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-\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

место практики \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

наименование юридического лица, ФИО ИП

Период прохождения:

с «\_\_\_» \_\_\_\_\_\_\_ 2024 г.

по «\_\_\_» \_\_\_\_\_\_\_ 2024 г.

Руководитель практики от

предприятия

должность\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

МПРуководитель практики от

техникума: Материкова А.А.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2024 года

г. Череповец

2024

**Содержание**

|  |  |
| --- | --- |
| Тема | **№ стр** |
| Введение:1. Общая характеристика предприятия | 3 |
| 1.1. Организационная структура предприятия | 4 |
| 1.2. Внутренний распорядок предприятия. | 4 |
| 1.3. Должностные инструкции работников предприятия | 5-7 |
| 2. Ревьюирование программных кодов | 8 |
| 2.1. Диаграммы | 8-10 |
| 2.2. Измерение характеристик программного кода | 10 |
| 2.3. Исследование созданного программного кода | 10-11 |
| 2.4. Используемые библиотеки и фреймворки | 11-18 |
| 3. Выполняемые задания | 19-21 |
| 3.1. Заключение | 21 |
| 3.2. Список литературы | 21 |
| 3.3. Приложения | 21 |

**Введение:**

**1. Общая характеристика предприятия.**

Прохождение производственной практики осуществлялось в компании ООО «Малленом Системс», которая является разработчиком систем технического зрения, машинного обучения и компьютерного моделирования для использования на транспорте и в промышленном производстве.

Глубокая специализация в области машинного зрения, большой опыт разработки и успешных внедрений, высокая квалификация сотрудников и широкая партнерская сеть делают «Малленом Системс» одной из самых надежных и привлекательных компаний на российском IT-рынке.

Продукция компании сегодня представлена в большинстве регионов РФ, странах СНГ и ЕС. С 2012 года «Малленом Системс» — единственный в России и СНГ официальный партнер-интегратор ведущего производителя систем технического зрения — американской компании Cognex.

Территориально офис предприятия находится в г. Череповце ул. Металлургов дом 21б, где размещена часть штата компании, начиная от обслуживающего персонала, юристов и программистов и заканчивая бухгалтерией и руководством, остальные работники компании работают удаленно. Численность сотрудников, работающих в данном офисе, составляет 25-30 человек. Всего в штате компании находится 80 человек. Не смотря на необычную обстановку в стране, практика на данном предприятии проходилась очно.

Научно-производственная компания «Малленом Системс» создана в 2011 году на базе уже сложившейся команды ученых и программистов Санкт-Петербургского политехнического университета Петра Великого при поддержке инвестиционной компании «Малленом». К настоящему времени сформирован профессиональный и сбалансированный коллектив, включая современный исследовательский сектор (1 доктор и 4 кандидата наук).

**1.1. Организационная структура предприятия**

Организационная структура компании включает исследовательские и проектные отделы, которые работают над задачами в сфере машинного зрения и аналитики. Внутренняя структура также обеспечивает гибкость в управлении проектами для удовлетворения специфических потребностей клиентов в различных отраслях.

**1.2. Внутренний распорядок предприятия.**

У предприятия есть 2 офиса

* **Ул. Металлургов 21Б**

Офис работает:

Пн-Пт. с 8:00 до 20:00

Сб-Вс. с 10:00 до 17:00

* **Ул. Ленина 110Б**

Офис работает:

Пн-Пт. с 8:00 до 18:00

Рабочий день

Полная ставка

Продолжительность рабочего времени составляет 40 часов в неделю

Два выходных дня –суббота и воскресенье

Неполная ставка

Продолжительность рабочего времени определяется долей ставки:

* 0,25 –10 часов в неделю
* 0,3 –12 часов в неделю
* 0,5 –20часов в неделю

Выходные такие же –суббота и воскресенье

В компании есть общепринятый режим работы для большинства

сотрудников, работающих на полную ставку:

* рабочее время с 9.00 до 18.00
* обед с 13.00 до 14.00

Два технологических перерыва по 20 минут в течение дня

Режим работы может быть установлен для работника индивидуально,

по согласованию с руководителем, но при условии отработки нормы

рабочего времени за неделю.

