МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

РОСТОВСКИЙ ГОСУДАРСТВЕННЫЙ ЭКОНОМИЧЕСКИЙ УНИВЕРСИТЕТ (РИНХ)

**Факультет Компьютерных технологий и информационной безопасности**

**Кафедра Информационных технологий и программирования**

**Индивидуальная работа**

**по дисциплине**

**«Инженерное программирование»**

на тему:

**«Разработка утилиты на Python для загрузки и хранения видео через прокси-сервер»**

Выполнил:

студент гр. ПРИ-311

Направление 09.03.04 Программная инженерия Д. В. Маршалкин

Руководитель:

доцент Е. В. Жилина

Дата сдачи работы:

Дата защиты:

Оценка:

Ростов-на-Дону

2025

### ****Введение****

В современном цифровом пространстве передача, хранение и обработка мультимедийного контента, в частности видеоматериалов, становятся неотъемлемой частью информационных технологий. Видеофайлы занимают значительный объем в общем интернет-трафике, а их доступность и качество воспроизведения напрямую зависят от используемых методов загрузки, хранения и передачи данных.

С увеличением количества пользователей, работающих с видеоконтентом, возникает потребность в разработке специализированных инструментов, позволяющих эффективно загружать, сохранять и воспроизводить видеоматериалы. Это особенно актуально для случаев, когда пользователи хотят управлять своими видеофайлами, минимизируя зависимость от сторонних платформ, а также обеспечивать доступность контента для последующего просмотра.

В данном индивидуальном проекте рассматривается разработка программного решения на языке Python, которое позволит загружать видеоролики через прокси-сервер, сохранять их на выделенном сервере и обеспечивать пользователям доступ к загруженному контенту. Архитектура проекта включает в себя два сервера:

**- Первый сервер** выполняет роль посредника — он подключается к видеохостингам, загружает файлы и передает их на второй сервер.

**- Второй сервер** отвечает за хранение видеофайлов и предоставление удобного интерфейса для их последующего просмотра.

Такой подход обеспечивает гибкость и надежность работы с видеоконтентом, минимизируя нагрузку на клиентские устройства, упрощая процесс управления файлами и повышая удобство использования системы.

Проект ориентирован на студентов и преподавателей, занимающихся изучением сетевых технологий, распределенных систем и мультимедийных данных. Он также демонстрирует применение языка Python в области автоматизированной обработки видеоконтента, что делает его полезным не только в образовательных, но и в практических целях.

**Актуальность тематики исследования**

В современном мире видеоконтент играет ключевую роль в различных сферах деятельности, включая образование, науку, медиаиндустрию и информационные технологии. С увеличением объема видеоматериалов и их повсеместным распространением пользователи сталкиваются с рядом сложностей, связанных с загрузкой, хранением и воспроизведением видеофайлов.

- Во-первых, видеофайлы требуют значительных вычислительных ресурсов и стабильного интернет-соединения. Низкая пропускная способность сети, ограничения со стороны интернет-провайдеров, а также географические или организационные блокировки доступа к видеохостингам создают серьезные препятствия для пользователей.

- Во-вторых, централизованное хранение видеоконтента на сторонних платформах приводит к зависимости от их политики, технических ограничений и возможных сбоев. Это особенно актуально для образовательных и исследовательских проектов, где необходима надежная и контролируемая система хранения видеоданных.

- В-третьих, отсутствие удобных инструментов для автоматизированной загрузки и управления видеоконтентом усложняет процесс работы с большими объемами мультимедийных данных.

Такое решение позволит:

- Оптимизировать процесс загрузки видео за счет использования промежуточного сервера, снижая нагрузку на клиентские устройства.

- Обеспечить удобное хранение и доступ к видеофайлам без риска их удаления или блокировки.

- Гибко адаптировать систему под различные сценарии использования, включая образовательные, архивные и исследовательские задачи.

Таким образом, данное исследование направлено на решение одной из актуальных проблем информационных технологий, связанных с эффективным управлением видеоконтентом. Разработанная утилита может быть востребована как в образовательных учреждениях, так и в других сферах, где необходим доступ к видеоинформации в условиях ограниченного интернет-доступа или блокировок.

### ****Технологический стек проекта и теоретическое обоснование****

Разрабатываемый проект предназначен для загрузки видео через прокси-сервер и их последующего хранения на выделенном сервере. Для успешной реализации системы был выбран стек технологий, обеспечивающий надежность, масштабируемость и удобство работы с мультимедийными данными.

