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.02 Осуществление интеграции программных модулей**

Выполнил студент 2 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\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

МП

Руководитель практики от

техникума: Материкова А.А.

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

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

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

г. Череповец 2024

Содержание

Введение 3

1.Общая характеристика предприятия (организации) 5

1.1.Организационная структура предприятия 6

1.2. Внутренний распорядок работы предприятия, охрана труда на предприятии 7

1.3.Должностные инструкции ИТ-специалистов предприятия 7

2.Осуществление интеграции программных модулей 8

2.1.Разработка требования к программным модулям на основе анализа проектной и технической документации на предмет взаимодействия компонент 8

2.2.Выполнение интеграции модулей в программное обеспечение 10

2.3.Выполнение отладки программного модуля с использованием специализированных программных средств 12

2.4.Осуществление разработки тестовых наборов и тестовых сценариев для программного обеспечения 14

2.5.Инспектирование компонент программного обеспечения на предмет соответствия стандартам кодирования 16

3.Выполняемые задания 17

Заключение 28

Список использованных источников 28

Приложения 30

Введение:

Место прохождения производственной практики – ООО «Малленом Системс»

Сроки прохождения производственного практики – с 24.11.24 по 7.12.24

Задачами данной практики являются подготовка обучающихся осознанному и углубленному изучению дисциплин, привитие им практических умений и получение первичных профессиональных навыков по выбранной специальности.

Целями производственной практики (по профилю специальности) являются:

закрепление И совершенствование общих И профессиональных

компетенций:

ОК 01. Выбирать способы решения задач профессиональной деятельности применительно к различным контекстам;

ОК 02. Использовать современные средства поиска, анализа и интерпретации информации и информационные технологии для выполнения задач профессиональной деятельности;

ОК 03. Планировать и реализовывать собственное профессиональное и личностное развитие, предпринимательскую деятельность B профессиональной сфере, использовать знания по финансовой грамотности в различных жизненных ситуациях;

ОК 04. Эффективно взаимодействовать и работать в коллективе и команде; ОК 05. Осуществлять устную и письменную коммуникацию на государственном языке Российской Федерации с учетом особенностей социального и культурного контекста;

ОК 06. Проявлять гражданско-патриотическую позицию, демонстрировать осознанное поведение на основе традиционных общечеловеческих ценностей, в том числе с учетом гармонизации межнациональных и межрелигиозных отношений, применять -стандарты антикоррупционного поведения;

ОК 07. Содействовать сохранению окружающей среды, ресурсосбережению, применять знания об изменении климата, принципы бережливого производства, эффективно действовать в чрезвычайных ситуациях;

ОК 08. Использовать средства физической культуры для сохранения и укрепления здоровья в процессе профессиональной деятельности и поддержания необходимого уровня физической подготовленности;

ОК 09. Пользоваться профессиональной документацией на государственном и иностранном языках.

ПК 2.1. Разрабатывать требования к программным модулям на основе анализа проектной и технической документации на предмет взаимодействия компонент

ПК 2.2. Выполнять интеграцию модулей в программное обеспечение

ПК 2.3. Выполнять отладку программного модуля с использованием специализированных программных средств использованием

ПК 2.4. Осуществлять разработку тестовых наборов и тестовых сценариев для программного обеспечения

ПК 2.5. Производить инспектирование компонент программного обеспечения на предмет соответствия стандартам кодирования

1.Общая характеристика предприятия (организации)

«Малленом Системс» — ведущий российский разработчик и интегратор систем компьютерного зрения и промышленной видеоаналитики, основанный в 2011 году при поддержке инвестиционной компании «Малленом».

Компания, состоящая из специалистов Санкт-Петербургского политехнического университета Петра Великого, использует передовые технологии машинного обучения и глубокого обучения для создания решений в различных отраслях, включая транспорт, машиностроение, нефтегазовый сектор, металлургию, пищевую промышленность, фармацевтику, алмазодобычу и атомную энергетику.

