Минобрнауки России

Юго-Западный государственный университет

Кафедра программной инженерии

ОТЧЕТ

о преддипломной (производственной) практике

наименование вида и типа практики

на (в) ООО ”МЦОБ ОНЛАЙН-СЕРВИСЫ”

наименование предприятия, организации, учреждения

Студента 5курса, группы ПО-12з

курса, группы

Елфимова Василия Ивановича

фамилия, имя, отчество

Руководитель практики от предприятия, организации, учреждения

Оценка _____

_____

должность, звание, степень

_____ _____

фамилия и. о. подпись, дата

Руководитель практики от университета

Оценка _____

к.т.н. доцент

должность, звание, степень

Чаплыгин А. А. _____

фамилия и. о. подпись, дата

Члены комиссии

_____ _____

подпись, дата фамилия и. о.

_____ _____

подпись, дата фамилия и. о.

_____ _____

подпись, дата фамилия и. о.

Курск 2026 г.

СОДЕРЖАНИЕ

ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

БД – база данных.

ИС – информационная система.

ИТ – информационные технологии.

КТС – комплекс технических средств.

ОМТС – отдел материально-технического снабжения.

ПО – программное обеспечение.

РП – рабочий проект.

СУБД – система управления базами данных.

ТЗ – техническое задание.

ТП – технический проект.

UML (Unified Modelling Language) – язык графического описания для объектного моделирования в области разработки программного обеспечения.

Python – язык программирования

1 Анализ предметной области

1.1 Характеристика предприятия и его деятельности

Предприятие, для которого осуществляется разработка программного обеспечения, является разработчиком программных решений для оптимизации работы с цифровыми данными. Основное направление деятельности компании - создание инструментов для повышения эффективности обработки, хранения и передачи информации. В условиях цифровой трансформации и постоянного роста объемов данных, с которыми работают как корпоративные клиенты, так и частные пользователи, компания фокусируется на разработке решений для управления цифровым контентом.

Деятельность предприятия характеризуется следующими аспектами:

– Разработка кроссплатформенного программного обеспечения с графическим интерфейсом;

– Создание инструментов для оптимизации использования дискового пространства;

– Предоставление решений для резервного копирования и архивирования данных;

– Разработка утилит для обработки мультимедийного контента.

1.2 Анализ проблемной ситуации и постановка задачи

В современной цифровой среде пользователи и организации сталкиваются с рядом проблем при работе с файлами и изображениями:

– Растущие объемы данных:С развитием технологий значительно увеличились размеры файлов, особенно мультимедийного контента (фотографии, видео), что приводит к быстрому заполнению дискового пространства;

– Неэффективное хранение информации: Многие пользователи хранят файлы в исходном формате, не используя возможности сжатия, что приводит к избыточному расходу места на накопителях;

– Сложность организации файлов: Отсутствие удобных инструментов для групповой обработки файлов и создания архивов усложняет структурирование данных и передачу множества файлов;

– Несовместимость форматов: Разнообразие форматов изображений и архивов создает сложности при выборе оптимального метода сжатия для конкретных задач;

– Отсутствие комплексного решения: Существующие утилиты часто решают только одну задачу - либо сжатие изображений, либо работа с архивами, что требует использования нескольких программ.

Для решения этих проблем была поставлена задача разработать универсальную программу, которая объединит следующие возможности:

– Сжатие изображений с сохранением визуального качества;

– Создание архивов в формате (ZIP, TAR.GZ, TAR.BZ2);

– Распаковка существующих архивов;

– Пакетная обработка файлов;

– Интуитивно понятный графический интерфейс;

– Отслеживание прогресса операций.

1.3 Обзор существующих систем и обоснование выбора технологий

Анализ существующих решений на рынке показал наличие следующих категорий программ:

1. Специализированные архиваторы (WinRAR, 7-Zip, WinZip):

– Преимущества: Высокая степень сжатия, поддержка множества форматов;

– Недостатки: Ограниченные возможности работы с изображениями, часто платные лицензии.

2. Редакторы изображений со сжатием (Adobe Photoshop, GIMP):

– Преимущества: Качественная обработка изображений;

– Недостатки: Избыточный функционал для простого сжатия, высокая стоимость, требовательность к ресурсам.

3. Онлайн-сервисы сжатия:

– Преимущества: Доступность, не требуют установки;

– Недостатки: Ограничения на размер файлов, необходимость подключения к интернету, вопросы конфиденциальности данных.

4. Утилиты командной строки:

– Преимущества: Мощные возможности, автоматизация.

– Недостатки: Сложность использования для неподготовленных пользователей, отсутствие графического интерфейса.

1.3.1 Обоснование выбора технологий для разработки:

Для реализации поставленной задачи был выбран язык программирования Python со следующими библиотеками:

1. Tkinter - для создания кроссплатформенного графического интерфейса

– Нативный фреймворк Python;

– Не требует дополнительных зависимостей;

– Поддержка всех основных операционных систем.