**1.3. Должностные инструкции работников предприятия**

Основные должности в компании:

**Инженер-программист**

* разработка приложений под ОС Windows;
* интеграция с алгоритмами машинного обучения;
* программирование UI;
* реализация алгоритмов машинного зрения;
* доработка существующих проектов;
* оптимизация и рефакторинг.

**Специалист по машинному обучению**

* дообучение / улучшение существующих нейросетей, используемых в production;
* создание и обучение нейросетей;
* анализ современных моделей на применимость их бизнес-задачам компании;
* визуализация данных;
* работа с датасетами.

**Инженер**

* проработка и согласование технических заданий по проектам;
* подбор оборудования и комплектующих, разработка спецификаций;
* подготовка оборудования к инсталляции;
* выполнение проектно-изыскательских работ;
* выполнение пусконаладочных работ на объектах внедрения (служебные командировки);
* обучение операционного персонала Заказчика;
* техническая поддержка клиентов;
* разработка технической документации

**Специалист по тестированию ПО**

* ручное тестирование;
* составление тестовых сценариев;
* поддержка и расширение документации по продуктам проекта;
* документирование и верификация дефектов, контроль исправления выявленных ошибок разработчиком;
* взаимодействие с командой разработки и технической поддержки;
* тестирование продуктов проекта;
* актуализация документации по продуктам проекта.

**Менеджер по продажам**

* Обработка входящих запросов от клиентов.
* Ведение коммерческих переговоров с клиентами, консультирование о продуктах Малленом Системс для транспортной отрасли.
* Подготовка ТКП (совместно с техническими специалистами), согласование конфигурации продукции под каждую задачу, подбор оборудования под проект.
* Заключение договоров (совместно с юристом) и их сопровождение.
* Контроль работы по отгрузке и доставке товаров покупателям по заключенным договорам, подготовка товара к отправке.
* Контроль оплаты договоров клиентами.
* Участие в торгах на поставку продукции Компании на торговых площадках
* Ведение информационных баз клиентов и партнеров, документооборота.

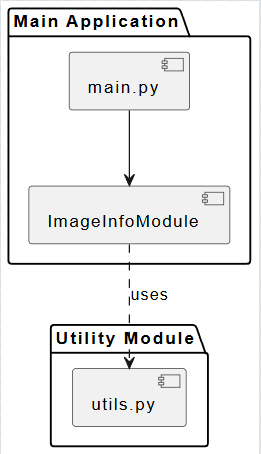
**2. Ревьюирование программных кодов**

**2.1. Диаграммы**

**Диаграмма компонентов**:

Показывает взаимодействие между модулями: главный скрипт main.py, модуль с интерфейсом пользователя ImageInfoModule и вспомогательный модуль utils.py.

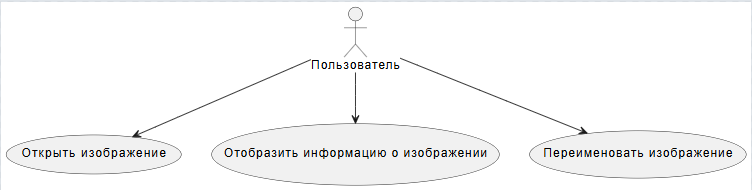
Взаимосвязь: ImageInfoModule использует utils.py для получения информации о файле.



**Диаграмма сценариев использования**:

Показывает три основные функции: открыть изображение, отобразить информацию о нем и переименовать изображение.

Пользователь является основным актором, который выполняет эти действия.

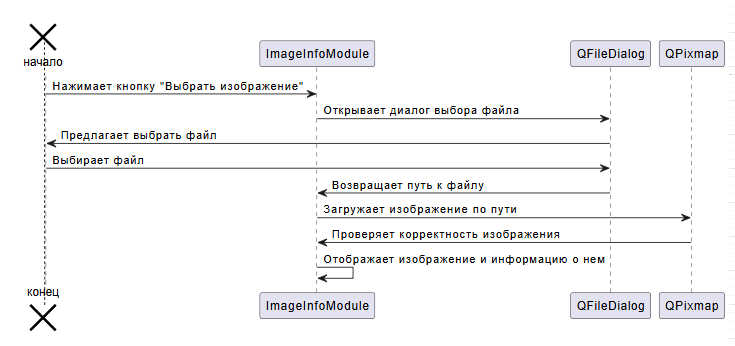


**Диаграмма последовательности**:

Показывает последовательность действий при выборе изображения:

Пользователь инициирует выбор изображения, диалог выбора файла открывается через QFileDialog.