Продукция компании представлена в России, странах СНГ и ЕС. Более 10 лет компания «Малленом Системс» является официальным партнером Cognex, мирового лидера в области машинного зрения, а с 2022 года — представителем Hikrobot (Hikvision) в РФ и ЕАЭС, предлагая оборудование для машинного зрения и робототехники. Компания входит в национальный рейтинг быстрорастущих технологических компаний «ТехУспех».

1.1.Организационная структура предприятия

1.Высшее руководство

Генеральный директор : Ответственен за стратегическое управление и общее руководство компанией.

Директор по развитию: занимается стратегическим планированием и реализацией инициатив, направленных на рост бизнеса

Главный технический директор : отвечает за технологическое развитие и инновации.

2.Отдел разработки программного обеспечения:

Руководитель отдела разработки

Разработка алгоритмов компьютерного зрения, программного обеспечения для анализа видео, интеграция с оборудованием. Может быть разделён на более мелкие команды, специализирующиеся на отдельных аспектах (например, обработка изображений, машинное обучение, разработка интерфейсов).

3.Отдел тестирования и контроля качества:

проверка качества программного и/или аппаратного обеспечения.

- Инженерные службы:

Руководитель направления системной интеграции: отвечает за управление проектами, связанными с интеграцией различных IT-систем и технологий в единую инфраструктуру.

- Отдел системной интеграции: Интегрирует решения в инфраструктуру клиентов.

- Отдел технической поддержки: Обеспечивает поддержку клиентов, решает технические проблемы.

1.2. Внутренний распорядок работы предприятия, охрана труда на предприятии (организации)

В компании «Малленом Системс» действует стандартный пятидневный рабочий график (5/2) с 9:00 до 18:00. Для обеспечения безопасности и соблюдения норм охраны труда в штате имеется специалист по охране труда, который проводит вводные инструктажи при приёме на работу и стажировке, а также выдаёт пропуска для инженеров на пусконаладочные работы. Специальная оценка условий труда от 20.09.2018 подтвердила отсутствие вредных производственных факторов на рабочих местах компании.

1.3. Должностные инструкции ИТ-специалистов предприятия

Техник выполняет следующие должностные обязанности:

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.

2. Осуществление интеграции программных модулей

2.1. Разработка требования к программным модулям на основе анализа проектной и технической документации на предмет взаимодействия компонент

1. Анализ проектной документации

• Изучение архитектуры системы: Определите основные компоненты системы, их функции и взаимосвязи.

• Определение бизнес-требований: Понять, какие задачи должен решать проект и какие требования предъявляются к функциональности.

2. Анализ технической документации

• Технические спецификации: Ознакомьтесь с техническими требованиями, такими как производительность, безопасность и совместимость.

• Стандарты и протоколы: Убедитесь, что все модули будут соответствовать установленным стандартам и протоколам взаимодействия.

3. Идентификация взаимодействий между компонентами

• Определение интерфейсов: Опишите, как различные модули будут взаимодействовать друг с другом. Это может включать API, сообщения или другие методы передачи данных.

• Протоколы взаимодействия: Установите, какие протоколы будут использоваться для обмена данными между компонентами.

4. Формирование требований к модулям

• Функциональные требования: Опишите, что каждый модуль должен делать. Например, обработка данных, взаимодействие с пользователем и т.д.

• Нефункциональные требования: Укажите требования к производительности, безопасности, надежности и удобству использования.

5. Документирование требований

• Создание спецификаций: Все требования должны быть четко задокументированы в формате, удобном для команды разработки.

• Использование моделей: Рассмотрите возможность использования UML-диаграмм или других визуальных средств для представления взаимодействий между компонентами.

6. Верификация и валидация требований

• Проверка на полноту и непротиворечивость: Убедитесь, что все требования полны и не противоречат друг другу.

• Обсуждение с заинтересованными сторонами: Привлеките всех заинтересованных лиц для обсуждения и подтверждения требований.

7. Управление изменениями

• Процесс управления изменениями: Установите процедуры для обработки изменений в требованиях на протяжении всего жизненного цикла проекта.

