бюджетное профессиональное образовательное учреждение Вологодской области

«Череповецкий лесомеханический техникум им. В.П. Чкалова»

Специальность **09.02.07** «Информационные системы и программирование»

**ОТЧЕТ ПО ПРОИЗВОДСТВЕННОЙ ПРАКТИКЕ**

**ПП по ПМ.03 РЕВЬЮИРОВАНИЕ ПРОГРАММНЫХ МОДУЛЕЙ**

Выполнил студент 3 курса группы ИС-31

Абрамова Анастасия Дмитриевна

подпись \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

место практики ООО «Малленом Системс»

Период прохождения:

с «8» декабря 2024 г.

по «21» декабря 2024 г.

Руководитель практики от

предприятия, должность:

Южакова Н.В., специалист по кадрам

подпись\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель практики от

техникума: Материкова А.А.

Оценка:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2024года

МП

Содержание

[1. Общая характеристика компании 3](#_Toc185327886)

[. 1.1 Организационная структура предприятия 3](#_Toc185327887)

[1.2 Внутренний распорядок работы компании и охрана труда 4](#_Toc185327888)

[1.3 Должностные инструкции ИТ-специалистов компании 5](#_Toc185327889)

[2. Ревьюирование программных продуктов 7](#_Toc185327890)

[2.1 Ревьюирование программного кода в соответствии с технической документацией 7](#_Toc185327891)

[2.2 Измерение характеристик компонентов программного продукта 8](#_Toc185327892)

[2.3 Исследование созданного программного кода с использованием специализированных программных средств 10](#_Toc185327893)

[2.4 Сравнительный анализ программных продуктов и средств разработки 12](#_Toc185327894)

[3. Выполняемые задания 13](#_Toc185327895)

[3.1 Задания первой недели 13](#_Toc185327896)

[3.2 Задания второй недели 14](#_Toc185327897)

[Заключение 22](#_Toc185327898)

[Список использованных источников 23](#_Toc185327899)

[Приложения 24](#_Toc185327900)

# Общая характеристика компании

Малленом Системс – ведущая российская компания в области разработки и внедрения систем компьютерного зрения, промышленной видеоаналитики на основе технологий машинного зрения и искусственного интеллекта (машинное обучение, нейронные сети глубокого обучения) и интеллектуальной обработки данных.

## . 1.1 Организационная структура предприятия

Линейная, есть отделы компании и в них есть руководители. Выше фотка, там названия отделов, ФИО руководителей писать нельзя, отделы можно. Краткая характеристика каждого отдела.

* Центр по развитию интеллектуальных систем, отдел разработки ПО. Проектирование, разработка, оптимизация ПО для клиентов компании.
* Производственно-технический отдел. Отдел с инженерами, которые проводят пусконаладочные работы на предприятиях, проектируют местонахождение оборудования на предприятии и устанавливают его.
* АУП (Административно-управленческий персонал). Руководство компании, которое формирует стратегии развития, управляет отделами, планирует деятельность предприятия, обеспечивает внешние коммуникации компании на выставках, в СМИ.
* Группа Маркетинга. Формирование маркетинговой стратегии компании, внутренний и внешний PR-компании, продвижение бренда и продуктов на рынке.
* Коммерческий отдел. Продажа продуктов компании заказчикам, поиск новых клиентов, участие в PR-продвижении компании.
* Отдел технической поддержки и контроля качества. Техническая поддержка пользователей и тестировка ПО на выявление ошибок и проблем.
* Отдел акселерационных и образовательных программ. Разработка и проведение обучающих курсов по машинному зрению и языку программирования, PR компании на рынке образовательных учреждений
* Отдел кадров. Управление персоналом компании, поиск, подбор, адаптация сотрудников, ведение кадрового документооборота, разработка стратегия развития персоналом предприятия
* Юридический отдел. Обработка всех документов в компании в соответствии с законодательством, взаимодействие с заказчиками и менеджерами по договорным обязательствам.
* Бухгалтерия. Ведение экономической деятельности предприятия, бухгалтерского учета, формирование бюджетов компании.
* ОХР (общественно-хозяйственные рабочие). Поддержание чистоты, порядка на рабочих местах, ремонт, уборка служебных помещений.

