, відновити втрачені деталі, покращити контрастність та роздільну здатність, що сприяє більш точній інтерпретації та аналізу медичної інформації [4].

По-друге, цифрова обробка медичних зображень використовується для передачі та зберігання зображень. За допомогою цифрової технології зображення можуть бути легко передані через мережу до різних медичних установ або збережені в електронному вигляді. Крім того, цифрова обробка дозволяє стискати зображення для ефективного зберігання без помітних втрат якості. Це сприяє зручній обміну медичними даними та забезпечує доступ до них у будь-який час та місце [2].

Обробка медичних знімків вимагає виявлення незримих деталей та об'єктів на зображенні, що важко розпізнати візуально без спеціальних маніпуляцій. Після аналізу стало зрозуміло, що існує безліч методів обробки зображень, але не всі з них підходять для медичних знімків. Важливою вимогою до таких методів є збереження повної інформації, оскільки якісний знімок допомагає лікареві поставити більш точний діагноз [5].

Таким чином, обробка медичних зображень відіграє важливу роль у покращенні діагностики та аналізу медичної інформації. Вона дозволяє виявити недосяжні для ока деталі, зберегти важливу інформацію та сприяє точнішому розумінню медичних знімків для ефективного лікування та дослідження [4].

1.7 Порівняння систем обробки медичних зображень

У процесі дослідження методів обробки медичних зображень було вирішено провести порівняльний аналіз існуючих готових рішень, зокрема Dicom Library, 3Dicom Viewer – Convert 2D DICOM Scans to Fully Immersive 3D, 3D Slicer image. Головна мета цього порівняння полягає в ідентифікації

переваг та недоліків кожної системи з метою вибору оптимального підходу для розробки програмного забезпечення [6].

Дослідження аналогічного готового продукту надає можливість вивчити та оцінити функціональність, особливості та недоліки. Це дозволяє зрозуміти потреби та вимоги користувачів, виявити прогалини в функціоналі або можливості для покращення. Також це допомагає виявити найкращі практики та інновації, що вже використовуються у сфері, і застосувати їх у нових проєктах [7]. Це сприяє розвитку та покращенню програмних рішень, що відповідають сучасним вимогам та потребам користувачів [8].

Враховуючи отримані результати порівняльного аналізу, ми зможемо визначити оптимальний підхід до розробки програмного забезпечення для аналізу медичних зображень, та його вдосконалення [8].

1.8 Dicom library

DICOM Library надає різноманітні інструменти для відображення медичних зображень та маніпуляцій над ними (рис. 1.1).

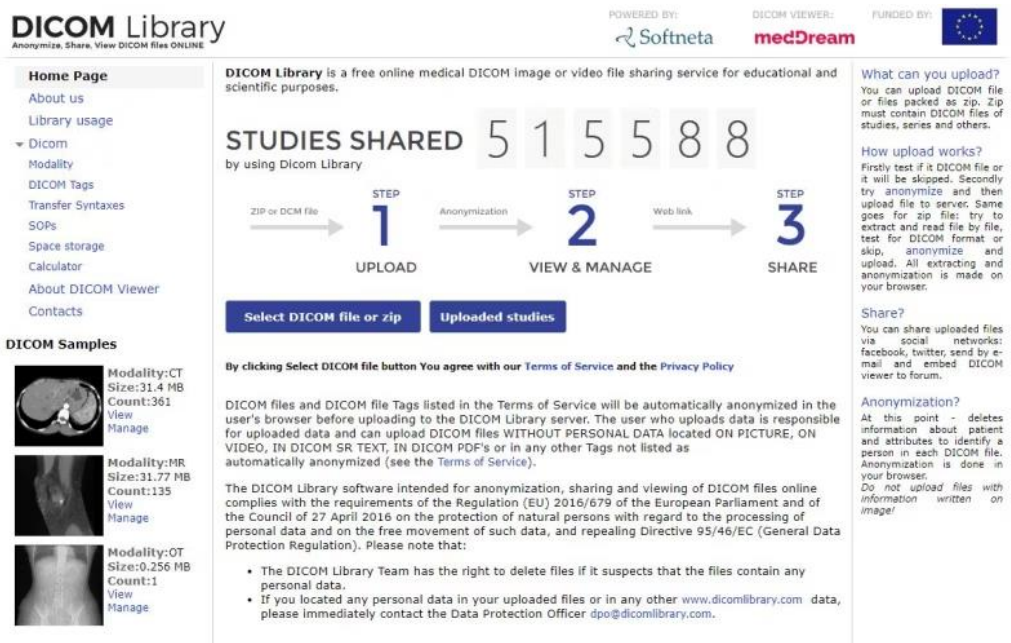

Рисунок 1.1 – Інтерфейс DICOM library

Ви можете використовувати різні режими візуалізації, такі як режими монохромної й кольорової, сегментації, прозорості та інші. Також є можливість регулювати яскравість, контрастність, гамму, фільтри та інші параметри зображення [9].

Сервіс дозволяє виконувати різні види аналізу та вимірювання. Можна здійснювати вимірювання розмірів, площі, об'єму, густини, інтенсивності та інше. Це корисно для визначення розмірів пухлин, оцінки ступеня ураження тканин та виконання інших медичних аналізів [9].

Особливості:

а) зручна взаємодія та обмін інформацією:

1) DICOM Library має інтуїтивно зрозумілий інтерфейс, який дозволяє легко взаємодіяти та обмінюватися медичними зображеннями;
2) можна надсилати зображення іншим користувачам, отримувати коментарі, використовувати функцію чату та спільно працювати над дослідженнями;

б) підтримка різних типів медичних зображень:

1) DICOM Library не обмежується певними типами медичних зображень;
2) інтерфейс підтримує широкий спектр форматів зображень, включаючи знімки з рентгенівських апаратів, комп'ютерної томографії, магнітно-резонансної томографії, ультразвуку та інших;

в) безпека та конфіденційність:

1) DICOM Library розуміє важливість медичних даних;
2) при завантаженні DICOM-файлів у публічну PACS-систему, слід мати на увазі, що конфіденційність не може бути повністю гарантована, тому важливо обережно використовувати сервіс, дотримуючись належних заходів безпеки;

г) мовна підтримка:

1) DICOM Library надає технічну підтримку на кількох мовах, зокрема англійській, іспанській, французькій;
2) дозволяє користувачам з різних країн та регіонів зручно користуватися сервісом [9].

1.9 3Dicom Viewer

3Dicom Viewer – це передовий онлайн-сервіс, спеціально розроблений для перегляду та аналізу DICOM-зображень з метою поліпшення діагностики та лікування у сфері медицини. Завдяки своїм розширеним функціональним можливостям, 3Dicom Viewer стає незамінним інструментом для медичних фахівців, дослідників та студентів (рис. 1.2) [10].

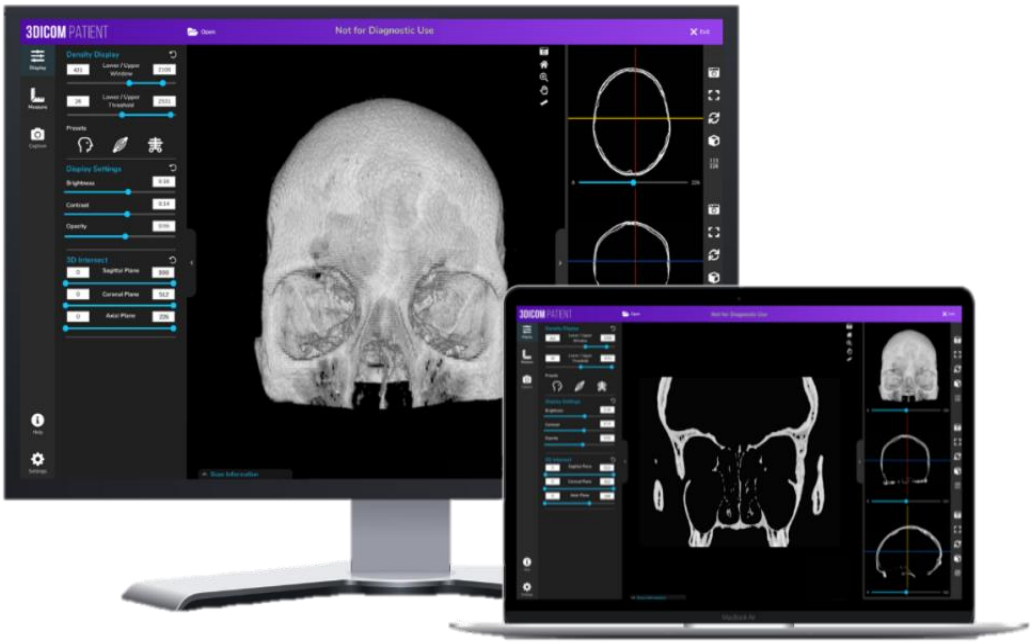

Рисунок 1.2 – 3Dicom Viewer

Розглянемо основні особливості.

Миттєвий доступ до DICOM-зображень. 3Dicom Viewer забезпечує швидкий та зручний перегляд DICOM-файлів без необхідності встановлення

спеціалізованого програмного забезпечення. Ви можете легко завантажити DICOM-зображення зі свого комп'ютера або ввести URL-адресу зображення для негайного перегляду.

Розширені можливості візуалізації. 3Dicom Viewer дозволяє вам ретельно досліджувати медичні зображення завдяки різноманітним інструментам візуалізації. Ви можете використовувати різні режими відображення, регулювати контрастність, яскравість та інші параметри, а також застосовувати фільтри для отримання найкращої якості зображення.

Зручні інструменти вимірювання та аналізу. 3Dicom Viewer надає вбудовані інструменти для вимірювання розмірів, площі, об'єму, густини та інших параметрів на DICOM-зображеннях. Це дозволяє проводити точні вимірювання та аналізувати дані для детальнішого вивчення патологій або уражень.

Взаємодія та спільна робота. 3Dicom Viewer дозволяє легко обмінюватися DICOM-зображеннями з іншими користувачами, спільно працювати над аналізом зображень і надавати допомогу в онлайн-режимі. Ви можете надіслати посилання на зображення колегам або пацієнтам.

Зручне зберігання та організація зображень. 3Dicom Viewer надає можливість зберігати та каталогізувати DICOM-зображення для подальшого використання. Ви можете створювати папки, маркувати зображення та здійснювати пошук за різними параметрами для легкого доступу до потрібних даних [10].

3Dicom Viewer – це потужний інтуїтивно зрозумілий інструмент, який допомагає медичним спеціалістам отримувати точнішу та швидшу діагностику, а також покращує співпрацю та обмін інформацією в медичній галузі. Завдяки цьому сервісу, процес аналізу DICOM-зображень стає ефективнішим, надійнішим та зручним [11].

1.10 3D Slicer

3D Slicer – це відкрите програмне забезпечення для візуалізації, аналізу та обробки медичних зображень. Розроблений у співпраці з медичними дослідниками та фахівцями з усього світу, 3D Slicer надає широкі можливості для вивчення медичних даних та покращення діагностики та лікування (рис. 1.3) [12].