# 2.2. Выполнение интеграции модулей в программное обеспечение

1. Подготовительный этап

1.1 Определение архитектуры

- Выбор архитектурного стиля: важно на этапе проектирования выбрать архитектурный подход, такой как микросервисы, модули в рамках монолита, или распределённая система.

- Документация: построение документации, описывающей компоненты системы и их взаимодействия, включая диаграммы, схемы и API.

1.2 Установление стандартов

- Кодирование: применение единых стандартов кодирования и стиля для всех модулей.

- API: проектирование и документирование интерфейсов, которым должны следовать все модули.

2. Процесс интеграции

2.1 Интеграция поэтапно

- Это подразумевает постепенное добавление модулей в систему, начиная с наиболее критичных.

- Тестирование на каждом этапе: после добавления каждый модуль нужно тестировать, чтобы убедиться, что все работает как ожидается.

3. Тестирование интеграции

3.1 Интеграционные тесты

- Разработка тестов, которые проверяют взаимодействие между модулями, чтобы убедиться, что система реагирует на вызовы должным образом.

- Используйте автоматизацию для повышения эффективности и снижения вероятности человеческой ошибки.

3.2 Нагрузочное тестирование

- Проверка производительности системы при высокой нагрузке для выявления узких мест, которые могут возникнуть при взаимодействии модулей.

4. Непрерывная интеграция и непрерывное развертывание

4.1 Автоматизация процессов

- Внедряйте CI/CD практики для автоматизации сборки, тестирования и развертывания приложения. Это позволяет быстро выявлять ошибки на ранних этапах разработки.

- Используйте инструменты для интеграции, такие как Jenkins, GitLab CI, или GitHub Actions, что позволит следить за изменениями кода и автоматически запускать тесты при каждом обновлении.

5. Управление зависимостями

- Обратите внимание на управление зависимостями между модулями, чтобы избежать проблем с версиями.

6. Мониторинг и поддержка

6.1 Логи и мониторинг

- Настройте системы мониторинга для отслеживания работы модулей после интеграции. Это поможет быстро реагировать на возникающие проблемы.

- Используйте инструменты для сбора и анализа логов для мониторинга и отладки.

6.2 Обратная связь и улучшения

Собирайте отзывы пользователей и разработчиков, чтобы выявить области для улучшения.

2.3. Выполнение отладки программного модуля с использованием специализированных программных средств

- Отладка — это процесс поиска и исправления ошибок в коде программы. Она включает в себя анализ поведения программы, проверку значений переменных, журналы событий и другие аспекты, чтобы выявить причину проблемы.

- Дебагеры — это основное средство для отладки. Они позволяют выполнить программу в пошаговом режиме, устанавливать точки останова (breakpoints) и исследовать состояние системы на каждом этапе выполнения.

Примеры дебагеров:

GDB (GNU Debugger) для C/C++

PDB (Python Debugger) для Python

Visual Studio Debugger для .NET

- Интегрированные среды разработки (IDE) - Это программа, в которой разработчики пишут, проверяют, тестируют и запускают код, а также ведут большие проекты.

Примеры IDE:

Visual Studio — поддерживает отладку для различных языков, включая C#, VB.NET и C++.

PyCharm — мощная среда для Python с удобными средствами отладки.

Eclipse — часто используется для Java, имеет встроенный дебаггер.

Пошаговое выполнение

После установки точек останова можно поэтапно выполнять код, чтобы наблюдать за изменением состояния программы. Инструменты отладки позволяют:

Выполнять строки кода по одной.

Пропускать определённые блоки кода.

Прокладывать путь выполнения программы в различных ветках (условия if, циклы и т.д.).

# 2.4. Осуществление разработки тестовых наборов и тестовых сценариев для программного обеспечения

1. Определение целей тестирования

• Выявление требований: Ознакомьтесь с функциональными и нефункциональными требованиями к программному обеспечению. Это поможет определить, что именно нужно тестировать.

• Определение типа тестирования: Решите, какие виды тестирования будут проводиться (функциональное, регрессионное, нагрузочное, пользовательское и т.д.).