## 1.2 Внутренний распорядок работы компании и охрана труда

В компании "Малленом Системс" действует график работы 5/2, с 09:00 до 18:00. В штате компании есть отдельный специалист по охране труда, который проводит вводные инструктажи при приеме на работу и практике, а также занимается выдачей пропусков для пусконаладочных работ инженеров. В компании 20.09.2018 г. была проведена специальная оценка условий труда, согласно которой рабочие места, на территории которых установлены вредные производственные факторы, отсутствуют.

Так же производят обучение сотрудников по промышленной безопасности, по электробезопасности, охране труда и обучению в области применения средств защиты и оказания первой медицинской помощи в специализированном учебном центре (Негосударственное частное образовательное учреждение дополнительного профессионального образования «Учебный центр «Экоконсалт») с последующей аттестацией в Северо-Западном управлении Ростехнадзора в Вологодской области (только для промышленной безопасности и электробезопасности).

## 1.3 Должностные инструкции ИТ-специалистов компании

* Выполняет работу по проведению необходимых технических расчетов;
* Осуществляет наладку, настройку, регулировку и опытную проверку оборудования и систем, следит за его исправным состоянием;
* Принимает участие в проведение экспериментов и испытаний;
* Принимает участие в разработке программ, инструкций и другой технической документации, в изготовлении макетов, а также в испытаниях и экспериментальных работах;
* Выполняет работу по сбору, обработке и накоплению исходных материалов, данных статистической отчетности, научно-технической информации;
* Составляет описания проводимых работ, необходимые спецификации, диаграммы, таблицы, графики и другую техническую документацию;
* Выполняет работу по оформлению плановой и отчетной документации, вносит необходимые изменения и исправления в техническую документацию в соответствии с решениями, принятыми при рассмотрении и обсуждении выполняемой работы;
* Систематизирует, обрабатывает и подготавливает данные для составления отчетов о работе;
* Принимает необходимые меры по использованию в работе современных технических средств.

# Ревьюирование программных продуктов

Ревьюирование программных продуктов — это процесс оценки и анализа программного обеспечения с целью выявления его соответствия требованиям, выявления дефектов, улучшения качества кода и повышения общей надежности продукта. Важно, чтобы ревью проводилось систематически и регулярно, а также включало в себя различные аспекты разработки.

## 2.1 Ревьюирование программного кода в соответствии с технической документацией

Ревьюирование программного кода — это важный этап разработки, который помогает убедиться, что код соответствует не только функциональным требованиям, но и технической документации. Вот основные шаги, которые нужно учитывать при ревьюировании кода на соответствие технической документации:  
Подготовительный этап:

* Ознакомление с документацией: Перед началом ревью ознакомьтесь с технической документацией, спецификациями и требованиями.
* Изучение кода: Получите доступ к последней версии кода. Убедитесь, что ваш обзор кода основан на актуальной сборке.  
  Процесс ревью:
* Функциональная соответствие: Проверьте, реализованы ли все функциональные требования, указанные в документации. Сравните код с описанием функций, методов и их взаимодействия.
* Структура и архитектура: Убедитесь, что архитектура системы соответствует описанной в документации. Проверьте, правильно ли организована структура модулей и классов.
* Качество кода: Оцените читаемость, поддерживаемость и согласованность кода. Проверьте стиль кода на соответствие принятому в команде или проекте.
* Обработка исключений и ошибки: Убедитесь, что код корректно обрабатывает ошибки и исключения в соответствии с требованиями документации.
* Тестирование: Если в документации описаны требования к тестированию, убедитесь, что соответствующие тесты написаны и выполняются.
* Производительность: Проверьте, учитываются ли аспекты производительности, указанные в технической документации.  
  Заключительный этап:
* Формирование отчета: Составьте отчет о проведенном ревью, включая как выявленные несоответствия, так и положительные аспекты. Обеспечьте конструктивный подход к каждому замечанию.
* Обсуждение с командой: Проведите встречу с командой для обсуждения вашего отчета и получения комментариев от разработчиков, чтобы получить их мнение о выявленных проблемах
* Рекомендации по улучшению: Если есть возможность, предложите рекомендации по улучшению кода и документирования.

## 2.2 Измерение характеристик компонентов программного продукта

Функциональная корректность:

* Тестирование: Проведение модульного, интеграционного и системного тестирования.
* Покрытие тестами: Измерение процентного соотношения кода, охваченного тестами, например через инструменты для покрытия кода (JaCoCo, Istanbul).