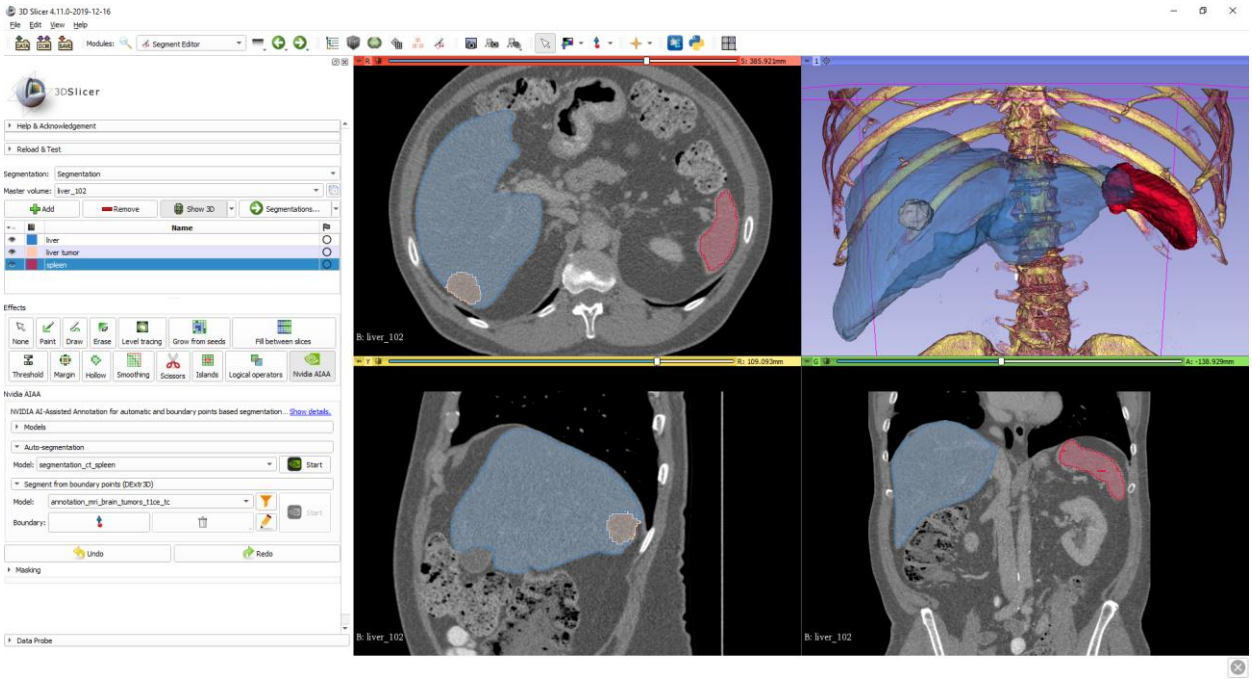

Рисунок 1.3 – 3D slicer

Розглянемо основні особливості.

Візуалізація та рендерінг. 3D Slicer дозволяє відтворювати медичні зображення у тривимірному просторі, що дозволяє отримувати більш детальну та об'ємну інформацію. Ви можете відтворювати зображення у різних режимах, налаштовувати освітлення та використовувати різні методи рендерінгу для досягнення найкращої візуалізації.

Обробка та сегментація. 3D Slicer надає інструменти для обробки та сегментації медичних зображень. Ви можете виокремлювати окремі структури або органи на зображеннях, створювати контури, видаляти шум та виконувати інші операції для покращення якості зображень та аналізу даних.

Аналіз та вимірювання. 3D Slicer надає набір інструментів для аналізу та вимірювання медичних зображень. Ви можете виконувати вимірювання розмірів, об'єму, інтенсивності й інших параметрів для отримання об'єктивних даних та докладного аналізу.

Моделювання та візуалізація 3D-моделей. 3D Slicer дозволяє створювати та візуалізувати тривимірні моделі на основі медичних зображень. Ви можете відтворювати складні структури, робити зрізи та досліджувати моделі з різних кутів для отримання додаткових візуальних інформацій.

Інтеграція з іншими інструментами. 3D Slicer підтримує інтеграцію з різними програмними рішеннями та бібліотеками для розширення функціональності. Ви можете використовувати додаткові модулі та плагіни для виконання специфічних завдань та розширення можливостей програмного забезпечення [13].

3D Slicer є професійним інструментом для аналізу та обробки медичних зображень, який сприяє покращенню діагностики, плануванню лікування та дослідженню в області медицини. З його допомогою медичні фахівці можуть отримати більш детальний та об'єктивний аналіз зображень [13].

1.11 Порівняння досліджених сервісів

DICOM Library, 3Dicom Viewer та 3D Slicer надають широкий спектр функцій для роботи з обробкою та візуалізацією медичних зображень. Проте кожен з цих сервісів має ряд переваг та недоліків, які показані в таблиці 1.1.

Таблиця 1.1 – Порівняння сервісів

	DICOM Library	**3Dicom Viewer**	**3D Slicer**
Переваги	Безкоштовний доступ	Зручний онлайн-сервіс для перегляду DICOM-зображень	Відкрите програмне забезпечення з

Продовження табл. 1.1

	DICOM Library	**3Dicom Viewer**	**3D Slicer**
			великим набором інструментів для візуалізації, аналізу та обробки медичних зображень
	Підтримка різних типів медичних зображень	Розширені функції візуалізації та аналізу зображень	Можливість моделювання та візуалізації 3D-моделей
	Можливість обміну та видалення інформації про пацієнта	Сумісність з сенсорними екранами	Інтеграція з іншими інструментами і бібліотеками
			Активна спільнота користувачів та розробників
Недоліки	Відсутність детальної інформації про технічну підтримку	Обмежений функціонал в порівнянні з іншими сервісами	Складність навчання для нових користувачів
	Відсутність розширених функцій для аналізу та	Відсутність деяких функцій аналізу	Вибагливість до апаратних характеристик

Продовження табл. 1.1

	DICOM Library	**3Dicom Viewer**	**3D Slicer**
	обробки медичних зображень		комп'ютера
	Відсутність спільноти користувачів	Обмежена підтримка PACS-систем	Необхідність завантаження та інсталяції програмного забезпечення
	Конфіденційність		

Загалом, кожен з цих сервісів має свої переваги та недоліки, і вибір залежить від конкретних потреб й вимог користувача. DICOM Library підходить для простого обміну DICOM-зображеннями, 3Dicom Viewer – для швидкого перегляду та базового аналізу, а 3D Slicer – для більш розширених можливостей аналізу та обробки медичних зображень.

1.12 Висновки до першого розділу

В цьому розділі було проаналізовано актуальність розробки програмного забезпечення, сформовано перелік завдань, та визначені загальні положення і методи, котрі можна застосувати при вдосконаленні. Також проаналізована проблема обробки медичних знімків. Було визначено загальну структуру програмного забезпечення та функціонал, який має бути присутнім для швидкого аналізу.

Окрім цього проведений аналіз аналогічних сервісів обробки медичних зображень. Виявлені спільні риси, недоліки й функції.

Специфікація даної роботи полягає у розробці додатку для швидкого

проведення аналізу. Акцент зроблений на мобільність додатку та швидкість з якою він дозволить працювати над медичними зображеннями.

2 ПРОЄКТУВАННЯ

2.1 Комп’ютерні процеси обробки

Зір є найбільш досконалим з органів чуття людини і максимальну кількість інформації людина сприймає очима. Однак, на відміну від людей, здатних сприймати електромагнітне випромінювання лише у видимому діапазоні, машинна обробка зображень охоплює практично весь електромагнітний спектр від гамма-випромінювання до радіохвиль [14].

У всьому діапазоні від обробки зображення до машинного зору можна розрізнити комп'ютерні процеси низького, середнього та високого рівнів. Процеси низького рівня стосуються лише примітивних операцій типу передобробки з метою зменшення шуму, підвищення контрасту або покращення різкості зображень [2].

Процеси середнього рівня охоплюють такі завдання, як сегментація (розподіл зображень на області або виділення на ньому об'єктів), опис об'єктів та стиснення їх у зручну для комп'ютерної обробки форму, а також класифікація (розпізнавання) окремих об'єктів. На вході зображення, виході – атрибути, що витягуються із зображення (наприклад межі областей, лінії контурів) [1].

Процеси високого рівня включають «осмислення» набору розпізнаних об'єктів, як це робиться в аналізі зображень [15].

В області обробки та аналізу зображень виділяють п'ять основних класів методів та алгоритмів [16].

Поліпшення якості – цей клас складається з методів, які використовуються для зменшення шумів та видалення артефактів, а також для підвищення контрасту області інтересу на зображеннях. Можливість використання тих чи інших процедур покращення якості суттєво залежить від того, як буде проводитись наступний аналіз – візуально або за допомогою

відповідних комп'ютеризованих методів. У разі комп'ютерного аналізу, кількість застосовуваних алгоритмів покращення візуальної якості зображень має бути зведена до мінімуму [3].

Сегментація зображень – відділення аналізованого, об'єкта, структури або області з навколишнього фону. Сегментація належить до базових кроків, якість виконання яких багато в чому визначає точність, а часом навіть можливість подальшого комп'ютеризованого аналізу зображень. Методи сегментації базуються на яскравій, градієнтній та текстурній інформації зображення [2].

Кількісний аналіз застосовується до відсегментованих об'єктів зображення з метою виділення суттєвої діагностичної інформації: розмір, форма, текстура об'єктів тощо. Цей клас методів використовується для поєднання двох цифрових зображень однієї і тієї ж частини людського тіла. Процес є доцільним, якщо отримана в результаті сувміщення карта може бути використана для подальшої обробки та аналізу зображення. Сумісними зображеннями можуть бути знімки одного й того пацієнта в різних ситуаціях або в різний час. Також у процесі діагностики часто виникає потреба у поєднанні зображення пацієнта із зображенням здорового людини. Отримувана карта відповідності може використовуватися для попіксельного порівняння зображень, моніторингу зміни форми та зростання новоутворень [16].

Наступний клас методів використовується для стиснення, архівування, зберігання, а також для пошуку в базах даних. Роль цього напряму постійно зростає, оскільки за останні роки значно збільшилась кількість та розміри цифрових діагностичних зображень. Щодня робиться безліч медичних знімків, над якими необхідно проводити різноманітні маніпуляції, які передбачають їхнє подальше зберігання, передачу, пошук, тощо [15].

На сьогоднішній день розвиток нових технологій та цифрової техніки призвело до появи великої кількості нових методів обробки, які можна розділити на дві категорії: просторові та частотні [3].

2.2 Формат .dcm

Формат файлу .dcm (Digital Imaging and Communications in Medicine) був розроблений в середині 1980-х років у рамках ініціативи з стандартизації обміну медичними зображеннями. Засновником та ініціатором розробки стандарту DICOM була Американська асоціація радіології (American College of Radiology – ACR) разом з Національним інститутом стандартів та технологій (National Institute of Standards and Technology – NIST) [2].