2. Проектирование тестовых наборов

• Создание тестового набора: Тестовый набор — это коллекция тестов, которые проверяют определенные аспекты системы. Разделите тесты на категории, такие как:

• Функциональные тесты

• Тесты производительности

• Тесты безопасности

• Тесты совместимости

• Определение критериев успешности: Установите четкие критерии, по которым будет оцениваться успешность прохождения тестов.

3. Разработка тестовых сценариев

• Создание сценариев: Каждый тестовый сценарий должен описывать конкретный случай использования или функциональность, которую нужно проверить. Сценарий должен включать:

- Идентификатор сценария: Уникальный номер или название для удобства отслеживания.

- Описание: Краткое описание того, что будет тестироваться.

- Предусловия: Условия, которые должны быть выполнены перед запуском теста.

- Шаги выполнения: Пошаговое руководство по выполнению теста.

- Ожидаемый результат: Четкое описание того, что должно произойти, если система работает корректно.

4. Подготовка данных для тестирования

- Создание тестовых данных: Подготовьте данные, необходимые для выполнения тестов. Это могут быть как валидные, так и невалидные данные для проверки различных сценариев.

- Обеспечение конфиденциальности: Если используются реальные данные, убедитесь в соблюдении норм конфиденциальности и защиты данных.

5. Автоматизация тестирования (при необходимости)

- Выбор инструментов автоматизации: Если проект позволяет, рассмотрите возможность автоматизации повторяющихся тестов с помощью специализированных инструментов (например, Selenium, JUnit, TestNG).

- Создание автоматизированных сценариев: Напишите скрипты для автоматизации выполнения тестов, если это целесообразно.

6. Выполнение тестирования

- Запуск тестов: Выполните тесты согласно разработанным сценариям.

- Запись результатов: Зафиксируйте результаты выполнения каждого теста, включая успешные и неуспешные случаи.

7. Анализ результатов и отчетность

- Анализ результатов: Проанализируйте результаты тестирования для выявления ошибок и проблем.

- Создание отчетов: Подготовьте отчеты о проведенных тестах, включающие статистику успешности, найденные дефекты и рекомендации по их исправлению.

8. Обновление тестовой документации

- Корректировка сценариев: На основе полученных результатов обновите тестовые сценарии и наборы для учета новых требований или изменений в системе.

- Документирование процесса: Зафиксируйте весь процесс разработки и выполнения тестов для будущих ссылок.

2.5. Инспектирование компонент программного обеспечения на предмет соответствия стандартам кодирования

1. Проверка кода.

Инспектирование кода нужно для нескольких важных вещей:

- Улучшение качества: Оно помогает сделать код более аккуратным и понятным.

- Соблюдение правил: Если все следуют одним и тем же стандартам кодирования, код будет легче понимать всем в команде.

- Безопасность: Проверка кода помогает выявить уязвимости и защитить программу от атак.

- Упрощение поддержки: Чистый и понятный код гораздо легче поправить и изменить в будущем.

2. Стандарты кодирования

Стандарты кодирования — это набор правил, которые помогают разработчикам писать код понятнее. Эти правила могут включать:

- Как правильно называть переменные и функции.

- Как форматировать код (отступы, пробелы, где оставлять пустые строки).

- Как организовать файлы в проекте.

- Как добавлять комментарии, чтобы объяснять сложные части кода.

Примеры стандартов:

- PEP 8 — для языка Python.

- Google Java Style Guide — для Java.

-Airbnb Style Guide — для JavaScript.

3. Инспектирование кода

Ручная проверка: Один или несколько разработчиков внимательно смотрят на код, написанный другим. Это помогает выявить ошибки и обсудить, как можно лучше написать код.

# 3.) Выполняемые задания

Что было сделано во время производственной практики:

1. Репозиторий:

- Отчет (Report) - Отчет по практике

- Задания (Src) - Программный код

- Документы (Docs) - Данные практики

3.1 Техническое задание на разработку модулей.

3.2 Стандарт кодирования PEP 8.