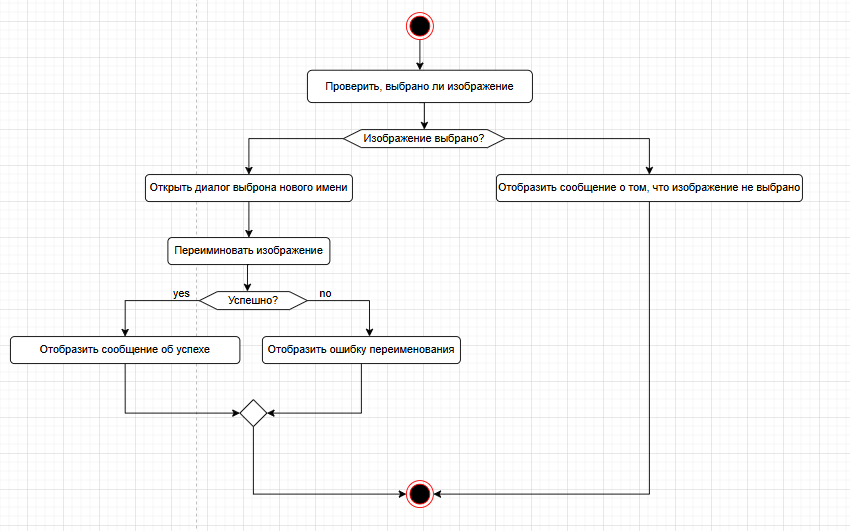
После выбора изображения QPixmap загружает и проверяет изображение.



**Диаграмма деятельности**:

Показывает процесс переименования изображения.

Сначала проверяется, выбрано ли изображение, затем выполняются шаги для переименования, с проверкой на успешность.



**2.2. Измерения характеристик программного кода**

**Скоростные показатели программы:**

get\_file\_creation\_time: 0.000128 секунд

format\_file\_size: 0.000012 секунд

open\_image: 2.040713 секунд

rename\_image: 2.758197 секунд

**Размеры программы:**

main.py — 243 байт

image\_info\_module.py — 3,03 КБ

utils.py — 328 байт

**2.3. Исследование созданного программного кода**

**1. Статический анализ кода**  
Цель: Выявить проблемы в коде, такие как нарушение PEP8, неиспользуемые импорты, циклические зависимости, отсутствие типизации.  
Инструменты:  
pylint: Проверяет соответствие PEP8 и выявляет потенциальные ошибки.  
flake8: Анализирует синтаксис.  
mypy: Проверяет правильность аннотаций типов.  
**2. Динамический анализ**  
Цель: Проверить производительность, использование памяти и выполнение кода.  
Инструменты:  
memory\_profiler: Отслеживает использование памяти.  
timeit: Измеряет время выполнения функций.  
**3. Генерация графов вызовов**  
Цель: Понять, как функции взаимодействуют друг с другом.  
Инструмент:  
pycallgraph: Визуализирует вызовы функций.  
**4. Тестирование функций**  
Цель: Проверить корректность работы функций.  
Инструмент:  
pytest: Для написания и выполнения тестов.

### **2.4. Используемые библиотеки и фреймворки**

PyQt5

**PyQt5** — это один из самых популярных фреймворков для разработки графических интерфейсов пользователя (GUI) на Python. Он является оберткой для библиотеки Qt, которая предоставляет мощные инструменты для создания кросс-платформенных приложений.

**Преимущества**:

* **Кросс-платформенность**: PyQt5 работает на Windows, macOS и Linux.
* **Обширный набор виджетов**: включает кнопки, диалоговые окна, текстовые поля и другие элементы управления.
* **Гибкость**: благодаря объектно-ориентированному подходу, позволяет разработчикам создавать сложные интерфейсы с небольшими усилиями.
* **Поддержка современных стандартов**: поддержка событийно-ориентированного программирования и асинхронных вызовов.