## **1. Теоретическая часть**

### ****1.1. Предметная область****

В современном цифровом мире видеоконтент играет ключевую роль в различных сферах: медиа, образовании, науке, маркетинге и развлечениях. Однако пользователи сталкиваются с рядом проблем:

**- Ограниченный доступ** – некоторые видеохостинги могут блокироваться в определенных регионах.

**- Высокая нагрузка на клиентские устройства** – загрузка и хранение видеофайлов требуют значительных ресурсов.

**- Проблемы с устойчивостью соединения** – нестабильные сети могут приводить к прерыванию загрузки.

Для решения этих задач разрабатываются инструменты, которые позволяют загружать, хранить и управлять видеоконтентом через прокси-серверы и удаленные хранилища.

### ****1.2. Анализ существующих решений****

На данный момент существует несколько категорий программных решений для работы с видеоконтентом. На рисунке 1 сравнение и **анализ существующих решений.**

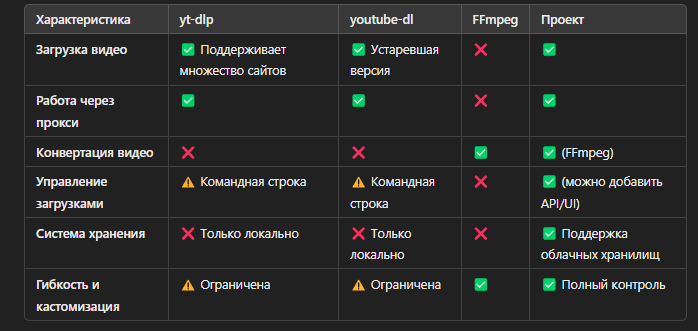


Рисунок 1 – сравнение и **анализ существующих решений**

Вывод:

- yt-dlp и youtube-dl удобны для разовых загрузок, но не предназначены для хранения и автоматизации.

- FFmpeg используется для обработки видео, но не поддерживает скачивание.

- Разрабатываемый проект объединяет возможности загрузки, прокси, хранения и обработки видео в одном решении.

**2. Технологический стек проекта**

### ****2.1. Язык программирования – Python****

Python был выбран из-за его гибкости и большого количества библиотек:

- requests**,** aiohttp – работа с HTTP-запросами и прокси.

- yt-dlp – загрузка видео с видеохостингов.

- ffmpeg-python – обработка и конвертация видеофайлов.

- paramiko – работа с удалёнными серверами по SSH.

### ****2.2. Серверная инфраструктура****

Используются два сервера:

1. **Основной сервер (Linux)** – для хранения и обработки видео.
2. **Второй сервер (TBD)** – в зависимости от задач (облако, база данных и т. д.).

### ****2.3. Система хранения видео****

**- Локальная файловая система** – для быстрого доступа.

**- Облачное хранилище (S3, MinIO)** – для масштабируемости.

**- SFTP/WebDAV** – для передачи файлов.

### ****2.4. Обработка видео****

**- FFmpeg** – кодирование, изменение форматов, сжатие.

**- Pymediainfo** – получение метаданных.

### ****2.5. Веб-интерфейс и API****

**- Flask / FastAPI** – для REST API.

**- Django** – если потребуется сложная система.

**- React / Vue.js** – если понадобится веб-интерфейс.

### ****2.6. Очереди задач****

**- Celery + Redis** – асинхронные задачи.

**- RQ (Redis Queue)** – более простая альтернатива.

### ****2.7. Контейнеризация и деплой****

**- Docker / Docker Compose** – удобное развертывание.

### ****2.8. Мониторинг и логирование****

Для успешной разработки и поддержки проекта крайне важно отслеживать производительность системы, а также эффективно логировать все ошибки и события, которые могут возникнуть в процессе работы. В этом контексте использование GitHub и других инструментов играет важную роль в организации мониторинга и логирования.

#### **Использование GitHub для мониторинга**

1. **GitHub Actions**:

**- Автоматизация CI/CD процессов**: GitHub Actions позволяет автоматизировать процесс тестирования, сборки и деплоя проекта. Каждое изменение в репозитории может инициировать автоматическое выполнение тестов, что гарантирует стабильность кода.

**- Мониторинг билдов**: GitHub предоставляет возможность отслеживать состояние каждого билда и теста, включая успешные и неудачные сборки. Это помогает своевременно выявлять проблемы на ранних этапах разработки.