2. Pillow (PIL) - для работы с изображениями

– Широкая поддержка форматов (JPEG, PNG, WebP и др.);

– Возможность тонкой настройки параметров сжатия;

– Сохранение метаданных изображений.

3. Стандартные библиотеки Python для работы с архивами:

– zipfile - поддержка ZIP форматов;

– tarfile - работа с TAR архивами;

– gzip, bz2 - алгоритмы сжатия.

4. Многопоточность (модуль threading):

– Обеспечение отзывчивости интерфейса во время длительных операций;

– Возможность отмены операций;

– Параллельная обработка файлов.

Такой выбор технологий позволяет создать легковесное, кроссплатформенное приложение с богатым функционалом, не требующее значительных вычислительных ресурсов и способное работать автономно без подключения к интернету.

2 Техническое задание

2.1 Основание для разработки

Полное наименование системы: «Разработка программы для сжатия файлов с графическим интерфейсом».

Основанием для разработки программы является приказ ректора ЮЗГУ от « » 2025 г. № «Об утверждении тем выпускных квалификационных работ» на тему «Разработка программы для сжатия файлов».

2.2 Цель и назначение разработки

Основной целью выпускной квалификационной работы является разработка многофункционального программного обеспечения для эффективного сжатия и архивирования файлов с поддержкой обработки изображений.

Программа предназначена для решения следующих задач пользователей:

– Сжатие изображений с настраиваемыми параметрами качества для уменьшения объема фотографий и графических файлов без значительной потери визуального качества.

– Создание архивов в популярных форматах (ZIP, TAR.GZ, TAR.BZ2) для объединения и сжатия множества файлов и папок.

– Распаковка существующих архивов с поддержкой защищенных паролем архивов.

– Пакетная обработка файлов и папок для массового сжатия и архивирования.

– Экономия дискового пространства за счет применения оптимальных алгоритмов сжатия.

Задачи разработки:

– разработка модуля сжатия изображений с использованием библиотеки Pillow;

– реализация модуля работы с архивами (создание, извлечение) через стандартные библиотеки Python;

– создание графического интерфейса на базе Tkinter с вкладками для различных операций;

– реализация системы отслеживания прогресса выполнения операций;

– разработка системы настроек и сохранения пользовательских параметров;

– обеспечение многопоточности для сохранения отзывчивости интерфейса;

– создание системы логгирования операций;

– реализация предпросмотра изображений перед сжатием.

2.3 Актуальность темы разработки

В эпоху цифровизации и роста объемов данных проблема эффективного хранения и передачи информации становится все более актуальной. С развитием высококачественных мультимедийных форматов значительно увеличились размеры файлов, что создает трудности при их хранении на локальных устройствах, передаче по сети и использовании в облачных сервисах.

Актуальность разработки программы для сжатия файлов обусловлена следующими факторами:

– Рост объемов данных – пользователи ежедневно создают и получают большие объемы файлов, особенно изображений и документов;

– Ограниченность ресурсов хранения – несмотря на снижение стоимости носителей информации, потребность в эффективном использовании дискового пространства остается высокой;

– Требования к передаче файлов – при отправке файлов по электронной почте или через мессенджеры часто существуют ограничения на размер вложений;

– Необходимость архивирования – организация и сжатие множества связанных файлов в единый архив упрощает их хранение и передачу;

– Отсутствие комплексных бесплатных решений – многие существующие программы либо платные, либо решают только часть задач, либо сложны в использовании для обычных пользователей.

Разрабатываемая программа актуальна, так как объединяет в одном интерфейсе функции сжатия изображений и работы с архивами, предоставляя пользователю универсальный инструмент для оптимизации хранения данных.

2.4 Требования пользователя к интерфейсу приложения

Приложение должно включать в себя следующие основные элементы интерфейса:

Вкладка ”Сжатие файлов”для работы с архивами:

– Поле для выбора файлов и папок;
– Настройки формата архива и уровня сжатия;
– Поле для ввода пароля (опционально);
– Указание имени выходного файла;
– Панель прогресса выполнения операции.

Вкладка ”Распаковка архивов”для извлечения файлов из архивов:

– Поле для выбора архива;
– Информация о содержимом архива;
– Указание папки для распаковки;
– Поле для ввода пароля (если требуется);
– Панель прогресса распаковки.

Вкладка ”Сжатие изображений”(при наличии библиотеки Pillow):

– Поле для выбора изображения;
– Предпросмотр изображения;
– Настройки качества сжатия;
– Выбор формата выходного файла;
– Пакетное сжатие изображений из папки;

Вкладка ”Настройки”для конфигурации программы:

– Настройка пути сохранения по умолчанию;
– Параметры сжатия по умолчанию;
– Настройки логгирования и уведомлений;

Область логов для отображения информации о выполненных операциях

Статус-бар для индикации текущего состояния программы

Все элементы управления должны быть достаточно крупными для удобного использования, с понятными метками и интуитивно понятным расположением.

Композиция основного экрана приложения представлена на рисунке 2.1.