Перші кроки у розробці DICOM були зроблені в 1983 році. На той момент, медичні зображення зберігалися на фізичних носіях, таких як рентгеновські плівки, і відсутність стандартів унеможливлювала ефективний обмін цими даними. З метою вирішення цієї проблеми було створено DICOM, який задовольняв потреби стандартизованого обміну та зберігання медичних зображень [2].

Перший реліз DICOM став доступним у 1985 році під назвою ACR-NEMA 1.0 (American College of Radiology – National Electrical Manufacturers Association). Протягом наступних років, розробка DICOM продовжувалася і стандарт постійно вдосконалювався. У 1988 році був випущений DICOM 2.0, а в 1993 році вийшла остаточна версія DICOM 3.0, яка стала базою для подальшого розвитку та впровадження стандарту [16].

DICOM (Digital Imaging and Communications in Medicine) є стандартним форматом для зберігання та обміну медичних зображень, таких як рентгенограми, томограми, магнітно-резонансні зображення та інші [14].

Формат .dcm використовує структуровані дані DICOM, які містять заголовок та набір атрибутів. Заголовок містить інформацію про тип файлу, версію стандарту, розмір даних та інші метадані. Атрибути містять самі дані, такі як інформація про пацієнта, тип зображення, параметри зйомки та інші важливі деталі. Формат використовує протокол передачі даних на основі транспортного рівня TCP/IP, що дозволяє легко обмінюватися медичними зображеннями між різними системами та пристроями [14].

2.3 Протокол TCP/IP

Використання протоколу передачі даних на основі транспортного рівня TCP/IP (Transmission Control Protocol/Internet Protocol) є одним з ключових аспектів протоколу DICOM (Digital Imaging and Communications in Medicine) [15].

Протокол TCP/IP (Transmission Control Protocol/Internet Protocol) є набором протоколів, що використовуються для передачі даних в комп'ютерних мережах, зокрема в Інтернеті. Він складається з двох основних протоколів: TCP, що відповідає за надійну передачу даних, та IP, що відповідає за маршрутизацію та адресацію пакетів [15].

TCP/IP гарантує надійну та послідовну передачу даних, забезпечуючи контроль потоку, цілісності та керування перевантаженням. Він розбиває дані на пакети, які передаються через мережу, а приймач збирає їх знову в початковий потік даних. Протокол також відповідає за адресацію пакетів й маршрутизацію, що дозволяє передавати дані між різними комп'ютерами у мережі [16].

Протокол TCP/IP є основою для передачі даних в інтернеті та широко використовується в комп'ютерних мережах. Він забезпечує надійну передачу даних й стандартизовану комунікацію між різними пристроями та системами. TCP (Transmission Control Protocol) впорядковує дані, забезпечує їх доставку відправнику, контролює цілісність та вирішує проблеми, такі як втрати даних або повторне передавання [17].

Протокол DICOM може використовувати додаткові заходи безпеки, такі як шифрування, для захисту конфіденційності медичних даних під час передачі [2].

TCP/IP дозволяє розбити дані на менші сегменти для передачі по мережі та їх подальше зібрання на приймачі. Це особливо важливо для передачі великих обсягів медичних зображень, які можуть вимагати розбиття на частини.

Загалом, використання протоколу передачі даних на основі TCP/IP в форматі .dcm дозволяє забезпечити стабільну, безпечну та надійну передачу між різними системами, що підтримують DICOM. Це дозволяє лікарям, дослідникам та іншим медичним професіоналам ефективно обмінюватися медичними зображеннями та іншими важливими даними [17].

2.4 Просторові методи обробки зображень

Категорія методів, що застосовуються у просторовій області, поєднує підходи, засновані на прямому маніпулюванні пікселями зображення. Частотні, у свою чергу, ґрунтуються на модифікації сигналу, що формується шляхом застосування перетворення Фур'є до зображення, що обробляється [2].

Процеси просторової обробки області описуються рівнянням:

$$g(x,y)=T(f(x,y)),$$

де $f(x,y)$ – вхідне зображення; $g(x,y)$ – оброблене зображення; T – оператор над f.

Інверсія в контексті обробки зображень означає зміну яскравості або кольору кожного пікселя на протилежне значення. Це призводить до створення негативного зображення, де світлі області стають темними, а темні – світлими [2].

Формула для інверсії зображення залежить від типу представлення кольору пікселів. Якщо зображення представлене у форматі сірого відтінку, то формула інверсії виглядає так:

$$g(x,y) = 255 - f(x,y),$$

де $f(x,y)$ – значення яскравості пікселя на позиції (x,y), а $g(x,y)$ –

значення інверсованого пікселя.

У випадку кольорового зображення, яке використовує модель RGB, формула інверсії застосовується до кожного каналу окремо. Тобто, для кожного пікселя з трьома каналами (червоний, зелений, синій), формула має наступний вигляд:

$$g(x,y) = (255 - r(x,y), 255 - g(x,y), 255 - b(x,y)),$$

де $r(x,y)$, $g(x,y)$, $b(x,y)$ – значення каналів червоного, зеленого і синього відповідно для пікселя на позиції (x,y), а $g(x,y)$ – значення інверсованого пікселя.

Степенева обробка (або потужність) є одним з методів обробки зображень, що використовується для зміни яскравості або контрастності зображення шляхом піднесення піксельних значень до певної степені. Цей метод може бути корисним для підсилення деталей або покращення візуальної якості зображення [2].

Формула для степеневої обробки зображення може бути виражена :

$$g(x,y) = c * f(x,y)^y,$$

де $g(x,y)$ представляє оброблене зображення, $f(x,y)$ відповідає вихідному зображенню; c є константою, яка контролює яскравість обробленого зображення.

Значення показника степені може бути додатним або від'ємним. Додатні значення (>1) збільшують контрастність та підсилюють яскраві ділянки зображення, тоді як від'ємні значення (<1) зменшують контрастність та затемнюють яскраві ділянки. Наприклад, якщо ми хочемо застосувати степеневу обробку зображення з показником степені 1.5, то формула буде виглядати наступним чином:

$$g(x,y) = c * f(x,y)^{\wedge}1.5.$$

Застосовуючи цю формулу до кожного пікселя зображення, ми отримаємо оброблене зображення зі зміненою яскравістю та контрастністю [1].

Варто зауважити, що реалізація степеневої обробки може варіюватися в залежності від конкретної програмної бібліотеки або середовища обробки зображень, яке використовується [2].

Лінійна фільтрація є одним з основних методів обробки зображень, що використовується для виокремлення певних характеристик зображення шляхом проходження його через лінійний фільтр. Цей метод використовується для згладжування, розмиття, підсилення контурів або виділення певних структур на зображенні [17].

Лінійний фільтр може бути представлений у вигляді згортки маски (ядро) зображення. Кожен вихідний піксель обчислюється шляхом перемноження вхідних пікселів на відповідні елементи маски та сумування результатів. Цей процес можна математично описати наступною формулою:

$$g(x,y) = \Sigma[f(x-k,y-l) * h(k,l)],$$

де $g(x,y)$ представляє оброблене зображення, $f(x,y)$ відповідає вхідному зображенню. $h(k,l)$ представляє маску (ядро) лінійного фільтра, де k та l – індекси елементів маски; Σ вказує на сумування по всіх елементах маски [2].

Функція лінійної фільтрації полягає в тому, щоб кожен піксель обробленого зображення був результатом лінійної комбінації пікселів вхідного зображення, зважених коефіцієнтами, визначеними маскою. Це дозволяє змінювати різні характеристики зображення, такі як рівень розмиття, контрастність або виділення деталей [8].

Застосування лінійної фільтрації включає широкий спектр операцій, таких як розмиття Гауса, розшаровування, розмивання рухомого об'єкту,

підвищення гостроти і багато інших, які можуть покращити візуальну якість та виконати різноманітні завдання обробки зображень [1].

Порогова фільтрація є одним з методів обробки зображень, який використовується для виділення об'єктів або підвищення контрастності на зображенні за допомогою встановленого порогового значення [18].

Функція порогової фільтрації полягає у визначенні пікселів, значення яких перевищує або менше за певне порогове значення, і призначенні їм нового значення. Цей метод може використовуватися для бінаризації зображень, де пікселі, що перевищують порогове значення, встановлюються на максимальне значення (наприклад, білий колір), а пікселі, що менші за поріг, встановлюються на мінімальне значення (наприклад, чорний колір) [19].

Порогова фільтрація може використовуватися для різних цілей, таких як виділення об'єктів, видалення шуму, покращення контрастності та розділення областей зображення. Наприклад, при аналізі медичних зображень, порогова фільтрація може допомогти виділити певні структури або патологічні ділянки, що мають відмінне значення яскравості від навколишніх тканин. Це сприяє поліпшенню візуалізації та діагностики на медичних зображеннях [2].

Порогова фільтрація використовує різні функції та рівняння для обробки зображень. Основні функції та рівняння, що використовуються в пороговій фільтрації, включають:

Функція порогової обробки (Thresholding Function):

$$g(x,y) = \{v1, \text{ if } f(x,y) > T,$$
$$v2, \text{ if } f(x,y) \leq T,$$

де $g(x,y)$ – вихідне оброблене зображення, $f(x,y)$ – вхідне зображення, T – порогове значення. $v1$, $v2$ – нові значення пікселів після застосування порогової фільтрації.

Бінаризація (Binary Thresholding):

$$g(x,y) = \{v1, \text{ if } f(x,y) > T,$$
$$v2, \text{ if } f(x,y) \leq T,$$

де $v1$ та $v2$ зазвичай приймають значення 0 та 255 для чорного та білого кольорів відповідно.

Адаптивна порогова фільтрація (Adaptive Thresholding):

$$g(x,y) = \{v1, \text{ if } f(x,y) > T(x,y),$$
$$v2, \text{ if } f(x,y) \leq T(x,y),$$

де $T(x,y)$ – адаптивне порогове значення, яке залежить від локального контексту пікселя (наприклад, середнє значення або медіана пікселів у певному околі).

2.5 Розробка концепційної моделі

Для розробки додатку було створену модель у вигляді контекстної діаграми (див. рис. 2.1).

Ця схема описує процес обробки медичних зображень. Першим кроком є використання цифрового зображення, яке отримане з медичного обладнання після діагностичної процедури. Зображення може бути отримане в цифровому форматі або оцифроване з плівкових рентгенограм [17].

Другим кроком є вибір категорії методів обробки зображення, що залежить від медичного працівника. Ця категорія може включати виділення контурів, підвищення деталізації, видалення шумів та інші техніки. Вибір категорії залежить від типу зображення та конкретних цілей, які ставить медичний працівник [19].

На виході отримується покращене медичне зображення, яке є обробленим відображенням. Це зображення оцінюється шляхом порівняння з

вихідним зображенням і використовується для постановки діагнозу. Управління цим процесом здійснюється медичним працівником, який керується рекомендаціями, що містяться в інструкції користувача для обробки медичних зображень [19].