Производительность:

* Время отклика: Измерение времени, необходимого для выполнения определенных действий или обработки запросов.
* Скорость обработки: Количество операций в секунду, которое приложение может выполнить.
* Нагрузочное тестирование: Оценка производительности системы под различными уровнями нагрузки.

Надежность:

* Время наработки на отказ (MTBF): Среднее время, в течение которого система работает без сбоев.
* Среднее время восстановления (MTTR): Время, необходимое для восстановления системы после сбоя.

Безопасность:

* Уязвимости: Оценка наличия известных уязвимостей (например, с помощью сканеров безопасности).
* Тестирование на проникновение: Проверка системы на наличие уязвимостей, которые могут быть использованы злоумышленниками.

Составность и сопровождение:

* Сложность кода: Использование метрик, таких как цикломатическая сложность, для оценки сложности отдельных функций и модулей.
* Документированность: Оценка наличия и качества документации на код и систему в целом.

Портируемость:

* Кроссплатформенность: Оценка способности системы работать на различных платформах и устройствах.
* Зависимости: Анализ внешних библиотек и модулей, от которых зависит проект.

Методики измерения:

* Автоматизированные инструменты: Использование статического и динамического анализаторов кода (SonarQube, ESLint и т. д.) для получения метрик кода.
* Системы мониторинга: Инструменты вроде New Relic или Grafana для сбора и анализа данных о производительности.
* Ручное тестирование: Тестировщики выполняют сценарии использования приложения и собирают данные.

## 2.3 Исследование созданного программного кода с использованием специализированных программных средств

Этапы исследования кода:

1. Статический анализ:

* Оценка структуры кода, соблюдения стилей программирования, наличия комментариев и документации.
* Проверка на наличие уязвимостей и ошибок.

1. Динамический анализ:

* Тестирование работы программы в режиме реального времени, отслеживание ее поведения.
* Профилирование производительности.

1. Тестирование:

* Написание и выполнение модульных тестов, интеграционных тестов и функциональных тестов.
* Использование систем автоматического тестирования.

1. Документирование:

* Создание документации по коду, его архитектуре, зависимостям и особенностям.
* Специализированные инструменты

Инструменты для статического анализа:

* SonarQube: анализатор кода, который помогает находить ошибки, уязвимости и улучшать качество кода.
* ESLint, Pylint: инструменты для проверки кода на соответствии стилю и наличию ошибок в языках JavaScript и Python соответственно.
* Coverity, Checkmarx: инструменты для выявления уязвимостей в коде.

Инструменты для динамического анализа:

* Valgrind: инструмент для обнаружения утечек памяти и проверки работы программы.
* JProfiler: профайлер для Java-приложений, позволяющий отслеживать производительность и использование памяти.

Инструменты для тестирования:

* JUnit, pytest: фреймворки для написания и выполнения тестов на Java и Python.
* Selenium: инструмент для автоматизации тестирования веб-приложений.

Git – система контроля версий, позволяет отслеживать изменения в коде, работать с ветвлением и объединением, а также упрощает совместную работу над проектами.

## 2.4 Сравнительный анализ программных продуктов и средств разработки

Основные критерии для сравнительного анализа: функциональные возможности, пользовательский интерфейс, производительность, совместимость, сообщество и поддержка, лицензирование и стоимость, интеграция.

Примеры программных продуктов и средств разработки:

* IDE (Интегрированные среды разработки):
* Visual Studio: Поддержка множества языков, мощные инструменты для отладки, но платная версия требует лицензии.
* Eclipse: Открытый исходный код, поддержка множества языков, но может быть сложен в настройке.

В процессе выбора программных продуктов и средств разработки важным является учитывать не только текущие потребности проекта, но и будущее развитие, масштабируемость и возможности для команды. Сравнительный анализ помогает принимать обоснованные решения и минимизировать риски.

# Выполняемые задания

## 3.1 Задания первой недели

Сначала я ознакомилась с организацией ООО «Малленом Системс». Узнала деятельность, направления, проекты и структурную организацию компании.

Далее я создала GIT репозиторий, в котором будут располагаться выполненные задания, отчёты и прочие документы текущей практики.