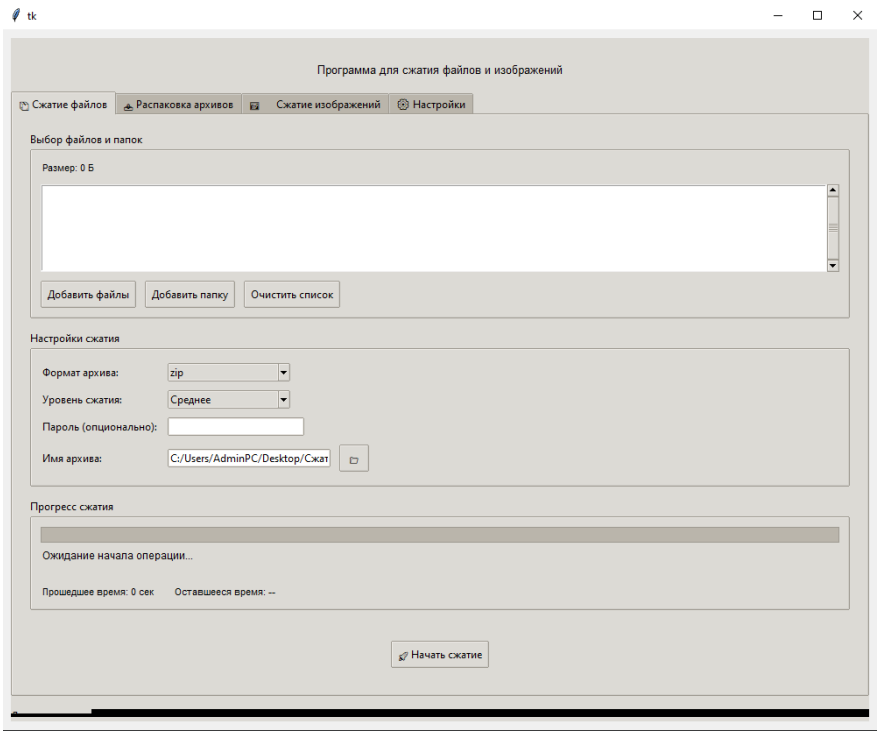

Рисунок 2.1 – Основной экран приложения

2.5 Моделирование вариантов использования

Для разрабатываемой программы была построена диаграмма вариантов использования UML, которая описывает функциональное назначение системы и взаимодействие пользователя с программой.

Актер: Пользователь (физическое лицо, нуждающееся в сжатии файлов или изображений)

Варианты использования

1. Сжать файлы в архив;

2. Распаковать архив;
3. Сжать изображение;
4. Сжать все изображения в папке;
5. Настроить параметры программы;
6. Просмотреть логи операций;

На рисунке 2.2 изображены прецеденты для покупателя.

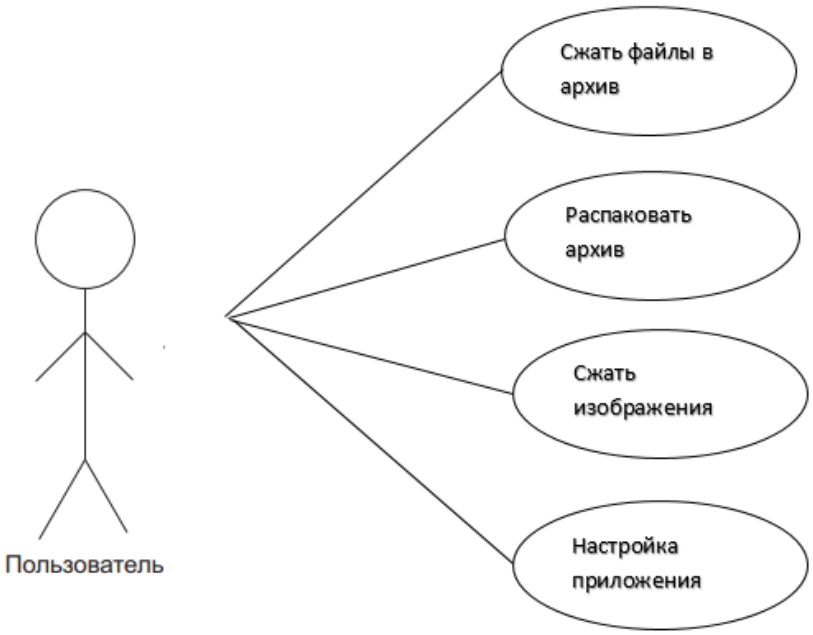

Рисунок 2.2 – Прецеденты для пользователя

2.6 Функциональные требования (сценарии прецедентов)

2.6.1 Сценарий прецедента «Сжать файлы в архив»

Предусловие: Программа запущена, открыта вкладка «Сжатие файлов».

Основной успешный сценарий:

1. Пользователь нажимает кнопку «Добавить файлы» или «Добавить папку»;
2. Выбирает файлы и/или папки для архивирования;

3. Выбирает формат архива (ZIP, TAR.GZ, TAR.BZ2);

4. Устанавливает уровень сжатия;

5. При необходимости вводит пароль для архива;

6. Указывает имя и путь для выходного файла;

7. Нажимает кнопку «Начать сжатие»;

8. Система отображает прогресс выполнения операции;

9. По завершении операция отображается сообщение об успешном создании архива.