**Недостатки**:

* **Размер пакета**: для простых приложений PyQt5 может оказаться чрезмерно тяжелым, особенно если приложение не использует все возможности фреймворка.
* **Сложность**: при глубоком погружении в Qt может возникнуть необходимость работы с низкоуровневыми аспектами, что может потребовать дополнительных знаний о Qt.

**Os**

Модуль os предоставляет функционал для взаимодействия с операционной системой, такой как работа с файлами и каталогами, получение информации о путях и т.д.

**Преимущества**:

* **Мощный API**: позволяет легко манипулировать файловыми путями, создавать, читать, писать файлы, а также изменять параметры ОС.
* **Портируемость**: работает на всех операционных системах, поддерживающих Python, что делает код независимым от платформы.

**Недостатки**:

* **Отсутствие многозадачности**: требует дополнительных библиотек для работы с параллельными процессами или многозадачностью.
* **Ограниченность в асинхронных операциях**: операции с файловой системой могут быть блокирующими, что плохо влияет на производительность, если они выполняются в главном потоке.

**Time**

Модуль time предоставляет базовые функции для работы с временем, включая замеры времени выполнения.

**Преимущества**:

* **Простота использования**: позволяет просто и быстро измерить время выполнения кода.
* **Синхронность**: идеально подходит для измерения времени работы синхронных операций.

**Недостатки**:

* **Низкая точность**: для более точных и подробных измерений времени, особенно в многозадачных приложениях, можно использовать более специализированные инструменты, такие как timeit или профилировщики.

### **2. Среда разработки**

Для разработки Python-программ можно использовать различные IDE (интегрированные среды разработки) и текстовые редакторы. Для вашего проекта подходят следующие инструменты:

**PyCharm**

PyCharm — мощная IDE для Python от JetBrains. Она поддерживает работу с PyQt, а также содержит встроенные средства для отладки, тестирования, работы с версиями и управления зависимостями.

**Преимущества**:

* **Автодополнение и IntelliSense**: улучшает продуктивность, помогает избегать ошибок.
* **Поддержка фреймворков**: поддерживает популярные фреймворки, включая PyQt.
* **Встроенная отладка**: интегрированные средства отладки и профилирования кода.

**Недостатки**:

* **Медленное время запуска**: PyCharm может потреблять много ресурсов, что делает его менее подходящим для небольших проектов.
* **Проблемы с производительностью на слабых машинах**.

**Visual Studio Code**

VSCode — легкий, расширяемый редактор, который поддерживает Python через расширения. Он часто используется для разработки как небольших, так и крупных приложений.

**Преимущества**:

* **Легкость и быстрота**: требует меньше системных ресурсов, чем PyCharm.
* **Широкая настройка**: поддержка плагинов для разных фреймворков и инструментов, включая PyQt5.
* **Интеграция с Git**: встроенная поддержка системы контроля версий.

**Недостатки**:

* **Меньше встроенных функций**: в отличие от PyCharm, некоторые функции нужно устанавливать вручную через расширения.
* **Отсутствие некоторых инструментов** для работы с большими проектами, таких как профилировщики и инспекторы кода.

**Jupyter Notebook**

Jupyter Notebook часто используется для прототипирования и анализа данных, но также может быть полезен для тестирования отдельных частей кода.

**Преимущества**:

* **Интерактивная среда**: позволяет запускать код по частям и наблюдать результаты немедленно.
* **Удобно для анализа и тестирования**: полезно для проверки работы отдельных функций или вычислений.

**Недостатки**:

* **Неудобен для графических интерфейсов**: не лучший инструмент для работы с GUI-приложениями.

### **3. Подходы к тестированию**

Для тестирования вашего кода можно использовать несколько подходов, начиная от простых юнит-тестов до интеграционных тестов и тестов производительности.

**unittest**

unittest — это стандартная библиотека Python для создания юнит-тестов.