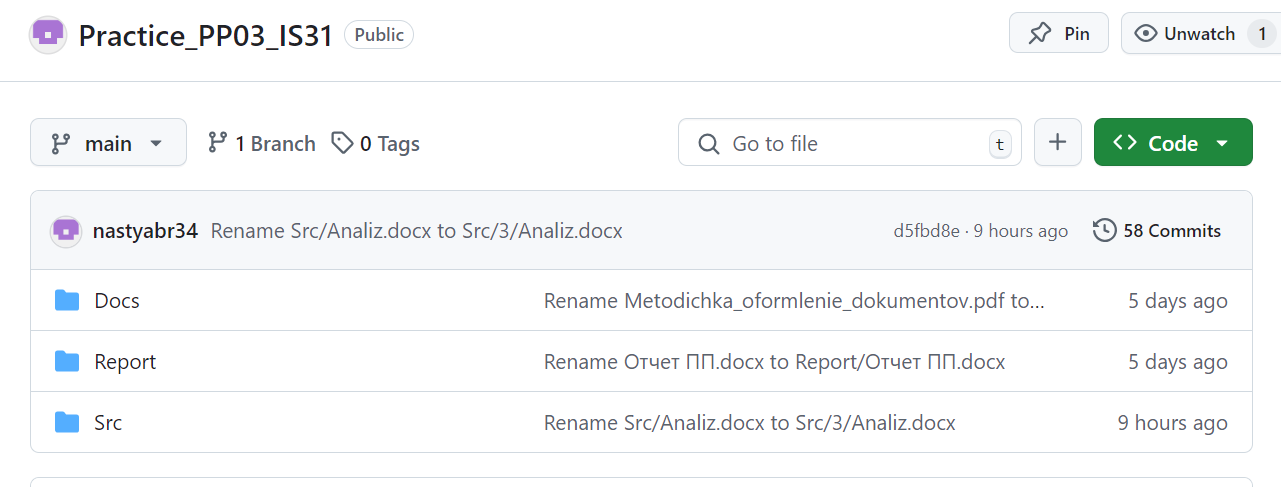


Рисунок 3.1 – Репозиторий «Practice\_PP03\_IS31»

В следующем задании я разработала два модуля модуль обработки и работы с изображениями и модуль взаимодействия с пользователем, формирование и хранений данных:

Изменение расширения файла (PNG to JPG, JPG to PNG).

Программный код представлен в Приложении А.

Код программы я разделила на следующие модули:

AppModule – это обычно основной модуль приложения. Код модуля представлен в Приложении Б.

MainModule – этот модуль может быть использован для организации основного функционала приложения.

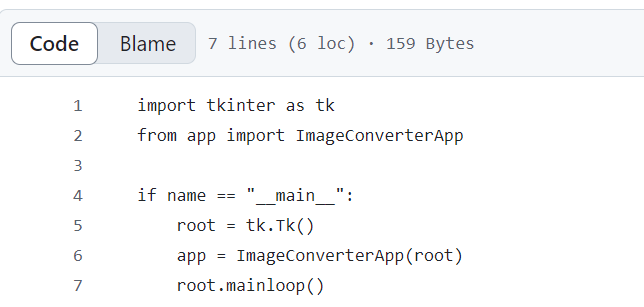


Рисунок 3.2 – «MainModule»

UtilsModule – этот модуль содержит утилитарные функции или службы, которые могут быть использованы в других частях приложения. Код модуля представлен в Приложении В.

## 3.2 Задания второй недели

В первом задании нужно провести обратное проектирование, используя графический язык UML. Я создала диаграммы Компонентов, Сценариев использования, Последовательностей, Деятельности.

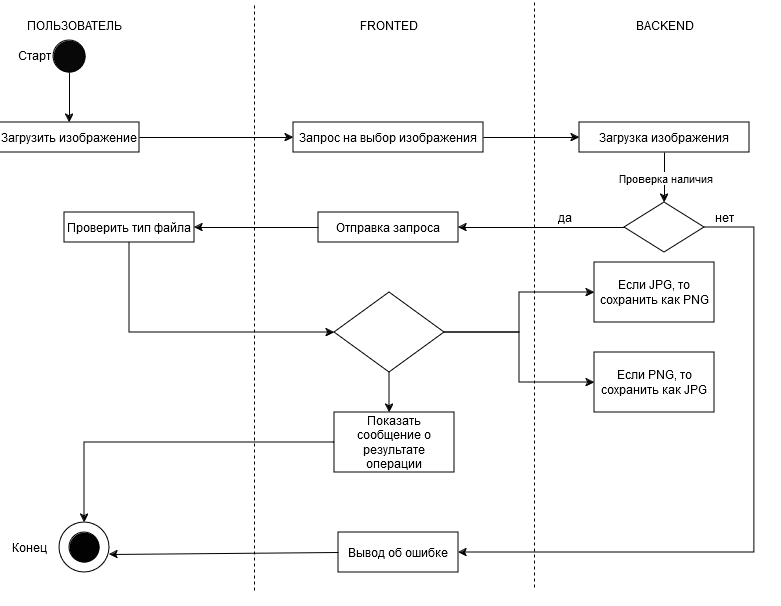


Рисунок 3.3 – Диаграмма Деятельности к программному коду «Изменение расширения файла»