2.6.2 Сценарий прецедента «Распаковать архив»

Предусловие: Открыта вкладка «Распаковка архивов».

Основной успешный сценарий:

1. Пользователь нажимает кнопку выбора архива;

2. Выбирает архивный файл;

3. Система автоматически показывает информацию о содержимом архива;

4. Пользователь указывает папку для распаковки;

5. При необходимости вводит пароль для доступа к архиву;

6. Нажимает кнопку «Распаковать»;

7. Система отображает прогресс распаковки;

8. По завершении операция отображается сообщение об успешной распаковке.

2.6.3 Сценарий прецедента «Сжать изображение»

Предусловие: Установлена библиотека Pillow, открыта вкладка «Сжатие изображений». **Основной успешный сценарий:**

1. Пользователь выбирает изображение для сжатия;

2. Система отображает предпросмотр изображения и его характеристики;

3. Пользователь устанавливает уровень качества сжатия;

4. Выбирает формат выходного файла;

5. Указывает имя и путь для сохранения сжатого изображения;

6. Нажимает кнопку «Сжать изображение»;

7. Система выполняет сжатие и отображает результат (исходный и конечный размер).

2.6.4 Сценарий прецедента «Сжать все изображения в папке»

Предусловие: Установлена библиотека Pillow.

Основной успешный сценарий:

1. Пользователь выбирает папку с изображениями;
2. Устанавливает параметры сжатия;
3. Нажимает кнопку «Сжать все изображения в папке»;
4. Система создает новую папку для результатов;
5. Последовательно обрабатывает все изображения в исходной папке;
6. Отображает прогресс обработки каждого файла;
7. По завершении выводит сводную информацию о результатах.

2.6.5 Сценарий прецедента «Настроить параметры программы»

Предусловие: Открыта вкладка «Настройки».

Основной успешный сценарий:

1. Пользователь изменяет путь сохранения по умолчанию;
2. Настраивает качество изображений по умолчанию;
3. Включает/выключает сохранение логов операций;
4. Настраивает отображение уведомлений;
5. Нажимает кнопку «Сохранить настройки»;
6. Система сохраняет настройки в файл конфигурации.

2.6.6 Сценарий прецедента «Установить библиотеку Pillow»

Предусловие: Библиотека Pillow не установлена.

Основной успешный сценарий:

1. Система отображает предупреждение об отсутствии Pillow;
2. Пользователь нажимает кнопку «Установить Pillow»;

3. Система запускает процесс установки через pip;

4. По завершении установки выводится сообщение об успехе;

5. Требуется перезапуск программы для активации функций работы с изображениями.

2.7 Требования пользователя к интерфейсу приложения

Приложение должно содержать следующие экраны и элементы управления:

1. Главное окно программы:

– Заголовок с названием программы и версией;

– Область вкладок для переключения между режимами работы;

– Область логов выполнения операций;

– Статус-бар с информацией о текущем состоянии.

2. Вкладка ”Сжатие файлов”:

– Список выбранных файлов и папок с возможностью очистки;

– Выпадающий список выбора формата архива;

– Выпадающий список уровня сжатия;

– Поле ввода пароля с возможностью скрытия символов;

– Поле указания имени выходного файла с кнопкой выбора пути;

– Прогресс-бар и информация о текущей операции;

– Кнопка запуска операции сжатия.

3. Вкладка ”Распаковка архивов”:

– Поле выбора архива с кнопкой обзора;

– Область информации о содержимом архива;

– Поле указания папки распаковки с кнопкой выбора;

– Поле ввода пароля для защищенных архивов;

– Прогресс-бар распаковки;

– Кнопки просмотра информации и распаковки.

4. Вкладка ”Сжатие изображений”:

– Поле выбора изображения с предпросмотром;

– Слайдер настройки качества с отображением значения;

– Выпадающий список формата выходного файла;
– Поле указания имени выходного файла;
– Панель пакетной обработки изображений;
– Прогресс-бар операций с изображениями.

5. Вкладка ”Настройки”:

– Настройка пути сохранения по умолчанию;
– Настройки качества изображений по умолчанию;
– Настройки логгирования и уведомлений;
– Информация о системе и установленных библиотеках;
– Кнопки сохранения и сброса настроек.

6. Область логов:

– Текстовое поле с прокруткой для отображения логов;
– Кнопки управления логами (очистка, копирование, сохранение);
– Автоматическая прокрутка к новым сообщениям.

2.8 Требования к оформлению документации

Разработка программной документации и программного изделия должна производиться согласно ГОСТ 19.102-77 и ГОСТ 34.601-90 «Единая система программной документации».

3 Технический проект

3.1 Общая характеристика организации решения задачи

Необходимо спроектировать и разработать универсальную программу для сжатия файлов и изображений с графическим пользовательским интерфейсом. Приложение должно предоставлять комплексный набор инструментов для оптимизации хранения цифровых данных, включая сжатие изображений, создание архивов в различных форматах и распаковку существующих архивов.