3.3 Инспектирование разрабатываемого программного кода.

3.4 Интеграция Модулей.

3.5 Отладка модулей.

3.6 Тестирование программных модулей

3.1) Техническое задание на разработку модулей.

Производная практика.

Техническое задание на разработку модулей.

Работу выполнил, практикант группы ИС-23

Смирнов Руслан

Череповец 2024

Оглавление

1. Введение
2. Основания для разработки
3. Назначение разработки
4. Требования к программе или программному изделию
5. Требования к программной документации
6. Технико-экономические показатели
7. Стадии и этапы разработки
8. Порядок контроля и приемки
9. Заключение

Введение

Современные технологии обработки изображений играют важную роль в различных областях, таких как искусственный интеллект, компьютерное зрение и веб-разработка. Разработка модулей для работы с изображениями и взаимодействия с пользователем позволит значительно упростить процессы обработки данных и повысить эффективность работы с визуальной информацией. Настоящее техническое задание описывает требования к разработке двух модулей: модуля обработки изображений и модуля взаимодействия с пользователем.

Основания для разработки

Разработка модулей основана на потребностях заказчика в автоматизации процессов обработки изображений и упрощении взаимодействия с пользователем. В условиях растущего объема визуальной информации и необходимости быстрого извлечения данных из изображений, создание таких модулей является актуальным и необходимым шагом.

Назначение разработки

1. Модуль обработки изображений

- Функциональность: Этот модуль будет отвечать за анализ изображений, включая извлечение метаданных, таких как размер, разрешение и дата создания.

- Переименование файлов: Пользователи смогут переименовывать изображения по заданным критериям, что упростит организацию и поиск файлов.

2. Модуль пользовательского интерфейса

- Интерфейс: Модуль обеспечит дружелюбный и интуитивный интерфейс для взаимодействия с пользователями.

- Ввод данных: Пользователи смогут легко вводить пути к изображениям и получать результаты обработки в удобном формате.

- Отображение результатов: Модуль будет выводить информацию о изображениях, позволяя пользователям быстро оценивать результаты работы системы.

Требования к модулю обработки и работы с изображениями:

- Извлечение информации об изображении:

- Размер изображения в байтах.

- Разрешение изображения (ширина x высота в пикселях).

- Дата создания файла (формат: ГГГГ-ММ-ДД ЧЧ:ММ:СС).

- Переименование изображения по заданному пути.

Нефункциональные требования:

- Язык программирования: Python.

- Обработка ошибок: модуль должен корректно обрабатывать ситуации, когда указанный файл не существует или не является изображением.

Требования к модулю взаимодействия с пользователем и формирование и хранений данных

- Предоставление пользователю интерфейса для ввода пути к изображению (консольный или графический).

- Передача введенного пути к модулю обработки изображений.

- Вывод результатов обработки в удобочитаемом формате.

Нефункциональные требования

- Простота использования интерфейса.

- Документирование всех функций модуля.

Общие требования

- Установленная библиотека pillow

- Формат изображение JPG

Требования к программной документации

Документация должна включать:

- Описание архитектуры системы.

- Руководство пользователя с примерами использования.

- Техническую документацию для разработчиков, включая описание API модулей.

Стадии и этапы разработки

1. Анализ требований и проектирование архитектуры модулей.

2. Разработка модуля обработки изображений.

3. Разработка модуля обработки изображений.

4. Разработка модуля взаимодействия с пользователем.

5. Тестирование модулей на соответствие требованиям.

6. Внедрение

Порядок контроля и приемки

- Контроль за выполнением этапов разработки будет осуществляться на каждом этапе через промежуточные отчеты.

- Приемка будет проводиться по завершении тестирования, при условии выполнения всех функциональных требований.

Заключение

Разработка модулей обработки изображений и взаимодействия с пользователем является важным шагом к автоматизации и оптимизации работы с графическими данными. В соответствии с представленным техническим заданием, предполагается создание двух взаимосвязанных модулей, которые обеспечат пользователям удобный и эффективный интерфейс для работы с изображениями.