* Пользователь вводит полное имя файла (включая его текущее расширение).
* Программа проверяет, что файл существует.
* Программа проверяет текущее расширение файла.
* Если расширение некорректное или файл не существует, процесс завершает выполнение и выводит сообщение об ошибке (переход к концу).
* Пользователь вводит новое расширение.
* Программа проверяет, что новое расширение корректно
* Программа формирует новое имя файла, заменяя старое расширение на новое.
* Если нового расширения не будет, процесс завершится, выводя сообщение об ошибке (переход к концу).
* Программа выполняет команду переименования файла на файловой системе.
* Если переименование прошло успешно, программа переходит к следующему шагу. В противном случае выводится сообщение об ошибке.
* Программа информирует пользователя о том, что расширение файла успешно изменено.

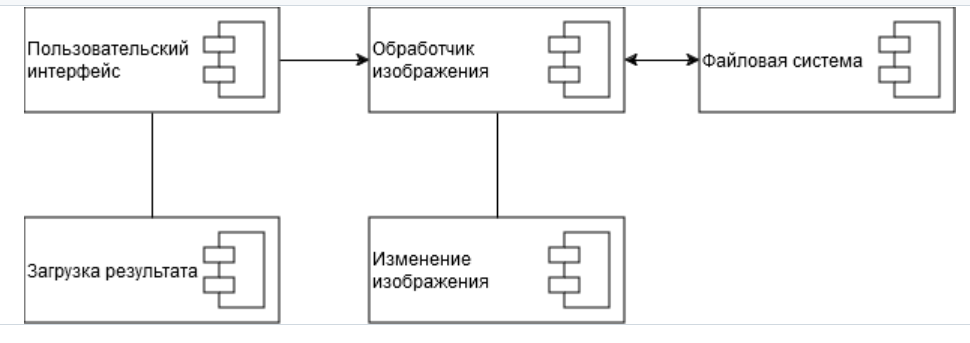


Рисунок 3.4 – Диаграмма Компонентов к программному коду «Изменение расширения файла»

* Пользователь загружает изображение через Пользовательский интерфейс.
* Интерфейс отправляет изображение в Обработчик изображений.
* Обработчик изображений выполняет проверку, изменяет формат файла и затем сохраняет его с помощью Файловой системы.
* После успешного завершения операции Обработчик изображений уведомляет Пользовательский интерфейс о завершении, и выводит ссылку на загруженное изображение или сообщение об ошибке.



Рисунок 3.5 – Диаграмма Последовательностей к программному коду «Изменение расширения файла»

* Пользователь выбирает изображение.
* Система загружает изображение.
* Пользователь нажимает «изменить расширение».
* Система проверяет допустимость нового расширения и применяет его к изображению.
* Система показывает сообщение о сохранении нового расширения файла.



Рисунок 3.6 – Диаграмма Сценариев использования к программному коду «Изменение расширения файла»

* Пользователь загружает изображение.
* Пользователь выбирает новое расширения файла.

Таб.3.1 – Скоростные показатели программы.

|  |  |
| --- | --- |
| Модуль обработки и работы с изображениями | Время выполнения загрузки изображения 0,036732 секунд. |
| Модуль AppModule | Время выбора файла 4,563218 секунд. |
| Модуль UtilsModule | Изображение загружено за 0.039974 секунд. |
| Модуль MainModul | Время выполнения load\_image: 0.027636 секунд. Время выполнения get\_creation\_date: 0.000651 секунд. Время выполнения get\_image\_info: 0.012335 секунд. |

Таб.3.2. – Размеры программ

|  |  |
| --- | --- |
| Модуль обработки и работы с изображениями | Размер (байты): 876340. |
| Модуль UtilsModule | Размер объекта 48 байт. |
| Модуль AppModule | Размер объекта 48 байт. |
| Модуль MainModul | Размер объекта 76640 байт. |

Я провела анализ средств разработки программ на языке программирования Python.