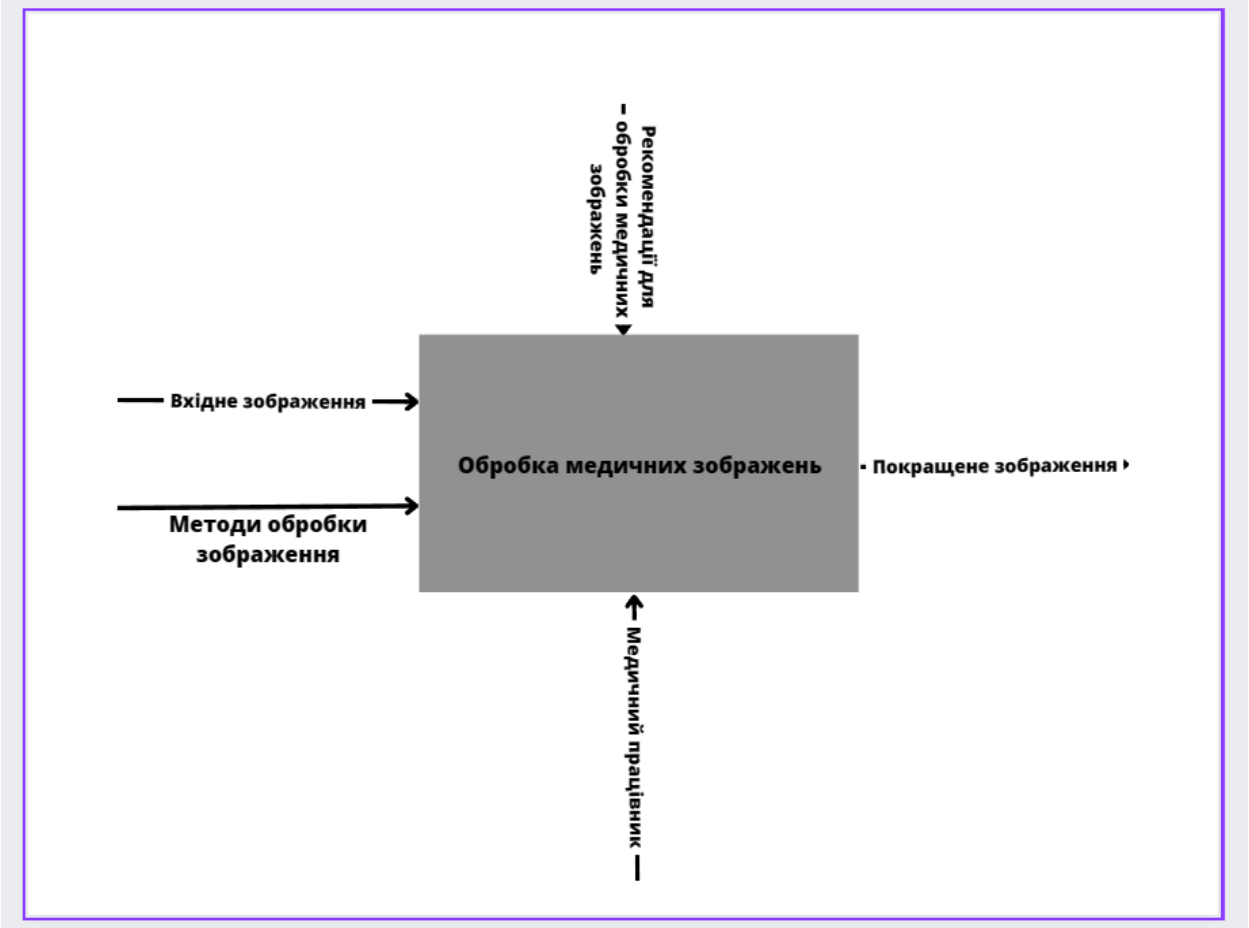

Рисунок 2.1 – Діаграма системи обробки

Ця схема показує, що обробка медичних зображень є важливим кроком у діагностичному процесі, де медичний працівник має можливість поліпшити якість зображення для більш точного діагностування та прийняття відповідних медичних рішень [17].

Для більш детального представлення цього процесу було побудовано декомпозицію концептуальної схеми (див. рис. 2.2).

У процесі оцінки результатів обробки медичного зображення застосовуються два підходи. По-перше, проводиться візуальне порівняння між

вихідним зображенням та обробленим знімком, яке здійснюється лікарем (суб'єктивна оцінка). Лікар оцінює відповідність обробленого зображення його очікуванням та визначає, чи поліпшилася візуальна якість зображення [3].

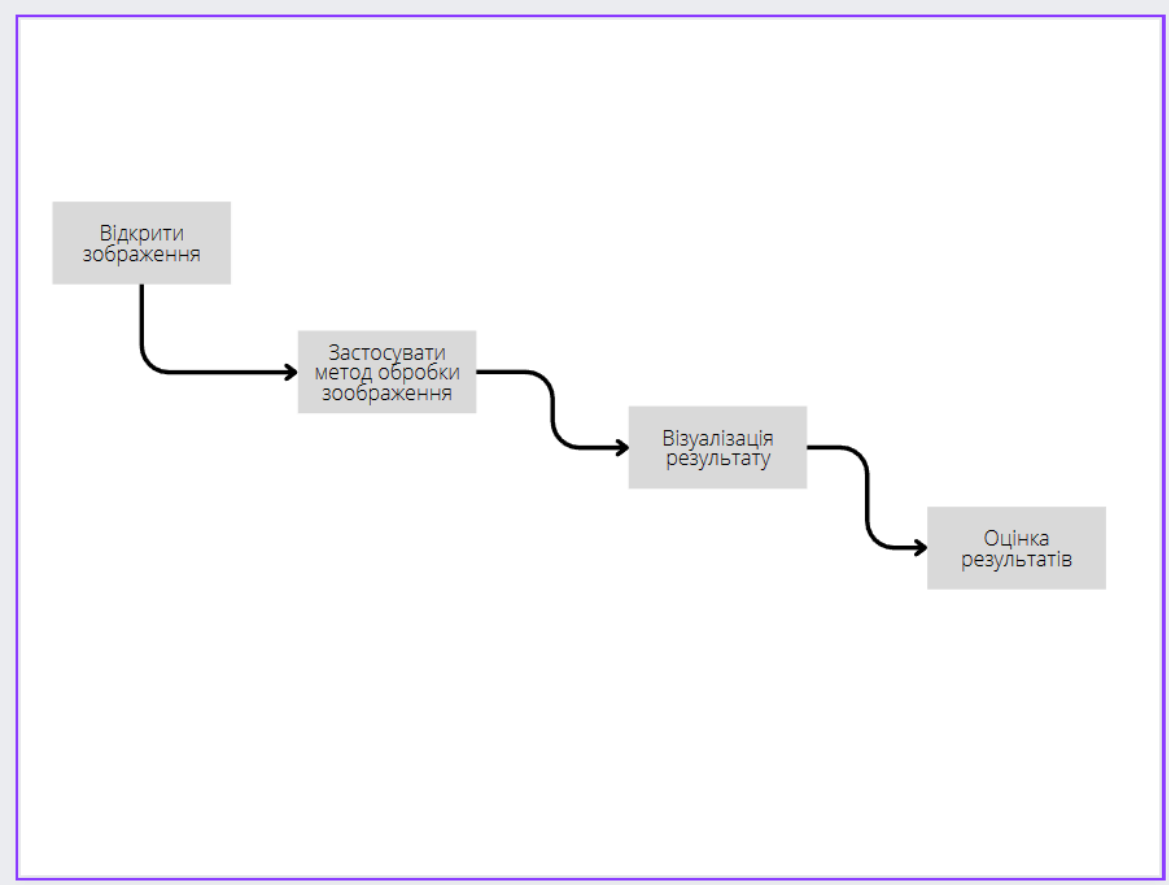

Рисунок 2.2 – Декомпозиція концептуальної діаграми

Об'єктивні показники оцінки надають кількісну оцінку якості обробленого зображення і можуть служити додатковими даними для аналізу. У поєднанні з суб'єктивною оцінкою лікаря вони допомагають зробити більш об'єктивне рішення щодо ефективності та придатності методів обробки медичних зображень [2].

Слід зазначити, що в деяких випадках використання одного методу обробки може не привести до досягнення бажаного результату. Тому в схемі, яка наведена (див. рис. 2.3), реалізовано два варіанти подальших дій після оцінки результатів обробки [1].

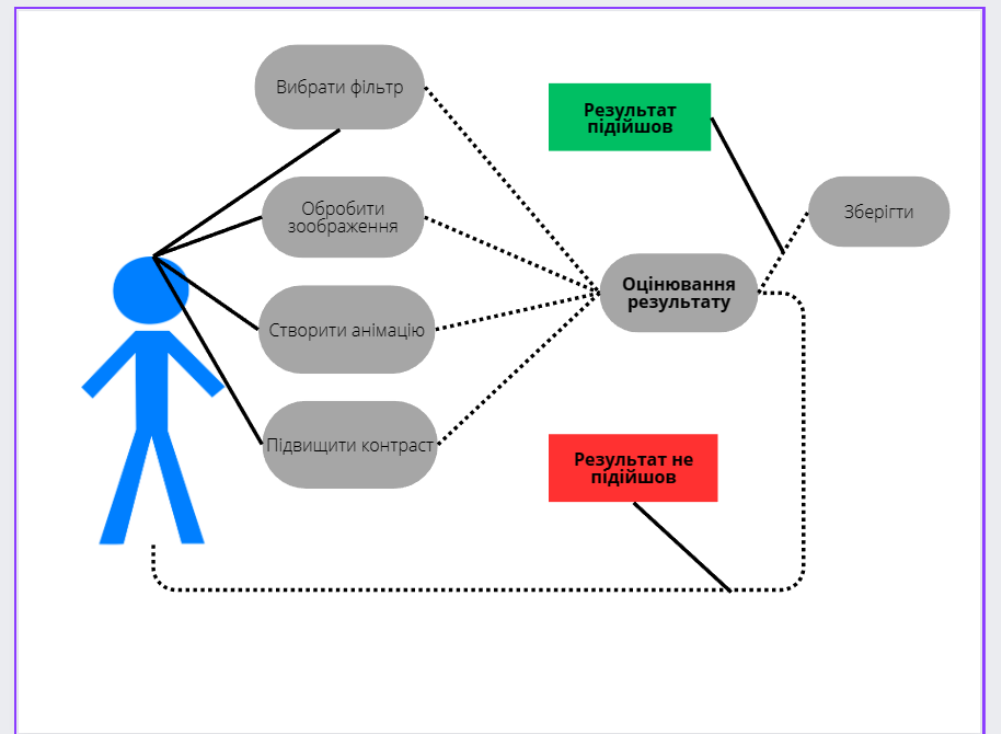

Рисунок 2.3 – UML-діаграма варіантів використання

Перший варіант передбачає повторне застосування методу обробки з новими параметрами або зміну самого методу для поліпшення якості зображення. Це дозволяє уточнити обробку і спробувати отримати кращі результати [5].