**Преимущества**:

* **Стандартный инструмент**: встроенная поддержка в Python.
* **Гибкость**: позволяет легко писать тесты для каждой функции.

**Недостатки**:

* **Проблемы с асинхронными тестами**: для асинхронных операций может потребоваться больше настройки.

**pytest**

pytest — это расширенная библиотека для тестирования, которая поддерживает более сложные сценарии, а также предоставляет удобный интерфейс для написания и запуска тестов.

**Преимущества**:

* **Простота и лаконичность**: предоставляет удобный синтаксис для тестирования.
* **Поддержка асинхронных операций**: легко интегрируется с асинхронными тестами.

**Недостатки**:

* **Отсутствие интеграции с UI**: для тестирования UI-интерфейсов, как в случае с PyQt, потребуется использовать дополнительные инструменты.

**Selenium (для GUI-тестирования)**

Selenium — это инструмент для автоматизации браузеров, но он также может быть использован для тестирования GUI-приложений, включая PyQt.

**Преимущества**:

* **Мощные возможности автоматизации**: может тестировать взаимодействие с пользователем в графическом интерфейсе.
* **Поддержка разных браузеров и приложений**.

**Недостатки**:

* **Сложность настройки**: требует дополнительной настройки и знаний для тестирования GUI-приложений.

### **4. Производительность**

Для тестирования производительности вашего кода можно использовать следующие подходы:

**timeit**

Модуль timeit — это стандартный инструмент для измерения времени выполнения небольших частей кода. Он особенно полезен для сравнительных замеров производительности различных алгоритмов.

**Преимущества**:

* **Высокая точность**: использует внутренние механизмы Python для точных измерений.
* **Легкость использования**.

**Недостатки**:

* **Не работает с многозадачными или многопоточными операциями**: может потребоваться дополнительная настройка для асинхронных задач.

**cProfile**

cProfile — это профилировщик для Python, который предоставляет детальную информацию о том, сколько времени каждый метод или функция занимает в ходе выполнения программы.

**Преимущества**:

* **Детальные отчеты**: может анализировать всю программу и выявлять "узкие места".
* **Поддержка многозадачности**.

**Недостатки**:

* **Перегрузка**: использование профилировщика может замедлить выполнение программы, так что его следует использовать только в процессе оптимизации.

**3. Выполняемые задания**

**Код:**

import sys

from PyQt5.QtWidgets import QApplication, QWidget, QPushButton, QLabel, QVBoxLayout, QFileDialog

from PyQt5.QtGui import QPixmap

import os

class ImageInfoModule(QWidget):

def \_\_init\_\_(self):

super().\_\_init\_\_()

self.setWindowTitle("Image Info Module")

self.image\_path = ""

layout = QVBoxLayout()

self.image\_label = QLabel("Выберите изображение")

layout.addWidget(self.image\_label)

self.image\_button = QPushButton("Выбрать изображение")

self.image\_button.clicked.connect(self.open\_image)

layout.addWidget(self.image\_button)

self.info\_label = QLabel("")

layout.addWidget(self.info\_label)

self.rename\_button = QPushButton("Переименовать изображение")

self.rename\_button.clicked.connect(self.rename\_image)

layout.addWidget(self.rename\_button)

self.setLayout(layout)

def open\_image(self):

options = QFileDialog.Options()

options |= QFileDialog.DontUseNativeDialog

self.image\_path, \_ = QFileDialog.getOpenFileName(self, "Выберите изображение", "", "Изображения (\*.jpg \*.png \*.bmp)", options=options)

if self.image\_path: # Проверяем, что путь не пуст

pixmap = QPixmap(self.image\_path)

if pixmap.isNull(): # Проверяем, удалось ли загрузить изображение

self.info\_label.setText("Ошибка загрузки изображения.")

return

self.image\_label.setPixmap(pixmap.scaled(300, 300, aspectRatioMode=1)) # Масштабируем для корректного отображения

image\_info = f"Размер: {os.path.getsize(self.image\_path)} байт\n"

image\_info += f"Разрешение: {pixmap.width()}x{pixmap.height()}\n"

image\_info += f"Дата создания: {os.path.getctime(self.image\_path)}"

self.info\_label.setText(image\_info)

else:

self.info\_label.setText("Файл не выбран.")