Программа представляет собой кроссплатформенное настольное приложение, написанное на языке Python с использованием библиотеки Tkinter для графического интерфейса. Для обработки изображений применяется библиотека Pillow (PIL), а для работы с архивами используются стандартные библиотеки Python (zipfile, tarfile, gzip, bz2). Приложение реализует многопоточность для сохранения отзывозивости интерфейса во время длительных операций сжатия и включает систему отслеживания прогресса выполнения операций.

Решение разрабатывается с учетом требований различных категорий пользователей: от обычных пользователей, нуждающихся в простом инструменте для уменьшения размера фотографий, до более продвинутых пользователей, требующих возможностей создания защищенных архивов и пакетной обработки файлов. Архитектура приложения построена по модульному принципу, где каждый компонент отвечает за определенный аспект функциональности, что упрощает сопровождение, тестирование и расширение системы в будущем.

3.2 Обоснование выбора технологии проектирования

Для реализации поставленной задачи был выбран стек технологий, обеспечивающий кроссплатформенность, простоту развертывания, минимальные требования к системным ресурсам и возможность быстрой разработки. Выбор конкретных технологических решений осуществлялся на осно-

ве анализа требований к функциональности, производительности и удобству использования приложения.

При проектировании системы учитывались следующие факторы:

– Необходимость работы на различных операционных системах (Windows, Linux, macOS) без модификации кода;

– Минимальные требования к установке зависимостей для конечного пользователя;

– Возможность обработки больших объемов данных без заморозки интерфейса;

– Простота использования для неподготовленных пользователей;

– Возможность расширения функциональности в будущем.

Выбранный технологический стек позволяет удовлетворить все перечисленные требования при сохранении приемлемой сложности реализации и обеспечивает возможность дальнейшего развития приложения.

3.2.1 Описание используемых технологий и языков программирования

3.2.1.1 Язык программирования Python

Python — интерпретируемый язык программирования высокого уровня с динамической типизацией и автоматическим управлением памятью. Был выбран в качестве основного языка разработки благодаря совокупности преимуществ:

1. Богатая стандартная библиотека — включает модули для работы с архивами (zipfile, tarfile, gzip, bz2), файловой системой, многопоточностью и JSON, что значительно сокращает объем кода, который необходимо писать с нуля;

2. Кроссплатформенность — приложение может работать на Windows, Linux и macOS без существенной модификации кода, что важно для охвата широкой аудитории пользователей;

3. Простота разработки и сопровождения — чистый синтаксис и ясная структура кода облегчают разработку, отладку и дальнейшее сопровождение программы;

4. Широкое сообщество и обилие документации — наличие многочисленных ресурсов, примеров кода и активного сообщества разработчиков ускоряет решение возникающих проблем;

5. Поддержка объектно-ориентированного программирования — позволяет создавать модульную, расширяемую архитектуру с четким разделением ответственности между компонентами;

6. Возможность создания исполняемых файлов — с использованием инструментов вроде PyInstaller или cx Freeze можно создать standalone-приложение, не требующее установки Python на целевом компьютере.

3.2.1.2 Библиотека Tkinter

Tkinter — стандартная библиотека Python для создания графических интерфейсов, основанная на Tk/Tcl. Выбор Tkinter обусловлен следующими соображениями:

1. Встроенность в Python — не требует дополнительной установки, что упрощает развертывание приложения;

2. Кроссплатформенность — обеспечивает одинаковое поведение интерфейса на разных операционных системах с адаптацией к нативному виду каждой платформы;

3. Простота использования — относительно низкий порог вхождения по сравнению с другими GUI-фреймворками, что ускоряет разработку;

4. Достаточная функциональность — предоставляет все необходимые виджеты для создания интуитивно понятного интерфейса: кнопки, поля ввода, выпадающие списки, фреймы, вкладки, прогресс-бары;

5. Поддержка ttk (Themed Tk) — позволяет создавать более современные интерфейсы с улучшенным внешним видом элементов;

6. Хорошая документация и множество примеров — облегчает решение типовых задач при разработке интерфейса.

3.2.1.3 Библиотека Pillow (PIL)

Pillow — форк библиотеки Python Imaging Library (PIL), предоставляющей расширенные возможности обработки изображений. Используется для сжатия изображений в программе:

1. Широкая поддержка форматов — JPEG, PNG, WebP, BMP, TIFF, GIF и другие популярные форматы изображений;
2. Гибкие параметры сжатия — позволяет точно настраивать качество, степень сжатия, оптимизацию и другие параметры сохранения изображений;
3. Сохранение метаданных — поддерживает сохранение EXIF-данных и другой метаинформации при сжатии;
4. Простотой API — интуитивно понятные методы для открытия, обработки и сохранения изображений;
5. Активное развитие — регулярные обновления, поддержка современных форматов и алгоритмов;
6. Хорошая производительность — эффективная реализация алгоритмов обработки изображений на C.