1. **GitHub Issues и Pull Requests**:

**- Использование Issues для отслеживания багов**: Для более удобного отслеживания проблем и дефектов можно использовать GitHub Issues. Каждую проблему можно категоризовать, назначать ответственным и отслеживать процесс решения.

**- Pull Requests для контроля изменений**: При внесении изменений в код через PR можно отслеживать, кто и какие изменения внес, а также оставить комментарии для обсуждения с командой. Это помогает сохранять прозрачность и эффективность в управлении проектом.

1. **Интеграция с внешними инструментами для мониторинга**:

**- Sentry**: Интеграция с Sentry позволяет отслеживать ошибки в реальном времени, отправляя уведомления о возникших проблемах.

**- Prometheus + Grafana**: Для более глубокого мониторинга и визуализации производительности можно использовать Prometheus и Grafana. Эти инструменты позволяют собирать метрики и строить графики, что помогает понять, какие части системы требуют оптимизации.

#### **Логирование**

Для качественного логирования можно использовать такие решения, как:

**- Loguru**: Это библиотека для Python, которая обеспечивает простое и гибкое логирование. Она поддерживает уровни логирования (info, warning, error, critical), форматирование и отправку логов в различные системы.

**- Elasticsearch + Kibana**: Для продвинутого логирования можно настроить Elasticsearch и Kibana для хранения и визуализации логов, что поможет эффективно отслеживать состояние проекта в реальном времени.

### ****2.9. Графические средства оформления и текста (Figma + Photoshop)****

Кроме того, важным аспектом проекта является создание графических материалов, таких как интерфейсы, схемы и изображения. Для этого удобно использовать инструменты дизайна, которые позволяют создавать высококачественные визуальные элементы.

#### **Figma**

**Figma** — это современный инструмент для создания UI/UX дизайнов, который предлагает следующие преимущества:

**- Совместная работа**: Все изменения в проекте видны в реальном времени, что позволяет команде работать одновременно над проектом, независимо от их местоположения.

**- Прототипирование**: Возможность быстро создавать прототипы интерфейсов и тестировать их на пользователях.

**- Гибкость**: Figma поддерживает векторную графику, что позволяет создавать адаптивные интерфейсы и элементы.

**- Библиотеки и плагины**: Удобные библиотеки компонентов и сторонние плагины для расширения возможностей работы.

**Преимущества Figma:**

**- Легкость в использовании**: Доступность всех инструментов для начинающих и профессионалов.

**- Работа в браузере**: Не нужно устанавливать дополнительные приложения, так как все работает в браузере.

**- Облачное хранилище**: Все работы сохраняются в облаке, что делает их доступными для всех участников проекта.

#### **Photoshop**

**Photoshop** — это мощное приложение для редактирования растровой графики, которое широко используется для создания высококачественных изображений. В проекте Photoshop может быть использован для:

**- Обработки изображений**: Изменение размеров, коррекция цвета, добавление текстур и других элементов.

**- Создания визуальных эффектов**: С помощью Photoshop можно создавать сложные графические элементы, которые могут быть использованы в интерфейсе или маркетинговых материалах.

**- Векторная графика**: Хотя Photoshop в основном используется для работы с растровыми изображениями, он также поддерживает создание векторных объектов для создания иконок и элементов интерфейса.

**Преимущества Photoshop:**

**- Мощные инструменты**: Photoshop предоставляет обширный набор инструментов для детальной работы с изображениями.

**- Высокая точность**: Программное обеспечение позволяет работать с пиксельной точностью, что важно для дизайнеров.

**- Интеграция с другими продуктами Adobe**: Легкая интеграция с Illustrator, After Effects и другими программами Adobe для создания комплексных проектов.

### ****2.10. Канбан-доска для управления задачами (YouGile)****

Для эффективного управления задачами в проекте используется **YouGile** — современная и удобная канбан-доска, которая позволяет отслеживать прогресс работы, организовывать задачи и назначать ответственных. В YouGile удобно визуализировать рабочий процесс, что помогает команде быстрее достигать целей и эффективно решать возникающие задачи.

#### **Преимущества использования YouGile**:

1. **Гибкость в настройке**:

- YouGile позволяет настроить доски под различные этапы разработки и легко адаптировать их под нужды команды. Можно настроить категории, колонки и фильтры для более детальной организации задач.