3.2) PEP 8 — это стандарт кодирования для кода на Python. Он описывает рекомендации по стилю, форматированию и организации кода для повышения читаемости и согласованности. Хотя PEP 8 не является строгим стандартом, которому нужно следовать дословно, придерживаться его рекомендаций сильно улучшает качество кода и делает его более понятным для других разработчиков (и для вас самих в будущем).

Вот основные моменты PEP 8:

1. Общее форматирование:

Отступы: Используйте 4 пробела для отступов. Не смешивайте пробелы и табуляцию.

Максимальная длина строки: Ограничьте длину строк до 79 символов. Для docstrings — 72 символа. Длинные строки можно разбить на несколько более коротких с использованием обратного слеша (\). Лучше использовать неявное разбиение строк с помощью круглых скобок.

Пустые строки: Используйте пустые строки для разделения логических блоков кода.

Комментарии: Пишите ясные и лаконичные комментарии. Не комментируйте очевидное. Комментарии должны быть актуальными и обновляться вместе с кодом.

2. Имена:

Имена переменных: Используйте нижний регистр с разделителями в виде подчеркивания (snake\_case) для переменных (например, my\_variable).

3. Импорт:

Порядок импортов: Сначала импортируйте стандартные библиотеки, затем сторонние библиотеки, затем локальные модули. Каждый блок импортов должен быть отделен пустой строкой.

4. Операторы:

Пробелы вокруг операторов: Ставьте пробел вокруг операторов =, +, -, \*, /, //, %, \*\*, <, >, <=, >=, ==, !=.

Пробелы после запятых: Ставьте пробел после запятых.

Двойной знак равно: Используйте == для сравнения, а не = (присваивание).

6. Другие рекомендации:

Один оператор на строку: Лучше не объединять несколько операторов на одной строке.

Вложенность: Избегайте чрезмерной вложенности. Если код становится слишком сложным, разбейте его на более мелкие функции.

Именование: Используйте осмысленные и описательные имена переменных, функций и классов.

3.3) Провести инспектирование разрабатываемого программного кода:

1)Модуль обработки и работы с изображениями

1. Выдача информации о изображении

- Именование функций и переменных: Все функции и переменные должны быть названы в нижнем регистре с использованием подчеркиваний для разделения слов (например, get\_image\_info, image\_size).

- Отступы: Используйте 4 пробела для отступов. Убедитесь, что нет смешивания табуляций и пробелов.

- Длина строки: Длина строк не должна превышать 79 символов.

- Докстринги: Каждая функция должна иметь документацию в виде строки документации (docstring), объясняющей её назначение, параметры и возвращаемое значение.

- Импорт библиотек: Импортируйте библиотеки в начале файла, соблюдая порядок: стандартные библиотеки, сторонние библиотеки, локальные модули.

- Читаемость кода: Код должен быть легко читаемым и понятным. Используйте осмысленные имена переменных и функций (например, rename\_image\_file).

- Обработка ошибок: Обработка исключений должна быть реализована с использованием try-except блоков. Исключения следует логировать или выводить пользователю с понятным сообщением.

- Форматирование строк: Используйте f-строки (если используете Python 3.6 и выше) для форматирования строк вместо конкатенации.

- Валидация пути: Проверка на существование файла должна быть реализована с использованием os.path.exists(), и результат должен быть обработан корректно.

- Комментирование: Важные участки кода следует комментировать, чтобы другие разработчики могли понять логику.

3.4) Интеграция модулей(Рисунок.3,4):

Метод Событийно-ориентированная интеграция - подход в разработке программного обеспечения и систем, который основывается на обработке событий как основного механизма взаимодействия между компонентами системы. В этой архитектуре компоненты системы реагируют на события, которые могут происходить как внутри системы, так и извне.

Модуль обработки изображений

Создадим модуль image\_processor.py, который будет отвечать за обработку изображений.

Модуль взаимодействия с пользователем

Создадим модуль user\_interaction.py, который будет взаимодействовать с пользователем и передавать данные в модуль обработки изображений.