Другий варіант полягає у застосуванні додаткових методів або комбінації методів обробки. Це може включати використання фільтрів, алгоритмів зміни контрастності, видалення шуму або виокремлення конкретних об'єктів на зображенні. Такий підхід дозволяє досягнути більш точної обробки та покращити візуальну якість медичного зображення [5].

Ці два варіанти продовження робіт після оцінки результатів обробки дають можливість медичному працівнику вибрати оптимальний шлях для досягнення необхідної мети, яка може включати покращення якості зображення, виділення певних деталей або виявлення патологічних змін на зображенні [17].

Перший варіант є задовільним закінченням обробки знімку та його подальшого збереження, оскільки він відповідає бажаному результату і не

потребує додаткових маніпуляцій. У цьому випадку, після оцінки результату, оброблене зображення вважається придатним для використання і може бути збережено для медичних цілей або подальшого аналізу [18].

Другий варіант передбачає потребу у застосуванні іншого методу обробки замість того, що вже був застосований, або використання додаткового методу, щоб досягти бажаного результату. Це означає, що перша спроба обробки може бути не задовільною або не досягнути поставленої мети. У такому випадку медичний працівник повинен вибрати інший метод або комбінацію методів, які дозволять досягти бажаного результату обробки знімку [18].

Варіанти після оцінки результатів обробки забезпечують медичному працівнику можливість прийняти рішення про подальші дії зі знімком. Перший варіант дозволяє завершити обробку та зберегти знімок, вважаючи його задовільним. Другий варіант вимагає додаткових зусиль та застосування іншого методу, щоб досягти бажаного результату [18].

2.6 Розробка DFD-діаграми компонентів

Для розробки інформаційної моделі системи доцільно використовувати нотацію DFD, яка здатна продемонструвати як кожен компонент перетворює свої вхідні дані у вихідні, а також показати відносини між цими процесами.

Результат побудови схеми потоку даних, використовуючи методологію DFD (див. рис. 2.4).

З діаграми можна помітити, що медичний працівник, який є механізмом управління, представлений у вигляді блоку сутності та виконує всю послідовність дій: відкриває растрове зображення; застосовує до візуалізованого відображеного зображення метод обробки; візуалізує результат застосування методу; проводить оцінку результату обробки, у результаті вихідним параметром є поліпшене медичне зображення, керуючись яким лікар ставитиме діагноз [1].

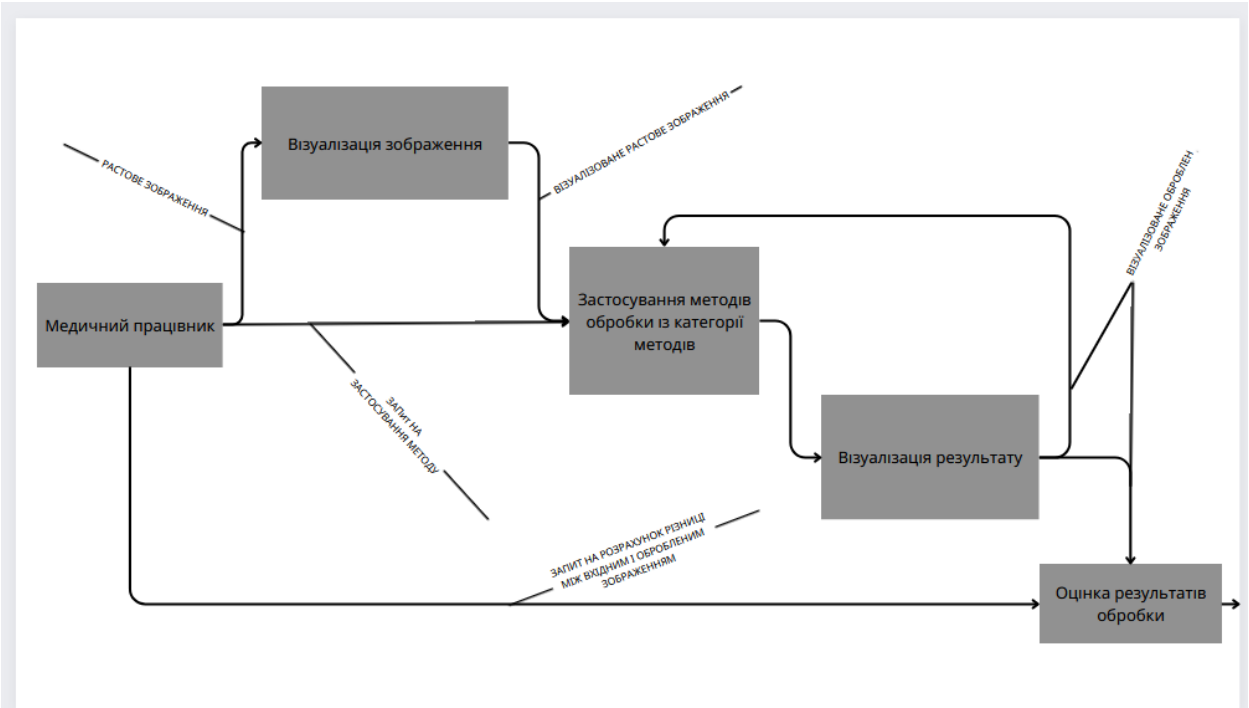

Рисунок 2.4 – DFD-діаграма потоку даних

2.7 Розробка інтерфейсу

При розробці інтерфейсу необхідно забезпечити максимальну функціональність та можливості редактора, але важливо також уникати перевантаження користувача надмірною кількістю меню, кнопок, зображень та тексту. Головна мета – забезпечити зручність та ефективність використання графічного інтерфейсу (рис. 2.5) [3].

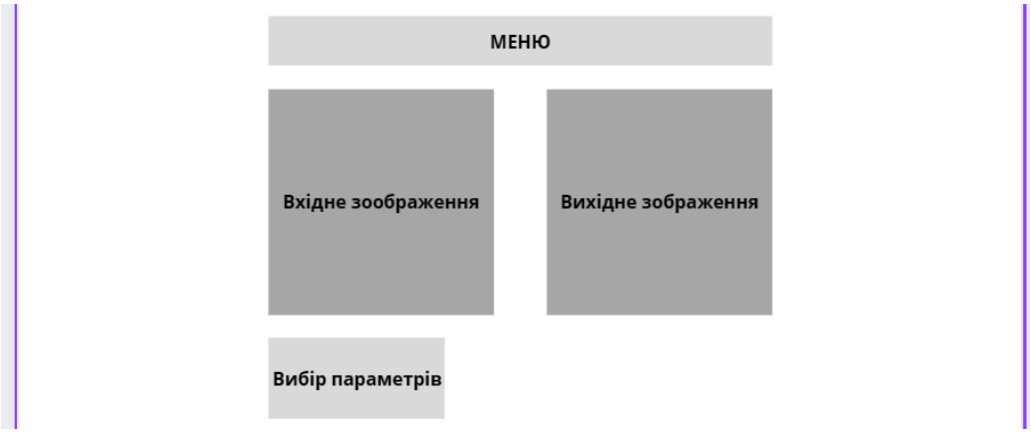

Рисунок 2.5 – Спрощений макет графічного інтерфейсу

Для досягнення цих цілей можна використовувати принципи простоти та зрозумілості. Графічний інтерфейс повинен бути інтуїтивно зрозумілим і легким у використанні, щоб користувачі могли швидко орієнтуватися в його функціях. Для цього варто уникати заплутаних меню і складних ієрархій [2].

Окрім того, слід забезпечити ефективну організацію простору на екрані. Необхідно уникати перевантаження інтерфейсу зайвими елементами, зображеннями та текстом. Розташування елементів повинно бути логічним та зручним для користувача, щоб він міг швидко знаходити потрібні функції [15].

При розробці графічного інтерфейсу також важливо враховувати потреби та навички цільової аудиторії. Наприклад, якщо програма призначена для новачків, можна спростити інтерфейс, забезпечити більше пояснень та допомоги. У разі, якщо вона призначена для досвідчених користувачів, можна надати більше просунутих функцій та швидкого доступу до них [14].

Отже, розробка графічного інтерфейсу повинна бути зосереджена на балансі між функціональністю та зручністю використання. Він має забезпечувати потрібні можливості без перевантаження користувача надмірними елементами і дозволяти зручно та ефективно працювати з редактором [14].

2.8 Висновки до другого розділу

У другому розділі кваліфікаційної роботи було розроблено концептуальну модель, яка включає у себе опис основних функцій та взаємодій користувача з програмним забезпеченням. Концептуальна модель дозволяє нам уявити загальну структуру редактора та визначити його основні компоненти.

Крім того, була розроблена контекстна діаграма, яка показує зовнішнє середовище редактора та взаємодію з ним. Ця діаграма допомагає нам уявити контекст, в якому працює редактор, і визначити зовнішні системи та акторів,

які взаємодіють з ним.

Декомпозиція контекстної діаграми дозволила розкрити більш детальну структуру редактора та виділити окремі компоненти та підсистеми, які входять в його склад. Це допомагає нам краще розуміти внутрішню організацію редактора та взаємозв'язки між його компонентами.

3 РОЗРОБКА ТА ТЕСТУВАННЯ

3.1 Вибір мови програмування

Мова програмування Python має багато переваг для реалізації програмного забезпечення обробки медичних знімків. Наприклад: простота використання, багата екосистема, швидкість розробки, платформова незалежність, інтеграція з іншими мовами, машинне навчання та інші.

Python має простий і зрозумілий синтаксис, що робить його дуже доступним для початківців та легко зрозумілим для розробників. Це дозволяє швидко вивчати мову і починати розробку програмного забезпечення для обробки медичних знімків. Багата екосистема включає в себе велику кількість сторонніх бібліотек та фреймворків, які сприяють обробці зображень. Наприклад, бібліотеки, такі як OpenCV, NumPy, SciPy, scikit-image та PyDICOM, надають потужні функціональні можливості для обробки, аналізу та візуалізації медичних зображень.

Швидкість розробки Python дозволяє розробляти програмне забезпечення швидко завдяки своїм вбудованим структурам даних та високорівневим функціям. Він також підтримує модульність і повторне використання коду, що сприяє ефективній розробці та підтримці проєктів. Python може працювати на різних операційних системах, включаючи Windows, macOS та різні дистрибутиви Linux. Це дозволяє розробляти програмне забезпечення для обробки медичних знімків, яке може запускатися на різних платформах.

Python може взаємодіяти з кодом, написаним на інших мовах програмування, таких як C ++ або Java. Це дає можливість використовувати швидкість та ефективність низькорівневих мов, разом з Python.