1. **Интуитивно понятный интерфейс**:

- Доски в YouGile имеют простой и понятный интерфейс, что позволяет быстро организовывать задачи. Перетаскивание карточек между колонками помогает наглядно отслеживать текущий статус каждой задачи.

1. **Совместная работа**:

- YouGile позволяет команде работать совместно, обсуждать задачи, оставлять комментарии и загружать файлы. Это помогает улучшить коммуникацию внутри команды и ускоряет решение возникающих проблем.

1. **Уведомления и интеграция с другими сервисами**:

- YouGile поддерживает уведомления о статусах задач, что помогает держать всех участников проекта в курсе изменений. Также инструмент может быть интегрирован с другими сервисами, такими как GitHub или Slack, для улучшения взаимодействия с кодом и командой.

1. **Простота в использовании**:

- Платформа имеет интуитивно понятный интерфейс и не требует длительного обучения для работы. Это позволяет команде сосредоточиться на решении задач, а не на освоении инструмента.

#### **Использование YouGile в проекте**:

1. **Управление задачами**:

- С помощью YouGile можно организовать задачи по стадиям разработки, таким как "To Do", "In Progress" и "Done". Это помогает визуализировать и отслеживать прогресс проекта.

1. **Назначение ответственных**:

- Для каждой задачи можно назначить ответственного разработчика, что помогает распределить нагрузку и гарантировать, что каждый этап работы будет выполнен своевременно.

1. **Отчеты и аналитика**:

- YouGile предоставляет простые средства для генерации отчетов и анализа выполнения задач, что позволяет команде и руководителям проекта видеть, как идет работа и где могут возникнуть проблемы.

**2.11. Блок-схема проекта**

Блок-схема проекта визуализирует основные процессы работы системы и взаимодействие её компонентов. Она помогает понять логику работы утилиты, упрощает разработку и облегчает дальнейшую поддержку кода. На рисунке 2 представлена

блок-схема проекта.

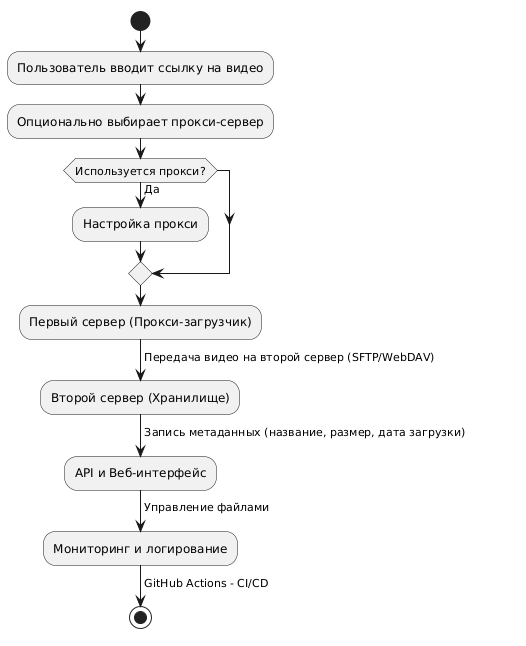


Рисунок 2 – блок-схема проекта

**Основные элементы блок-схемы:**

1. Начало работы (ввод данных пользователем)

- Пользователь вводит URL видео и при необходимости указывает прокси-сервер.

- Эти данные передаются в систему для дальнейшей обработки.

1. Первый сервер – Прокси-загрузчик

- Подключается к видеохостингу через указанный прокси.

- Загружает видеофайл с помощью библиотеки yt-dlp.

- Проверяет корректность загрузки (наличие ошибок, статус HTTP-запросов).

- (Опционально) Обрабатывает видео (ffmpeg-python), например, меняет формат или снижает качество.

- После успешной загрузки передаёт файл на второй сервер хранения.

1. Второй сервер – Хранилище видео

- Получает файл от прокси-загрузчика.

- Сохраняет видео локально или в облачное хранилище (S3, MinIO).

- Записывает метаданные (размер, название, дата загрузки).

- Позволяет пользователю получать список файлов и скачивать видео через API.

1. API и Веб-интерфейс

- Обеспечивает доступ к загруженным видео через Flask/FastAPI.

- Позволяет пользователю просматривать, скачивать и управлять файлами.

- При необходимости можно добавить React/Vue.js для веб-интерфейса.

1. Мониторинг и логирование

- Все процессы логируются через Loguru (ошибки, успешные загрузки).

- Для мониторинга работы серверов используются Prometheus и Grafana.