1. PyCharm

Плюсы:

* Мощные инструменты для отладки и анализа кода.
* Интеграция с системами контроля версий (Git, SVN и др.).
* Возможности для работы с веб-разработкой, Django, Flask и т.д.
* IntelliJ IDEA поддерживает множество языков и фреймворков.

Минусы:

* Требует значительных ресурсов системы.
* Бесплатная версия (Community Edition) имеет ограниченные возможности.

1. Visual Studio Code (VS Code)

Плюсы:

* Легкий и быстрый редактор с поддержкой множества языков.
* Расширяемость через плагины и интеграция с библиотеками для Python.
* Интуитивно понятный интерфейс и возможности кастомизации.

Минусы:

* Процесс настройки может занять время.
* Некоторые функции отладки могут быть менее удобными по сравнению с полноценными IDE.

1. IDLE

Плюсы:

* Входит в стандартную библиотеку Python, не требует установки дополнительных пакетов.
* Легкий в использовании, подходит для новичков.
* Имеет встроенную поддержку языка Python, что делает его доступным сразу после установки.

Минусы:

* Ограниченные возможности по сравнению с полноценными IDE.
* Интерфейс может показаться устаревшим для некоторых пользователей.
* Нет мощных инструментов для отладки и анализа кода.

Почему я выбрала IDLE? Потому что IDLE является отличным выбором для тех, кто ищет простоту и доступность. Она позволяет быстро начать программирование на Python. Программисты могут сразу же сосредоточиться на написании кода и изучении языка.

Поддержка языка и функции IDLE:

* Подсветка синтаксиса: IDLE поддерживает подсветку синтаксиса Python, что упрощает чтение и написание кода.
* Отладка: Встроенные инструменты для отладки кода, такие как установка точек останова и пошаговое выполнение.
* Интерактивный режим: Позволяет выполнять код по строкам и получать мгновенные результаты.
* Автозаполнение: Упрощает процесс написания кода, предлагая подсказки по именам переменных и функциям.

Анализ языка программирования Python

Язык программирования Python имеет широкий спектр поддержки и предоставляет множество функций, которые делают его популярным среди разработчиков.

Простота и читаемость: Python имеет простой и интуитивно понятный синтаксис, что облегчает обучение и написание кода.

Библиотеки и фреймворки: Python обладает огромным количеством сторонних библиотек и фреймворков, таких как:

* Flask и Django для веб-разработки,
* Pandas и NumPy для анализа данных,
* TensorFlow и PyTorch для машинного обучения,
* Matplotlib и Seaborn для визуализации данных.

Поддержка многопоточности и асинхронного программирования: Python предлагает возможности для работы с многопоточностью через стандартную библиотеку threading, а также поддерживает асинхронное программирование с помощью asyncio.

Кроссплатформенность: Python может работать на различных операционных системах, включая Windows, macOS и Linux.

Интерактивная среда разработки: Python поддерживает использование интерактивных оболочек, таких как IPython и Jupyter Notebook, что полезно для быстрого тестирования кода и работы с данными.

Обширное сообщество и документация: Python имеет большое сообщество разработчиков, что обеспечивает доступ к множеству ресурсов, сайтов и форумов для решения проблем и получения советов.

Поддержка объектно-ориентированного программирования: Python предоставляет возможности для создания классов, наследования и инкапсуляции, что делает код более организованным и повторно используемым.

Интеграция с другими языками: Python можно интегрировать с другими языками программирования, такими как C, C++ и Java, что позволяет использовать его в больших системах.

# Заключение

За эту практику я научилась следующему:

* Осуществлять ревьюирование программного кода в соответствии с технической документацией.
* Выполнять измерения характеристик компонентов программного продукта.
* Производить исследования созданного программного кода с использованием специализированных программных средств.
* Проводить сравнительный анализ программных продуктов и средств разработки.