Дана мова має потужні бібліотеки, такі як TensorFlow, PyTorch та scikit-

learn, які підтримують розвиток алгоритмів машинного навчання та розвитку штучного інтелекту. Це дозволяє використовувати розширені методи обробки медичних зображень, такі як сегментація, класифікація та виявлення аномалій.

Загалом, Python є потужним та гнучким інструментом для реалізації програмного забезпечення обробки медичних знімків, завдяки своїй простоті, багатій екосистемі та широкому колу підтримки спільноти розробників. Він надає зручність, продуктивність та можливості для виконання складних завдань обробки зображень в медицині.

3.2 Бібліотека pydicom

Однією з основних бібліотек, котра дозволить відкривати та проводити маніпуляції над медичними зображеннями – pydicom. Саме вона є основною для розробників медичного програмного забезпечення на мові Python.

Бібліотека дозволяє, у першу чергу, відкрити зображення рентгенограм, комп’ютерної томографії, магнітно-резонансної томографії та інші. Також pydicom надає простий інтерфейс для запису, зчитування та маніпуляцій з зображеннями DICOM.

Pydicom також забезпечує функціональність для обробки DICOM-зображень, включаючи перетворення піксельних даних, зміну розміру, обрізку, зміну контрастності, розмиття, фільтрацію та інші операції обробки зображень. Крім того, бібліотека дозволяє зберігати зображення у форматі DICOM та експортувати їх у різні інші формати зображень, такі як PNG або JPEG.

Завдяки своїм можливостям та простоті використання, Pydicom став популярним вибором для розробників, які працюють з медичними зображеннями в Python. Він забезпечує зручність та потужність для обробки й аналізу.

3.3 Графічний інтерфейс

Основою взаємодії між користувачем та машиною являється графічний інтерфейс. Він забезпечує більш інтуїтивне та візуальне сприйняття інформації, що являється його перевагою над командними запитами.

Завдяки графічному інтерфейсу користувач має швидко адаптуватися, ефективно використовувати продукт, зменшити кількість помилок, поліпшити досвід користування та віддати перевагу на користь інтуїтивно зрозумілого інтерфейсу.

З цього слідує, що графічний інтерфейс має бути інтуїтивно зрозумілим та не перевантажуватись великою кількістю елементів.

Розроблений мною інтерфейс програми має зрозумілий та простий вигляд. Це можна побачити на рисунку 3.1. Він включає в себе чотири початкові клавіші, котрі зустрічають користувача,а саме: “Open Image”, “Create Animation”, “Convert”, “Quit”. Кожен з них включає в себе функції або інші елементи інтерфейсу.

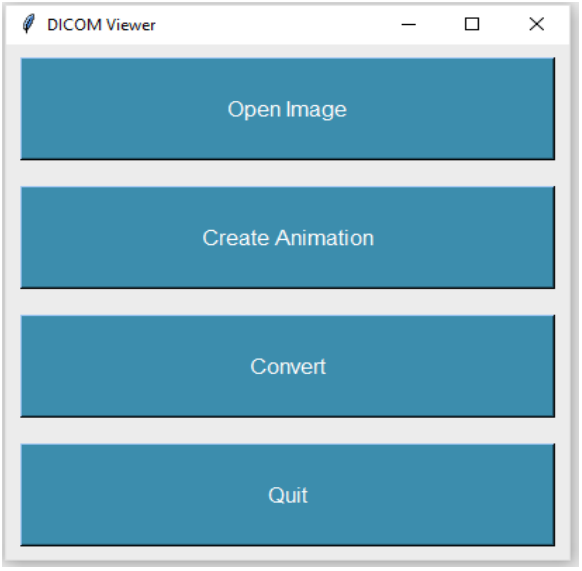

Рисунок 3.1 – Головне меню

Для початку роботи з зображенням, користувачу потрібно обрати необхідний пункт та натиснути на клавішу. Так, наприклад, якщо потрібно

обробити зображення, то натискаємо “Open Image”. Відкривається меню вибору файлів формату dcm (рис. 3.2).

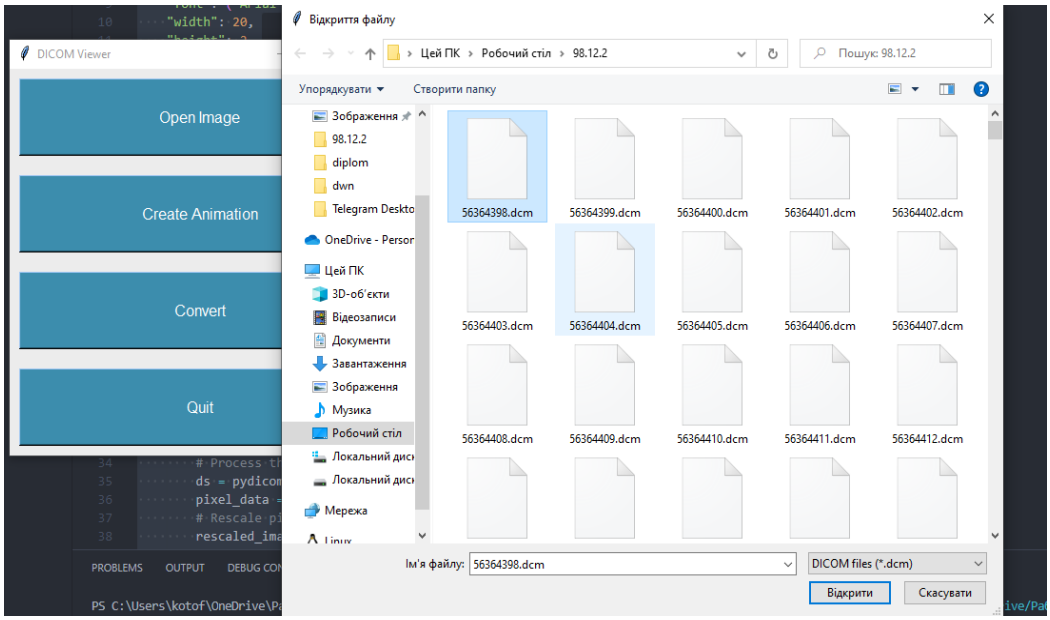

Рисунок 3.2 – Меню вибору зображення

3.4 Функціонал

Після того, як користувачем було обрано зображення, він зможе задіяти світові фільтри, накласти їх одне на одного або очистити їх за допомогою клавіші “Clear” (рис. 3.3).

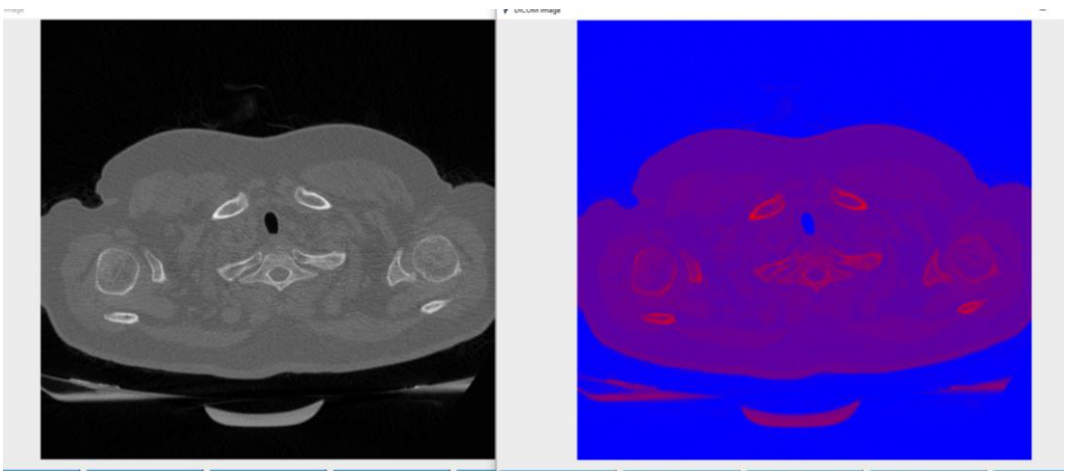

Рисунок 3.3 – Порівняння зображень

Наступна функція в головному меню програми “Create animation”. При натисканні на клавішу, по аналогії з “Open Image”, відкриється меню вибору. Але в цей раз можна вибрати лише папку (рис. 3.4).

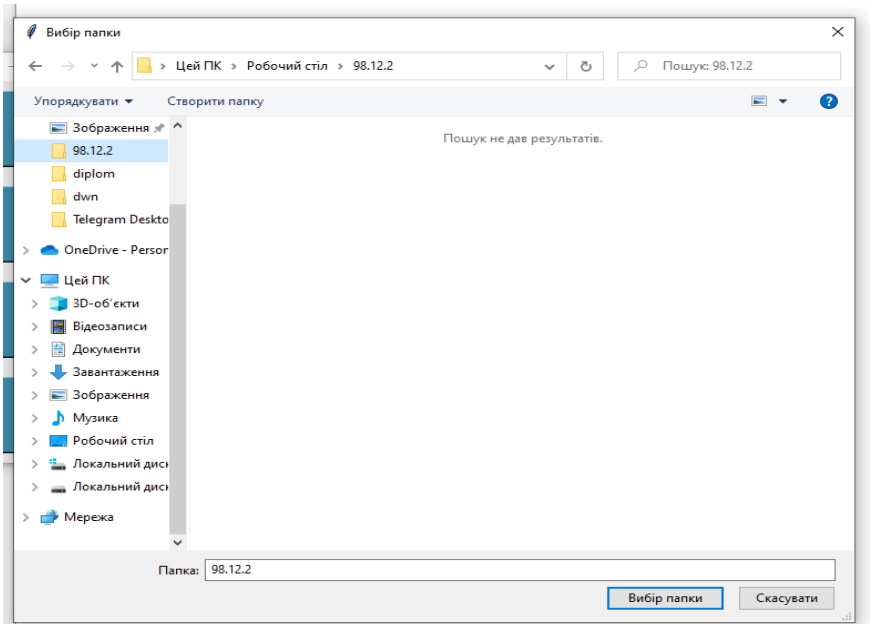

Рисунок 3.4 – Меню вибору папки

Після вибору папки і деякого часу очікування, буде створена GIF анімація із зображень, котрі там знаходяться. Саме вікно має додаткові клавіші “Play” і “Stop”, які мають відповідний функціонал. Також є повзунок для покадрового перегляду (рис. 3.5).