3.2.1.4 Стандартные библиотеки Python для работы с архивами

Для работы с архивами используются стандартные библиотеки Python:

1. zipfile — для создания и извлечения ZIP-архивов с поддержкой различных алгоритмов сжатия и шифрования;
2. tarfile — для работы с TAR-архивами, которые могут комбинироваться с gzip или bz2 сжатием;
3. gzip и bz2 — модули для сжатия данных алгоритмами GZIP и BZIP2 соответственно.

Преимущества использования стандартных библиотек:

– Не требуют дополнительной установки;
– Хорошо протестированы и надежны;
– Полная интеграция с экосистемой Python;
– Поддерживают все необходимые операции с архивами.

3.2.1.5 Многопоточность (threading)

Модуль threading стандартной библиотеки Python используется для реализации многопоточности:

1. Отзывчивость интерфейса — позволяет выполнять длительные операции сжатия/распаковки в отдельных потоках, не блокируя основной поток интерфейса;

2. Возможность отмены операций — управление потоками позволяет реализовать механизм отмены выполнения операций;

3. Параллельная обработка — возможность одновременной обработки нескольких файлов (в будущих версиях);

4. Простота реализации — по сравнению с multiprocessing, threading проще в реализации для I/O-bound операций.

3.2.1.6 JSON для хранения настроек

Формат JSON используется для хранения пользовательских настроек:

1. Человекочитаемость — позволяет пользователям при необходимости вручную редактировать файл настроек;

2. Простота сериализации — встроенная поддержка в Python через модуль json;

3. Компактность — достаточно эффективное представление данных;

4. Универсальность — широко поддерживаемый формат, легко читаемый другими программами.

3.3 Диаграмма компонентов и схема обмена данными между файлами компонента

Диаграмма компонентов описывает архитектурные компоненты разрабатываемой системы и их взаимодействие. Основными компонентами являются:

1. **Графический интерфейс (GUI)** — отвечает за отображение информации, обработку пользовательского ввода и предоставление визуальной об-

ратной связи. Состоит из нескольких вкладок, каждая из которых отвечает за определенный тип операций;

2. **Менеджер сжатия изображений (ImageCompressor)** — реализует алгоритмы сжатия изображений с использованием библиотеки Pillow. Поддерживает различные форматы и параметры качества;

3. **Менеджер работы с архивами (FileCompressor)** — отвечает за создание и распаковку архивов в различных форматах (ZIP, TAR.GZ, TAR.BZ2). Реализует поддержку паролей и различных уровней сжатия;

4. **Трекер прогресса (ProgressTracker)** — отслеживает ход выполнения операций, рассчитывает оставшееся время и предоставляет информацию для отображения прогресса в интерфейсе;

5. **Менеджер настроек (SettingsManager)** — управляет сохранением и загрузкой пользовательских настроек в формате JSON;

6. **Логгер операций** — записывает информацию о выполненных операциях в лог и предоставляет интерфейс для их просмотра и сохранения;

7. **Менеджер многопоточности** — координирует выполнение операций в отдельных потоках для сохранения отзывчивости интерфейса.

Каждый компонент разработан с соблюдением принципа единственной ответственности. Взаимодействие между компонентами организовано через четко определенные интерфейсы, что снижает степень связанности и повышает модульность архитектуры.

На рисунке 3.1 изображена диаграмма компонентов системы, демонстрирующая взаимосвязи между основными модулями приложения и внешними системами.

Диаграмма наглядно показывает, что архитектура системы построена по многоуровневому принципу. Графический интерфейс взаимодействует с контроллером операций, который, в свою очередь, делегирует выполнение конкретных задач специализированным модулям (ImageCompressor, FileCompressor). Трекер прогресса используется всеми модулями для инфор-

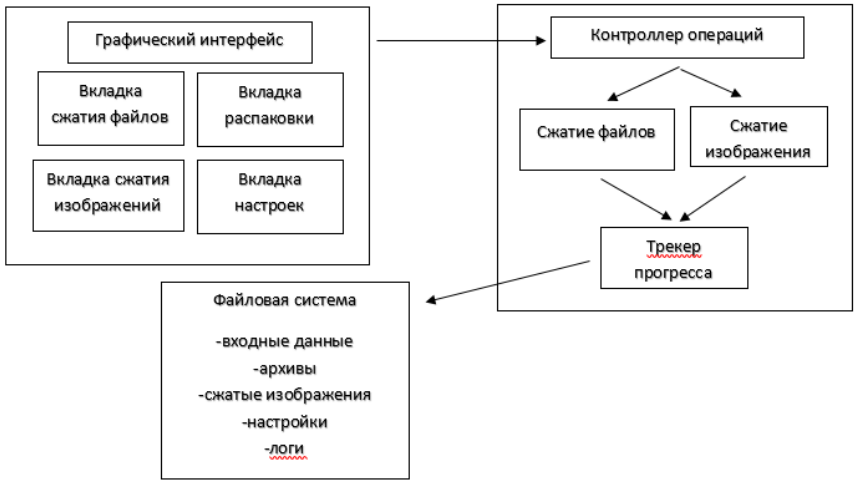

Рисунок 3.1 – Диаграмма компонентов программы для сжатия файлов

мирования о ходе выполнения операций. Все компоненты работают с файловой системой для чтения входных данных и записи результатов.