# Список использованных источников

<https://metanit.com/python/tutorial/2.10.php>

<https://metanit.com/sharp/tutorial/3.46.php>

<https://app.diagrams.net/>

<https://www.mallenom.ru/?ysclid=m4l2ditzeb835497887>

<https://practicum.yandex.ru/blog/uml-diagrammy/>

<https://www.geeksforgeeks.org/how-to-check-the-execution-time-of-python-script/>

# Приложения

Приложение А



Рисунок А.1 – Первая часть программного кода «Изменение расширения файла»



Рисунок А.2 – Вторая часть программного кода «Изменение расширения файла»

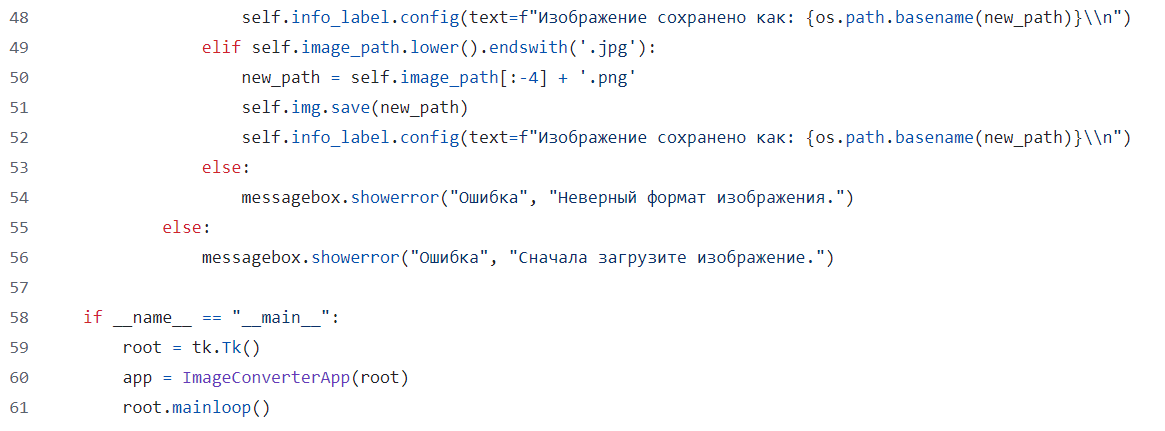


Рисунок А.3 – Третья часть программного кода «Изменение расширения файла»