def rename\_image(self):

if self.image\_path:

new\_name, \_ = QFileDialog.getSaveFileName(self, "Выберите новое имя для изображения", "", "Изображения (\*.jpg \*.png \*.bmp)")

if new\_name:

try:

os.rename(self.image\_path, new\_name)

self.image\_path = new\_name

self.info\_label.setText("Изображение успешно переименовано.")

except Exception as e:

self.info\_label.setText(f"Ошибка переименования: {e}")

else:

self.info\_label.setText("Выберите изображение перед переименованием.")

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

app = QApplication(sys.argv)

window = ImageInfoModule()

window.show()

sys.exit(app.exec\_())

**Заключение**

В рамках производственной практики мной был выполнен комплексный анализ программных продуктов и их компонентов. В задачи входили ревьюирование кода, измерение производительности, использование инструментов анализа и проведение сравнительного обзора технологий.  
  
Работа над проектом дала мне возможность глубже разобраться в процессах проектирования, тестирования и оптимизации программного обеспечения. Практика в ООО "Малленом Системс" способствовала совершенствованию моих навыков разработки и анализа, а также обеспечила ценный опыт работы с современными библиотеками и инструментами Python.

**Список литературы:**

1. UML - https://practicum.yandex.ru/blog/uml-diagrammy/?

2. Пример измерения скорости используя Time - https://www.geeksforgeeks.org/how-to-check-the-execution-time-of-python script/

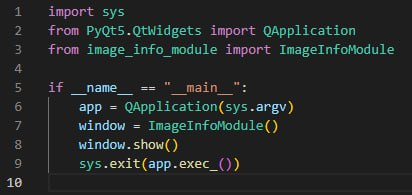
3. https://app.diagrams.net/

4. Работа с модулями Python - https://metanit.com/python/tutorial/2.10.php

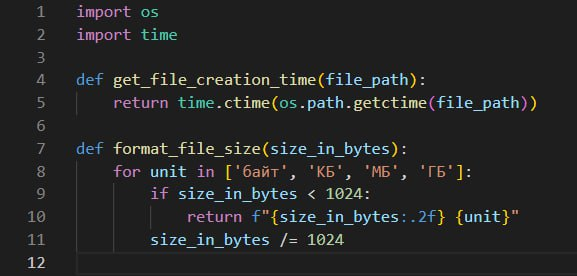
5. Работа с библиотеками <https://metanit.com/sharp/tutorial/3.46.php>