3.4 Диаграмма размещения

Диаграмма размещения отражает физическую архитектуру системы и показывает, как программные компоненты распределены по аппаратным узлам.

Система является standalone-приложением и развертывается на компьютере пользователя. Все компоненты выполняются в рамках одного процесса:

1. **Клиентское устройство** — компьютер пользователя (Windows, Linux или macOS), на котором установлена программа. Все компоненты приложения выполняются на этом устройстве.

2. **Файловая система устройства** — используется для:

– Хранения входных файлов для обработки

– Сохранения результатов сжатия и архивов

– Хранения файла настроек программы

– Записи логов операций

3. **Оперативная память** — используется для:

– Загрузки изображений при предпросмотре

– Буферизации данных при работе с архивами

– Временного хранения промежуточных результатов

4. **Внешние зависимости** (устанавливаются при необходимости):

– Библиотека Pillow (для работы с изображениями)

– Python интерпретатор (если программа распространяется как исходный код)

На рисунке 3.2 представлена диаграмма размещения системы, иллюстрирующая физическое распределение компонентов и направления обмена данными между ними.

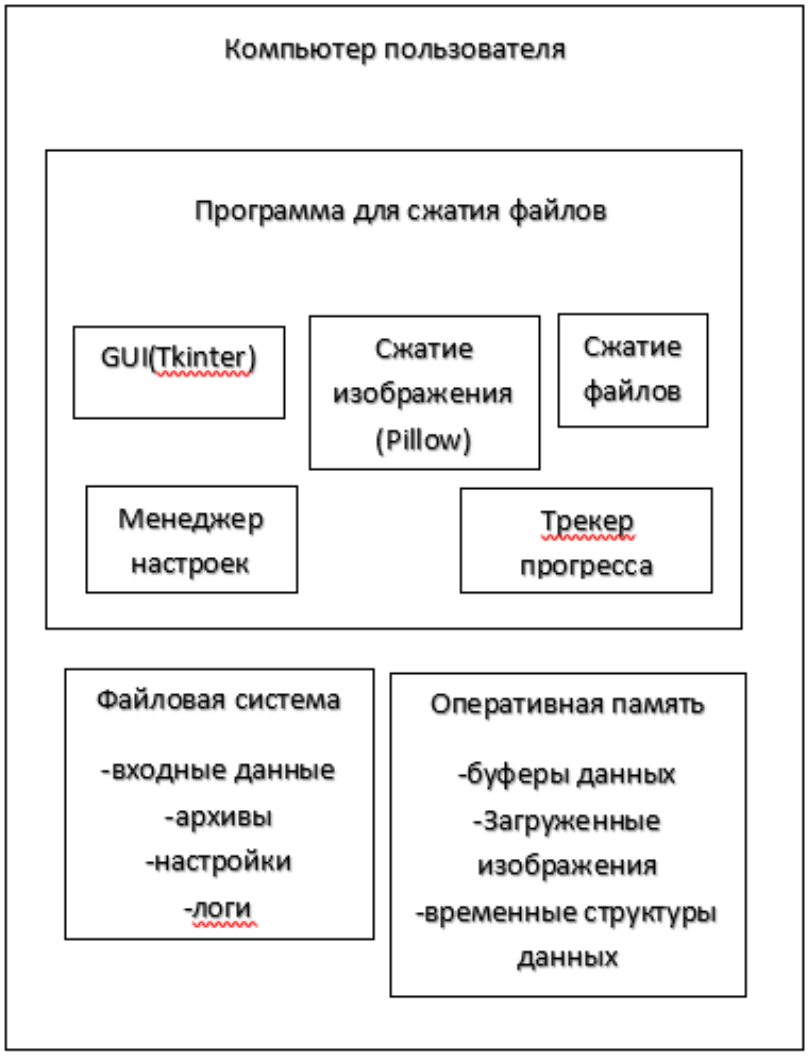

Рисунок 3.2 – Диаграмма размещения системы

Особенностью архитектуры размещения является ее самодостаточность — приложение не требует подключения к интернету или доступа к внешним сервисам для своей работы (за исключением установки библиотеки

Pillow при первом запуске). Все операции выполняются локально на компьютере пользователя, что обеспечивает:

– Конфиденциальность данных (файлы не передаются на внешние серверы)

– Работу в офлайн-режиме

– Высокую скорость обработки (нет задержек на передачу данных по сети)

– Независимость от доступности внешних сервисов

3.5 Содержание данных. Основные сущности

Проанализировав требования к системе и особенности работы с файлами и изображениями, можно выделить следующие основные сущности системы:

– Изображение (Image) — сущность, представляющая графический файл, подлежащий сжатию. Содержит информацию о файле изображения и параметрах его обработки;

– Архив (Archive) — сущность, представляющая архивный файл, создаваемый или обрабатываемый программой. Включает информацию о формате архива, параметрах сжатия и списке файлов;

– Операция (Operation) — сущность, описывающая процесс сжатия или распаковки. Содержит информацию о ходе выполнения, прогрессе и результатах;

– Настройки программы (Settings) — конфигурация приложения, хранящая пользовательские предпочтения и параметры по умолчанию.