Приложение Б

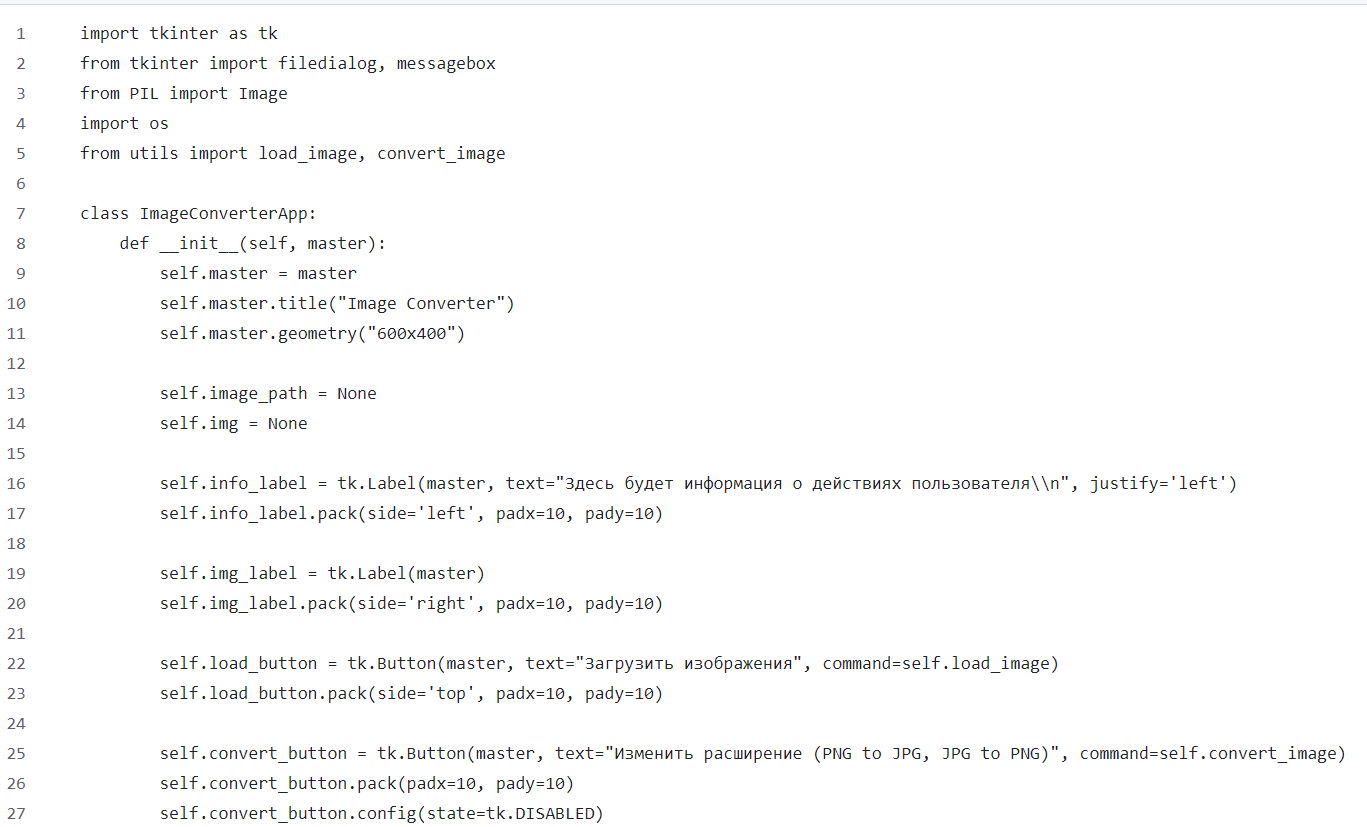


Рисунок Б.1 – Первая часть кода «AppModule»

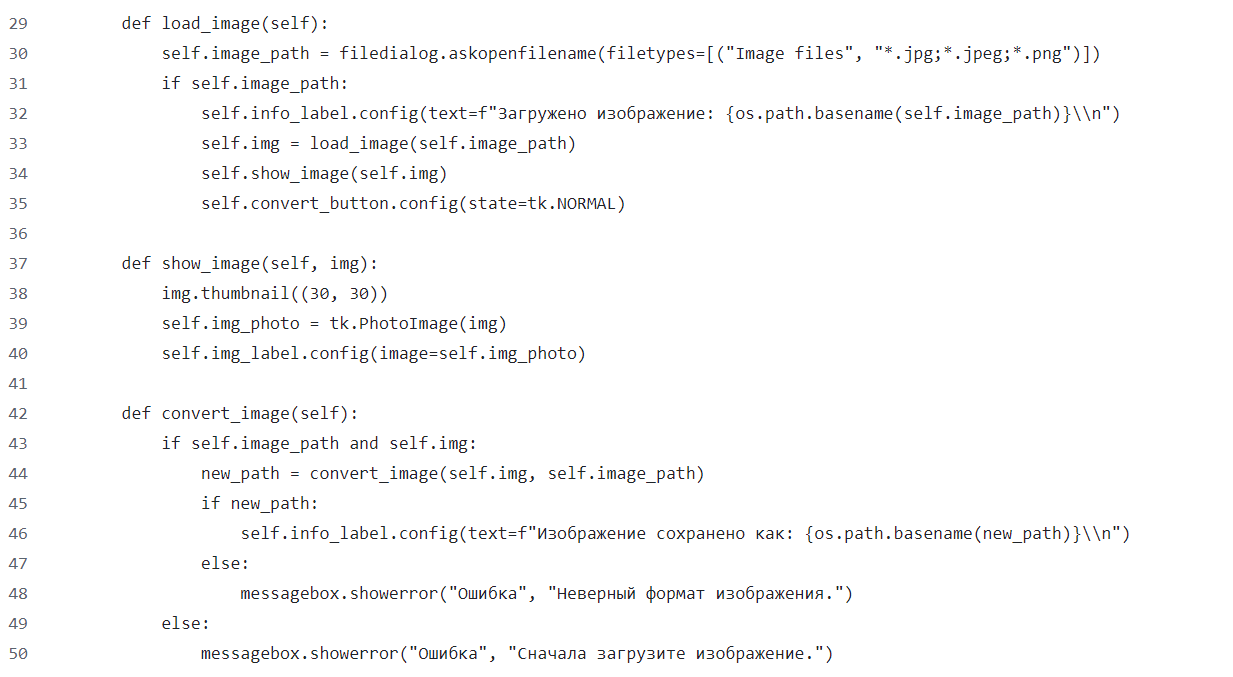


Рисунок Б.2 – Вторая часть кода «AppModule»

Приложение В



Рисунок В.1 – «UtilsModule»