**Приложения:**

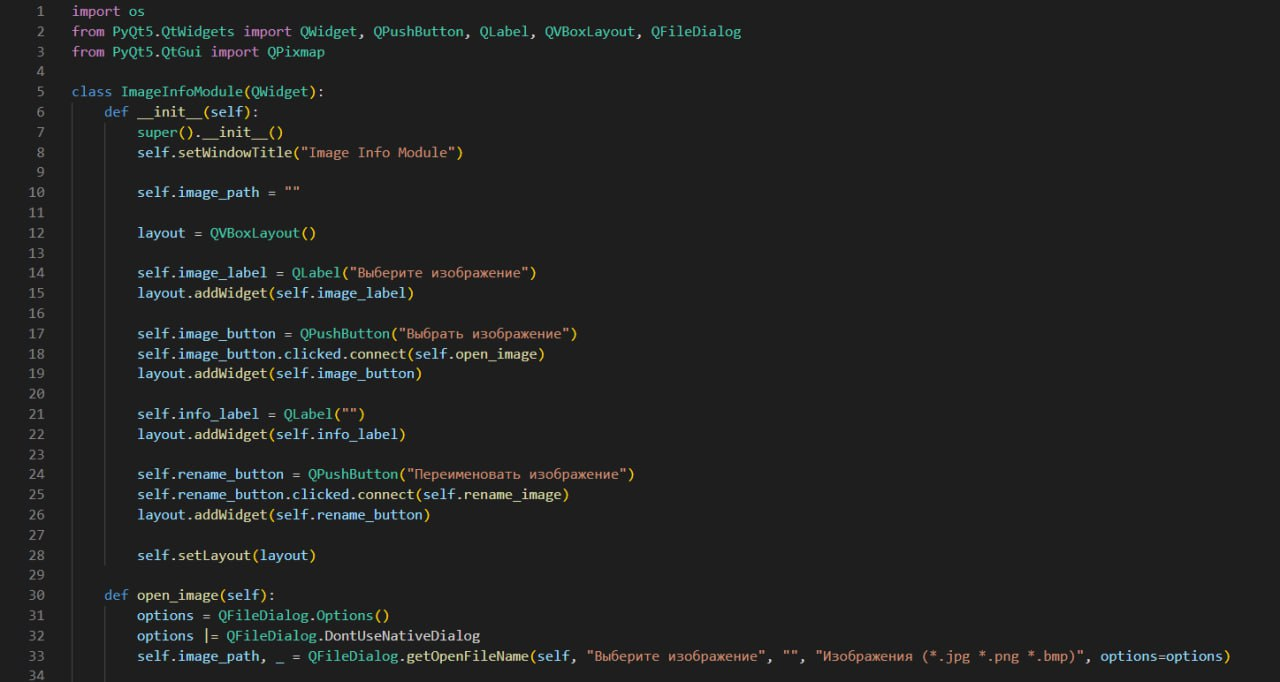
**Main.py**

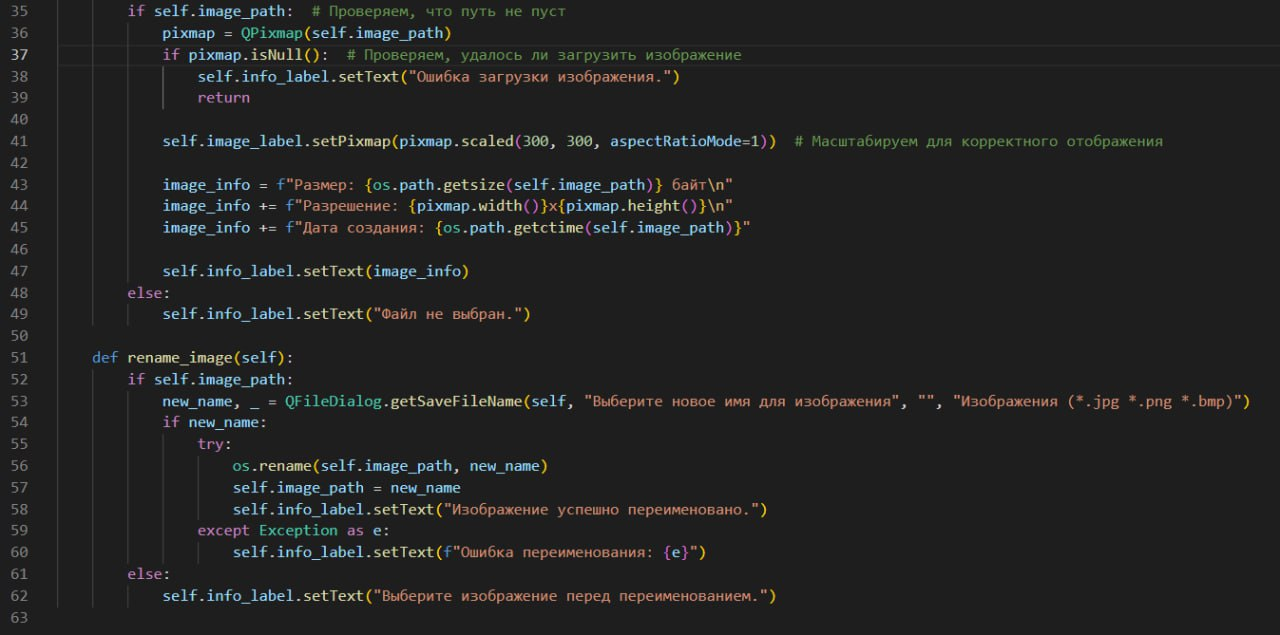


**Utils.py**



**image\_info\_module.py**





**Рабочая программа**