Каждая сущность обладает определенным набором атрибутов, которые описывают ее свойства и характеристики. Атрибуты выбраны с учетом требований к функциональности приложения и удобству работы пользователя.

В таблице 3.1 представлены атрибуты сущности «Изображение», которые используются при обработке графических файлов.

Таблица 3.1 – Атрибуты сущности «Изображение»

Поле	Тип	Обязательное	Описание
1	2	3	4
path	String	Да	Путь к файлу изображения в файловой системе
format	String	Да	Формат изображения (JPEG, PNG, WebP, BMP, TIFF, GIF)
width	Integer	Да	Ширина изображения в пикселях
height	Integer	Да	Высота изображения в пикселях
size	Integer	Да	Размер файла изображения в байтах
quality	Integer	Нет	Уровень качества для сжатия (0-100)
output_format	String	Нет	Формат выходного файла после сжатия
output_path	String	Нет	Путь для сохранения сжатого изображения

В таблице 3.2 представлены атрибуты сущности «Архив», которые используются при создании и обработке архивных файлов.

Таблица 3.2 – Атрибуты сущности «Архив»

Поле	Тип	Обязательное	Описание
1	2	3	4
format	ObjectId	Да	Формат архива (ZIP, TAR.GZ, TAR.BZ2)
compression_level	String	Да	Уровень сжатия (0-9, где 0 - без сжатия, 9 - максимальное)

Продолжение таблицы 3.2

1	2	3	4
password	String	Нет	Пароль для защиты архива (опционально)
files	Date	Да	Список путей к файлам и папкам для архивирования
output_path	String	Да	Путь для сохранения создаваемого архива
total_size	Integer	Нет	Общий размер всех файлов в архиве
compressed_size	Integer	Нет	Размер архива после сжатия

Сущности реализуются в виде классов Python, а их атрибуты — в виде свойств этих классов. Взаимодействие между сущностями осуществляется через методы соответствующих модулей. Например, класс ImageCompressor работает с сущностью Image, FileCompressor — с сущностью Archive, а ProgressTracker — с сущностью Operation.

Выделенные сущности и их атрибуты образуют информационную модель приложения, которая служит основой для разработки модулей работы с данными, пользовательского интерфейса и бизнес-логики системы. Данная модель спроектирована с учетом как текущих функциональных требований, так и возможных направлений развития системы в будущем.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Лутц, М. Изучаем Python / М. Лутц. – 5-е изд. – Санкт-Петербург : Символ-Плюс, 2022. – 1648 с. – ISBN 978-5-93286-256-6. – Текст: непосредственный.

2. Саммерфилд, М. Программирование на Python 3. Подробное руководство / М. Саммерфилд. – Санкт-Петербург : Символ-Плюс, 2019. – 608 с. – ISBN 978-5-93286-211-5. – Текст: непосредственный.

3. Россум, Г. Язык программирования Python / Г. ван Россум, Ф. Л. Дрейк. – Москва : ДМК Пресс, 2020. – 454 с. – ISBN 978-5-93700-107-1. – Текст: непосредственный.

4. Крокфорд, Д. JSON: Основы / Д. Крокфорд. – Москва : СимволПлюс, 2017. – 128 с. – ISBN 978-5-93286-213-9. – Текст: непосредственный.

5. Купер, А. Интерфейс: основы проектирования взаимодействия / А. Купер, Р. Рейман, Д. Кронин. – 4-е изд. – Санкт-Петербург : Питер, 2021. – 720 с. – ISBN 978-5-4461-1231-3. – Текст: непосредственный.

6. Тидвелл, Д. Разработка пользовательских интерфейсов / Д. Тидвелл. – 2-е изд. – Санкт-Петербург : Питер, 2020. – 528 с. – ISBN 978-5-4461-1152-1. – Текст: непосредственный.

7. Таненбаум, Э. С. Распределенные системы. Принципы и парадигмы / Э. С. Таненбаум, М. ван Стеен. – 2-е изд. – Санкт-Петербург : Питер, 2020. – 1120 с. – ISBN 978-5-4461-1456-0. – Текст: непосредственный.

8. Соммервилл, И. Инженерия программного обеспечения / И. Соммервилл. – 10-е изд. – Москва : Вильямс, 2019. – 736 с. – ISBN 978-5-8459- 2077-5. – Текст: непосредственный.

9. Куликов, С. Ю. Тестирование программного обеспечения / С. Ю. Куликов. – Москва : Бином. Лаборатория знаний, 2018. – 280 с. – ISBN 978-5-9963-3604-2. – Текст: непосредственный.

10. Material Design 3. Руководство по дизайну [Электронный ресурс]. – Режим доступа: https://m3.material.io/ (дата обращения: 15.12.2024). – Текст: электронный.