3.5) Отладка:

При отладке было выполнено:

1. Анализ логов: Проверить логи обоих модулей на наличие ошибок и предупреждений, чтобы выявить возможные проблемы.

2. Тестирование функций: Написать и выполнить юнит-тесты для ключевых функций обработки изображений и взаимодействия с пользователем, чтобы убедиться в их корректной работе.

3. Проверка интерфейсов: Убедиться, что интерфейсы между модулями корректно определены и данные передаются без потерь или искажений.

4. Профилирование производительности: Использовать инструменты профилирования для выявления узких мест в производительности, особенно в модуле обработки изображений.

5. Пользовательское тестирование: Провести тестирование с участием пользователей для получения обратной связи о взаимодействии с интерфейсом.

6. Ревизия кода: Провести код-ревью для выявления потенциальных ошибок и улучшения читаемости кода.

3.6) тестирование программных модулей:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Наименование  теста | Тестовые  данные | Ожидаемый  результат | Фактический  результат | Результат  тестирования | Комментарий |
| Test1 | Ширина: 961  Длина: 686 | Информация об изображении:  Размер: (961, 686)  Режим: RGB  Дата создания: 2024-11-20 22:52:18  Разрешение: (94, 94)  меняет название изображения | Информация об изображении:  Размер: (961, 686)  Режим: RGB  Дата создания: 2024-11-20 22:52:18  Разрешение: (94, 94)  меняет название изображения | Код работает отлично, без ошибок, но принимает только .Jpg |  |
| Test2 | Ширина: 961  Длина:  686 | Получает и отображает информацию об изображении.  Запрашивает путь к изображению у пользователяю  Выводим словарь с информацией | Получает и отображает информацию об изображении.  Запрашивает путь к изображению у пользователяю  Выводим словарь с информацией | Код работает отлично. |  |

Заключение

Производственная практика позволила мне расширить свои знания Python, результатом чего стало создание двух взаимодействующих модулей: обработки изображений и взаимодействия с пользователем

Список использованных источников

1. работа с Pillow - <https://pythonru.com/biblioteki/osnovnye-vozmozhnosti-biblioteki-python-imaging-library-pillow-pil>

2. Os - <https://docs.python.org/3/library/os.html>

3. Работа с модулями - <https://metanit.com/python/tutorial/2.10.php>

4. ГОСТ 7.80-2000 «Библиографическая запись. Заголовок. Общие

требования и правила составления» [Электронный ресурс]/ Электронный

правовой и нормативно-технической документации- режим доступа: <http://docs.cntd.ru/document/gost-7-80-2000>

5. ГОСТ 7.32-2001 «Отчет о научно-исследовательской работе. Структура и правила оформления» [Электронный ресурс]/ Электронный правовой и нормативно-технической документации- режим доступа:

<http://docs.cntd.ru/document/gost-7-32-2001-sibid>

6. ГОСТ 7.1-2003 № 332-ст «Система стандартов по информации, библиотечному и издательскому делу. Библиографическая запись. Библиографическое описание. Общие требования и правила составления»

[Электронный ресурс]/ Электронный правовой и нормативно-технической

документации- режим доступа: <http://docs.cntd.ru/document/gost-7-1-2003-sibid>

7. ГОСТ Р 7.0.5-2008 «Система стандартов по информации, библиотечному и издательскому делу. Библиографическая ссылка. Общие требования и правила составления» [Электронный ресурс]/ Библиотека ГОСТов стандартов и нормативов- режим доступа: <http://www.standartov.ru/norma_doc/53/53649/index.htm>

8. ГОСТ Р 7.0.11-2011 «Система стандартов по информации, библиотечному и издательскому делу. Диссертация и автореферат диссертации. Структура и правила оформления» [Электронный ресурс]/ Электронный правовой и нормативно-технической документации- режим доступа: <http://docs.cntd.ru/document/gost-r-7-0-11-2011>