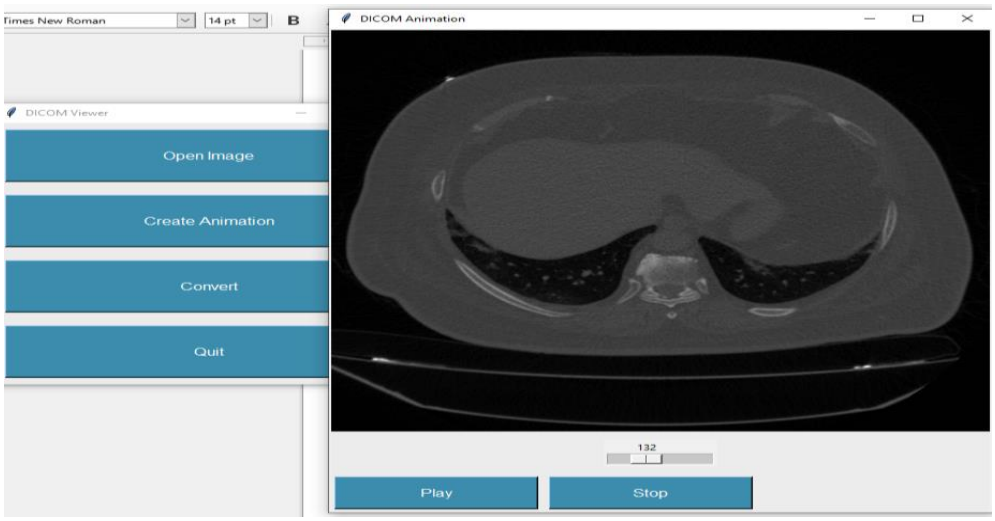

Рисунок 3.5 – Функціональне вікно “Create animation”

Остання наявна функціональна кнопка – це “Convert”. Після роботи з зображенням вона дозволяє зберегти отримані результати у форматі PNG за бажанням.

Вона знадобиться саме для того, щоб зображення можна було адаптувати і відкрити на будь-якому ПК. Цей формат не потребує наявності спеціального програмного забезпечення та може бути опрацьований внутрішніми програмами системи Windows, Linux (рис. 3.6).

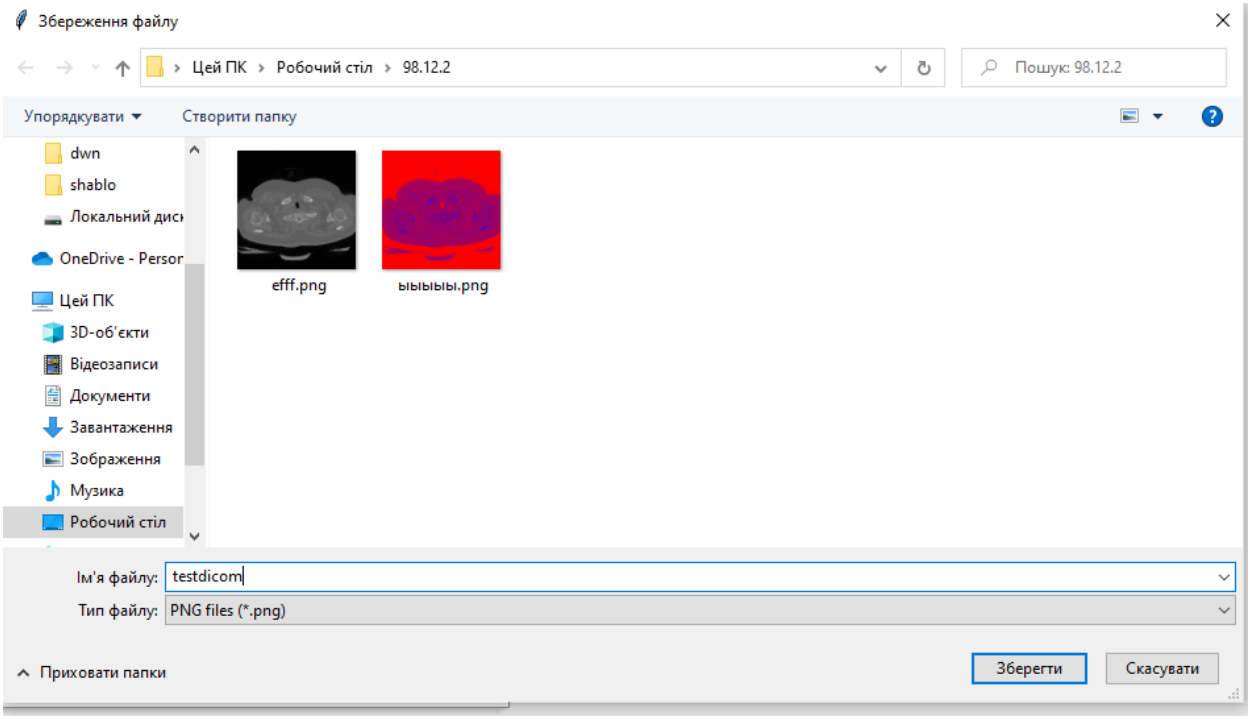

Рисунок 3.6 – Збереження кінцевого зображення у форматі .png

3.5 Висновки до третього розділу

Розділ три демонструє розроблене програмне забезпечення у дії, описує функціонал. Також було взято до уваги та досліджено необхідну для розробки бібліотеку pydicom.

ВИСНОВКИ

У даній кваліфікаційній роботі були досліджені аналогові готові продукти для обробки медичних зображень та розроблено особисте програмне забезпечення. Окрім цього було глибоко досліджено область та її перспективи, а також стан розвитку на даний момент.

Досліджено історію формату Dicom (.dcm) та галузь застосування. Також просторові методи обробки зображень та створення анімації із кількох зображень.

Слід зазначити, що були враховані особливості роботи з медичними зображеннями, їх недоліки та проблеми.

ПЕРЕЛІК ПОСИЛАНЬ

1. Razzak M., Naz S., Zaib A. Deep Learning for Medical Image Processing: Overview, Challenges and the Future. Classification in BioApps. *Part of the Lecture Notes in Computational Vision and Biomechanics book series*. 2017. Vol. 26. P. 323–350. URL: https://link.springer.com/chapter/10.1007/978-3-319-65981-7_12 (дата звернення: 22.03.2023).

2. Gonzalez R., Woods R. Digital Image Processing. Prentice Hall, 2012. 1072 p.

3. Мельник В. М., Муляр Н. В. Автоматична сегментація напівтонових зображень методом ЦДОВ. *Science and Education a New Dimension. Natural and Technical Sciences*. 2018. Vol. 19, Issue 171. С. 43–46.

4. Chollet F. Python documents. URL: https://github.com/fchollet (дата звернення: 22.03.2023).

5. Francisco P.M. OliveiraI, João Manuel R.S. Tavares.Medical image registration: a review. *Computer Methods in Biomechanics and Biomedical Engineering*. 2014. Vol. 17, Issue 2. P. 73–93. URL: https://www.tandfonline.com/doi/abs/10.1080/10255842.2012.670855 (дата звернення: 26.03.2023).

6. Wang J., Zhang M. DeepFLASH: An Efficient Network for Learning-Based Medical Image Registration. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. P. 4443–4451. URL: https://ieeexplore.ieee.org/document/9157045 (дата звернення: 20.03.2023).

7. Fu Y., Lei Y., Wang T., Curran W.J., Liu T., Yang X. Deep learning in medical image registration: a review. *Physics in Medicine & Biology*. Vol. 65, Issue 20. URL: https://iopscience.iop.org/article/10.1088/1361-6560/ab843e (дата звернення: 20.03.2023).

8. Haskins G., Kruger U., Yan P. Deep learning in medical image registration: a survey. *Machine Vision and Applications*. 2020. Vol. 31, Issue 8. P.1–

18. URL: https://link.springer.com/article/10.1007/s00138-020-01060-x (дата звернення: 22.03.2023).

9. Bob D. de Vos, Jelmer M. Wolterink, Pim A. de Jong, Leiner T., Max A. Viergever, Išgum I. ConvNet-Based Localization of Anatomical Structures in 3-D Medical Images. IEEE Transactions on Medical Imaging. 2017. Vol. 36, № 7. P. 1470–1481. URL: https://ieeexplore.ieee.org/document/7862905 (дата звернення: 25.03.2023).

10. Song Y., Cai W., Huang H., Wang Y., Dagan Feng D., Chen M. Region-based progressive localization of cell nuclei in microscopic images with data adaptive modeling. BMC Bioinformatics. 2013. Vol. 14, № 173. URL: https://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-14-173 (дата звернення: 25.03.2023).

11. Sharma H., Sethia J., Bansal P., Gupta S. Feature Extraction and Classification of Chest X-Ray Images Using CNN to Detect Pneumonia. *2020 In: Proceedings of the 10th international conference on cloud computing, data science & engineering(confluence)*. P. 227–231. URL: https://ieeexplore.ieee.org/document/9057809 (дата звернення: 25.03.2023).

12. Abbas A., Abdelsamea M. M., Gaber M. M. Classification of COVID-19 in chest X-ray images using DeTraC deep convolutional neural network. *Applied Intelligence*. 2021. Vol. 51. P. 854–864. URL: https://link.springer.com/article/10.1007/s10489-020-01829-7 (дата звернення: 04.04.2023).

13. Kowsari K., Sali R., Ehsan L., Adorno W., Ali A., Seen Moore, Amadi B., Kelly P., Syed S., Brown D. HMIC: Hierarchical Medical Image Classification, A Deep Learning Approach. Information. 2020. Vol. 11, № 6. P. 318. URL: https://www.mdpi.com/2078-2489/11/6/318 (дата звернення: 04.04.2023).

14. Singh S.P., Wang L., Gupta S., Goli H., Padmanabhan P., Gulyas B. 3D Deep Learning on Medical Images: A Review. Journal «Sensors». 2020. Vol. 20, № 18. P. 5097. URL: https://www.mdpi.com/1424-8220/20/18/5097 (дата звернення: 04.04.2023).

15. Darcy Mason and pydicom contributors. Image processing. URL: https://pydicom.github.io/pydicom/stable/auto_examples/index.html (дата звернення: 04.04.2023).

16. Shen D., Wu G., Suk H. Deep Learning in Medical Image Analysis. Annual Review of Biomedical Engineering. 2017. Vol.19. P. 221–248. URL: https://www.annualreviews.org/doi/10.1146/annurev-bioeng-071516-044442 (дата звернення: 04.04.2023).

17. Wang S.H., Phillips P., Sui Y., Bin L., Yang M., Cheng H. Classification of Alzheimer's Disease Based on Eight-Layer Convolutional Neural Network with Leaky Rectified Linear Unit and Max Pooling. *Journal of Medical Systems*. 2018 Vol. 42, № 85. URL: https://doi.org/10.1007/s10916-018-0932-7 (дата звернення: 04.04.2023).

18. Ravishankar H., Prabhu S.M., Vaidya V., Singhal N. Hybrid approach for automatic segmentation of fetal abdomen from ultrasound images using deep learning. *2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI).* P. 779–782 URL: https://ieeexplore.ieee.org/document/7493382 (дата звернення: 22.03.2023).

19. Suganyadevi S., Shamia D., Balasamy K. An IoT-Based Diet Monitoring Healthcare System for Women. Smart Healthcare System Design: Security and Privacy Aspects. 2021. URL: https://onlinelibrary.wiley.com/doi/10.1002/9781119792253.ch8 (дата звернення: 22.03.2023).