9. Официальный сайт Microsoft Office [Электронный ресурс]- режим

доступа: <https://products.office.com/ru-RU/>

10. Сайт компании нанимателя- <https://www.mallenom.ru/>

11. PEP8 - <https://peps.python.org/pep-0008/>

12. Отладка - <https://habr.com/ru/companies/vk/articles/205426/>

Приложения



Рисунок 1



Рисунок 2

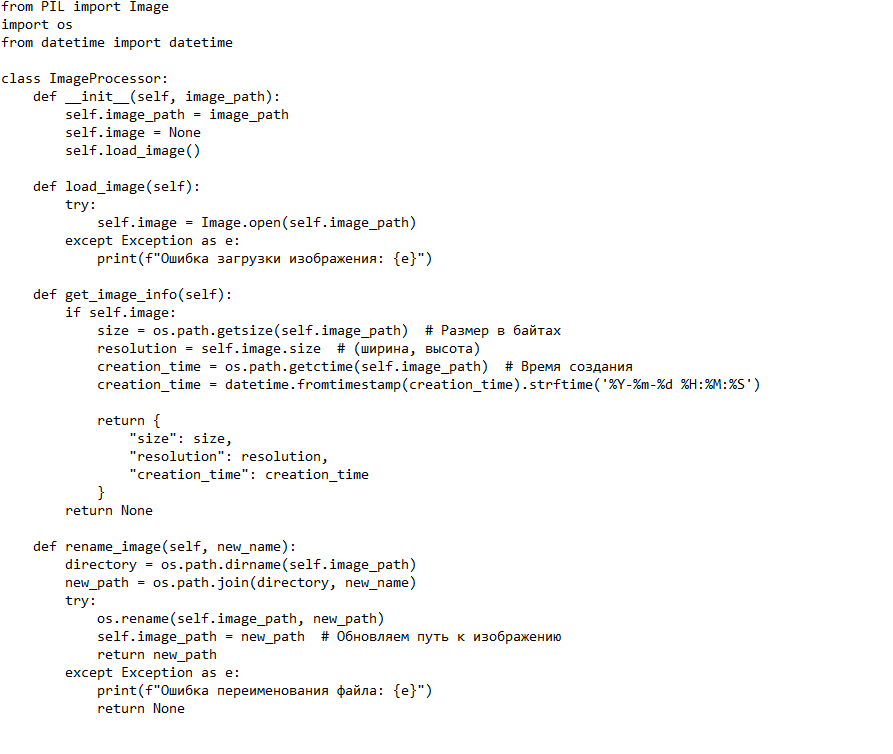


Рисунок 3

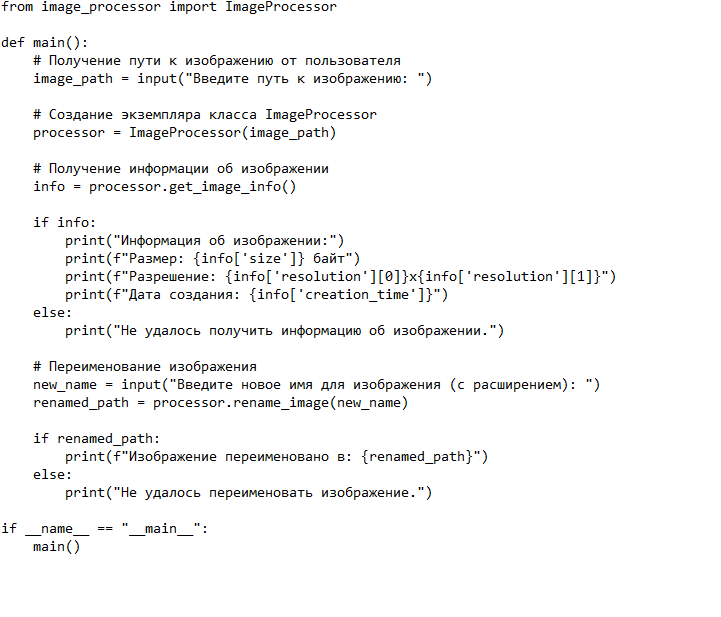


Рисунок 4