ДОДАТОК А

Реалізація програмного коду

Імпортуємо необхідні бібліотеки для роботи з графічним інтерфейсом, файловою системою, DICOM-зображеннями та зображеннями.

```
import tkinter as tk
from tkinter import filedialog
import pydicom
from PIL import ImageTk, Image, ImageOps
import os
```

Визначаємо кастомний стиль для кнопок, міток та полотна.

```
# Create a custom style for the buttons
BUTTON_STYLE = {
    "font": ("Arial", 12),
    "width": 20,
    "height": 2,
    "bg": "#3C8DAD",
    "fg": "white",
    "activebackground": "#52A8D7",
    "activeforeground": "white",
}

# Create a custom style for the labels
LABEL_STYLE = {
    "font": ("Arial", 14),
    "bg": "#ECECEC",
```

```
    "fg": "#333333",
}

# Create a custom style for the canvas
CANVAS_STYLE = {
    "bg": "#ECECEC",
}
```

Визначаємо функцію open_image(), яка відкриває вибраний користувачем DICOM-файл, обробляє його та створює нове вікно для відображення DICOM-зображення. Зображення відображається на полотні (Canvas) за допомогою мітки (Label).

```
def open_image():
    file_path = filedialog.askopenfilename(filetypes=[("DICOM files", "*.dcm")])
    if file_path:
        # Process the opened DICOM image file
        ds = pydicom.dcmread(file_path)
        pixel_data = ds.pixel_array
        # Rescale pixel values to 8-bit range
        rescaled_image = (pixel_data / pixel_data.max() * 255).astype("uint8")
        image = Image.fromarray(rescaled_image)

        # Create a new window to display the DICOM image
        image_window = tk.Toplevel()
        image_window.title("DICOM Image")

        # Create a canvas to hold the image and tools
        canvas = tk.Canvas(image_window, **CANVAS_STYLE)
        canvas.pack(fill=tk.BOTH, expand=True)
```

```
# Create a label to display the image
tk_image = ImageTk.PhotoImage(image)
image_label = tk.Label(canvas, image=tk_image)
image_label.pack()
```

Визначаємо функцію zoom(), яка забезпечує можливість зуму зображення за допомогою миші.

```
# Create zoom functionality
zoom_factor = 1.0

def zoom(event):
    nonlocal zoom_factor
    if event.delta > 0:
        zoom_factor *= 1.2
    else:
        zoom_factor /= 1.2
    update_image()
```

Визначаємо функції apply_filter() та update_image(), які використовуються для застосування фільтрів до зображення та оновлення відображеного зображення після зуму. Прив'язуємо прокручування миші до функції zoom().

```
def apply_filter(filter_name):
    nonlocal image
    if filter_name == "Invert":
        image = ImageOps.invert(image)
    elif filter_name == "Hot_r":
        image = image.convert("RGB")
```

```
            image = image.convert("L")
            image = ImageOps.colorize(image, "#FFFF00", "#FF0000")
        elif filter_name == "Rainbow":
            image = image.convert("RGB")
            image = image.convert("L")
            image = ImageOps.colorize(image, "#FF0000", "#0000FF")
        elif filter_name == "Jet":
            image = image.convert("RGB")
            image = image.convert("L")
            image = ImageOps.colorize(image, "#0000FF", "#FF0000")
        update_image()

    def clear_filters():
        nonlocal image, filter_buttons
        image = Image.fromarray(rescaled_image)
        # Enable filter buttons
        for button in filter_buttons:
            button.configure(state=tk.NORMAL)
        update_image()

    def update_image():
        nonlocal zoom_factor
        width = int(image.width * zoom_factor)
        height = int(image.height * zoom_factor)
        resized_image = image.resize((width, height))
        tk_image_resized = ImageTk.PhotoImage(resized_image)
        image_label.config(image=tk_image_resized)
        image_label.image = tk_image_resized  # Keep a reference to prevent
garbage collection
```

```
    # Bind mouse wheel events for zooming
    image_label.bind("<MouseWheel>", zoom)

    # Create filter buttons
    filter_frame = tk.Frame(canvas, **CANVAS_STYLE)
    filter_frame.pack(pady=10)

    filters = ["Invert", "Hot_r", "Rainbow", "Jet"]
    filter_buttons = []

    def create_filter_buttons():
        for f in filters:
            filter_button = tk.Button(filter_frame, text=f, command=lambda filter_name=f: apply_filter(filter_name), **BUTTON_STYLE)
            filter_button.pack(side=tk.LEFT, padx=5, pady=5)
            filter_buttons.append(filter_button)

    create_filter_buttons()

    # Create the "Clear" button
    clear_button = tk.Button(filter_frame, text="Clear", command=clear_filters, **BUTTON_STYLE)
    clear_button.pack(side=tk.LEFT, padx=5, pady=5)

    image_window.mainloop()
```

Визначаємо функцію open_folder(), яка відкриває вибрану користувачем папку з DICOM-файлами, обробляє їх та створює нове вікно для відображення анімації з DICOM-зображень. Зображення відображається на полотні (Canvas) за допомогою мітки (Label).

```
def open_folder():
    folder_path = filedialog.askdirectory()
    if folder_path:
        # Process the opened folder containing DICOM files
        print("Opened folder:", folder_path)
        files = os.listdir(folder_path)
        image_list = []
        for file in files:
            if file.endswith(".dcm"):
                ds = pydicom.dcmread(os.path.join(folder_path, file))
                pixel_data = ds.pixel_array
                # Rescale pixel values to 8-bit range
                rescaled_image = (pixel_data / pixel_data.max() * 255).astype("uint8")
                image = Image.fromarray(rescaled_image)
                image_list.append(image)

        if image_list:
            # Create a new window to display the GIF animation
            animation_window = tk.Toplevel()
            animation_window.title("DICOM Animation")

            # Create a canvas to hold the animation and slider
            canvas = tk.Canvas(animation_window, **CANVAS_STYLE)
            canvas.pack(fill=tk.BOTH, expand=True)

            # Create a label to display the animation
            tk_images = [ImageTk.PhotoImage(img) for img in image_list]
            animation_label = tk.Label(canvas, image=tk_images[0])
            animation_label.pack()
```

```
        # Create zoom functionality
        zoom_factor = 1.0

        def zoom(event):
            nonlocal zoom_factor
            if event.delta > 0:
                zoom_factor *= 1.2
            else:
                zoom_factor /= 1.2
            update_image()

        def update_image():
            nonlocal zoom_factor
            index = slider.get()
            image = image_list[index]
            width = int(image.width * zoom_factor)
            height = int(image.height * zoom_factor)
            resized_image = image.resize((width, height))
            tk_image_resized = ImageTk.PhotoImage(resized_image)
            animation_label.config(image=tk_image_resized)
            animation_label.image = tk_image_resized  # Keep a reference to prevent
garbage collection


        # Bind mouse wheel events for zooming
        animation_label.bind("<MouseWheel>", zoom)

        # Create the slider
        slider = tk.Scale(canvas, from_=0, to=len(tk_images) - 1,
orient=tk.HORIZONTAL)
```

```
        slider.pack(pady=10)

        def play_animation():
            global animation_id
            frame_delay = int(1000 / 30)  # Delay between frames (in milliseconds) for 30 fps
            animation_id = animation_window.after(0, lambda: update_animation(frame_delay))

        def update_animation(frame_delay):
            global animation_id
            index = slider.get()
            animation_label.config(image=tk_images[index])
            animation_label.image = tk_images[index]  # Keep a reference to prevent garbage collection
            next_index = (index + 1) % len(tk_images)  # Get the index of the next frame
            slider.set(next_index)  # Update the slider position
            animation_id = animation_window.after(frame_delay, lambda: update_animation(frame_delay))

        def stop_animation():
            global animation_id
            animation_window.after_cancel(animation_id)

        slider.bind("<ButtonRelease-1>", update_animation)
        slider.bind("<B1-Motion>", update_animation)

        # Create the "Play" button
        play_button = tk.Button(canvas, text="Play", command=play_animation,
```

```
**BUTTON_STYLE)
        play_button.pack(side=tk.LEFT, padx=5, pady=5)

        # Create the "Stop" button
        stop_button = tk.Button(canvas, text="Stop", command=stop_animation,
**BUTTON_STYLE)
        stop_button.pack(side=tk.LEFT, padx=5, pady=5)

        animation_window.mainloop()

def convert_image():
   file_path = filedialog.askopenfilename(filetypes=[("DICOM files", "*.dcm")])
   if file_path:
      # Process the opened DICOM image file
      ds = pydicom.dcmread(file_path)
      pixel_data = ds.pixel_array
      # Rescale pixel values to 8-bit range
      rescaled_image = (pixel_data / pixel_data.max() * 255).astype("uint8")
      image = Image.fromarray(rescaled_image)

      # Save the converted image
      save_path = filedialog.asksaveasfilename(defaultextension=".png",
filetypes=[("PNG files", "*.png")])
      if save_path:
         image.save(save_path)



root = tk.Tk()
root.title("DICOM Viewer")
root.geometry("400x400")
```

Створюємо головне вікно програми з міткою для відкриття DICOM-файлу або папки. Запускаємо головний цикл вікна за допомогою root.mainloop().

```
# Create the main frame
main_frame = tk.Frame(root, **CANVAS_STYLE)
main_frame.pack(fill=tk.BOTH, expand=True)

# Create the "Open Image" button
open_image_button = tk.Button(main_frame, text="Open Image", command=open_image, **BUTTON_STYLE)
open_image_button.pack(side=tk.TOP, padx=10, pady=10, fill=tk.BOTH, expand=True)
open_image_button.bind("<Alt-i>", lambda event: open_image())  # Bind the shortcut Alt+i to the button

# Create the "Create Animation" button
open_folder_button = tk.Button(main_frame, text="Create Animation", command=open_folder, **BUTTON_STYLE)
open_folder_button.pack(side=tk.TOP, padx=10, pady=10, fill=tk.BOTH, expand=True)
open_folder_button.bind("<Alt-a>", lambda event: open_folder())  # Bind the shortcut Alt+a to the button

# Create the "Convert" button
convert_button = tk.Button(main_frame, text="Convert", command=convert_image, **BUTTON_STYLE)
convert_button.pack(side=tk.TOP, padx=10, pady=10, fill=tk.BOTH, expand=True)
convert_button.bind("<Alt-c>", lambda event: convert_image()) # Bind the shortcut
```

Alt+c to the button

```
# Create the "Quit" button
quit_button = tk.Button(main_frame, text="Quit", command=root.quit, **BUTTON_STYLE)
quit_button.pack(side=tk.TOP, padx=10, pady=10, fill=tk.BOTH, expand=True)
quit_button.bind("<Alt-q>", lambda event: root.quit())  # Bind the shortcut Alt+q to the button

# Configure the weights of the main frame's children to make them resize with the window
main_frame.columnconfigure(0, weight=1)
main_frame.rowconfigure(0, weight=1)
main_frame.rowconfigure(1, weight=1)
main_frame.rowconfigure(2, weight=1)
main_frame.rowconfigure(3, weight=1)

root.mainloop()
```