- CI/CD процессы автоматизируются с помощью GitHub Actions.

Зачем нужна блок-схема?

- Упрощает понимание архитектуры проекта.

- Позволяет разработчикам быстрее вникнуть в работу системы.

- Помогает выявить потенциальные узкие места и улучшить структуру проекта.

- Облегчает поддержку и масштабирование системы.

Блок-схема демонстрирует логический поток данных в системе, обеспечивая чёткую и понятную структуру проекта.

**3. Разработка программного продукта**

**3.1 Инструментальные и программные средства разработки**

**3.1.1 Инструментальные средства разработки**

Задача заключалась в разработке программного продукта для упрощения загрузки видео и аудио. Программа должна была предоставлять интуитивно понятный графический интерфейс, позволяющий пользователю вводить ссылку на видео, выбирать качество и формат загрузки, просматривать и сохранять миниатюру (превью), а также вести лог всех операций. Целью разработки было создание удобного инструмента, подходящего как для технически неподготовленных пользователей, так и для продвинутых. В рамках проекта использовались следующие программные средства:

Visual Studio Code (VS Code) — это кроссплатформенный текстовый редактор с открытым исходным кодом от Microsoft, который предоставляет широкие возможности для разработки программ. Благодаря богатой экосистеме расширений и поддержке отладки, автодополнения кода и интеграции с системами контроля версий, VS Code стал идеальной средой разработки для проекта. Он особенно удобен при работе с Python, поскольку позволяет настраивать окружения, управлять зависимостями и запускать скрипты прямо из редактора.

В качестве языка программирования был выбран **Python**, благодаря его простоте и широким возможностям. Основной функционал скачивания видео реализован с использованием библиотеки yt-dlp — это активно поддерживаемый форк youtube-dl, предоставляющий расширенные возможности загрузки с различных видеохостингов.  
Библиотека позволяет получать информацию о видео, выбирать доступные форматы, а также управлять процессом загрузки. Для конвертации медиафайлов использовалась интеграция с ffmpeg, путь к которому указывается вручную в коде.

Для создания графического интерфейса была использована стандартная библиотека tkinter, поставляемая вместе с Python. Интерфейс включает элементы управления вводом (текстовые поля, выпадающие списки, кнопки), просмотр миниатюры видео, а также окно для вывода логов, что делает взаимодействие с программой удобным и наглядным.

Дополнительно использовались библиотеки Pillow и requests для обработки изображений и получения превью видео с YouTube.  
Пользователь может выбрать разрешение миниатюры (например, maxresdefault), просмотреть её в окне приложения и при желании сохранить в локальную папку. Обработка изображений выполняется средствами Pillow, а загрузка превью — через requests.

Весь процесс разработки велся с использованием системы контроля версий Git, а размещение проекта и отслеживание изменений осуществлялось через платформу GitHub.  
Использование GitHub обеспечило:

* контроль версий и историю изменений;
* возможность отката к стабильным версиям проекта;
* прозрачность процесса разработки;
* резервное копирование кода;
* публикацию проекта в открытый доступ для обратной связи и демонстрации возможностей.

Также проект оформлен с использованием файла README.md, в котором описана инструкция по запуску, список зависимостей и скриншоты интерфейса. Это делает приложение легко устанавливаемым и понятным для конечного пользователя. На рисунке 3-4 представлены скриншоты репозитория и коммитов.

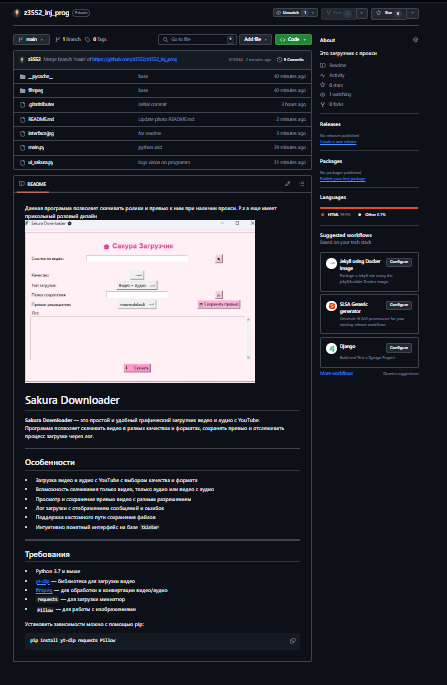


Рисунок 3 – Скриншот репозитория GitHub.

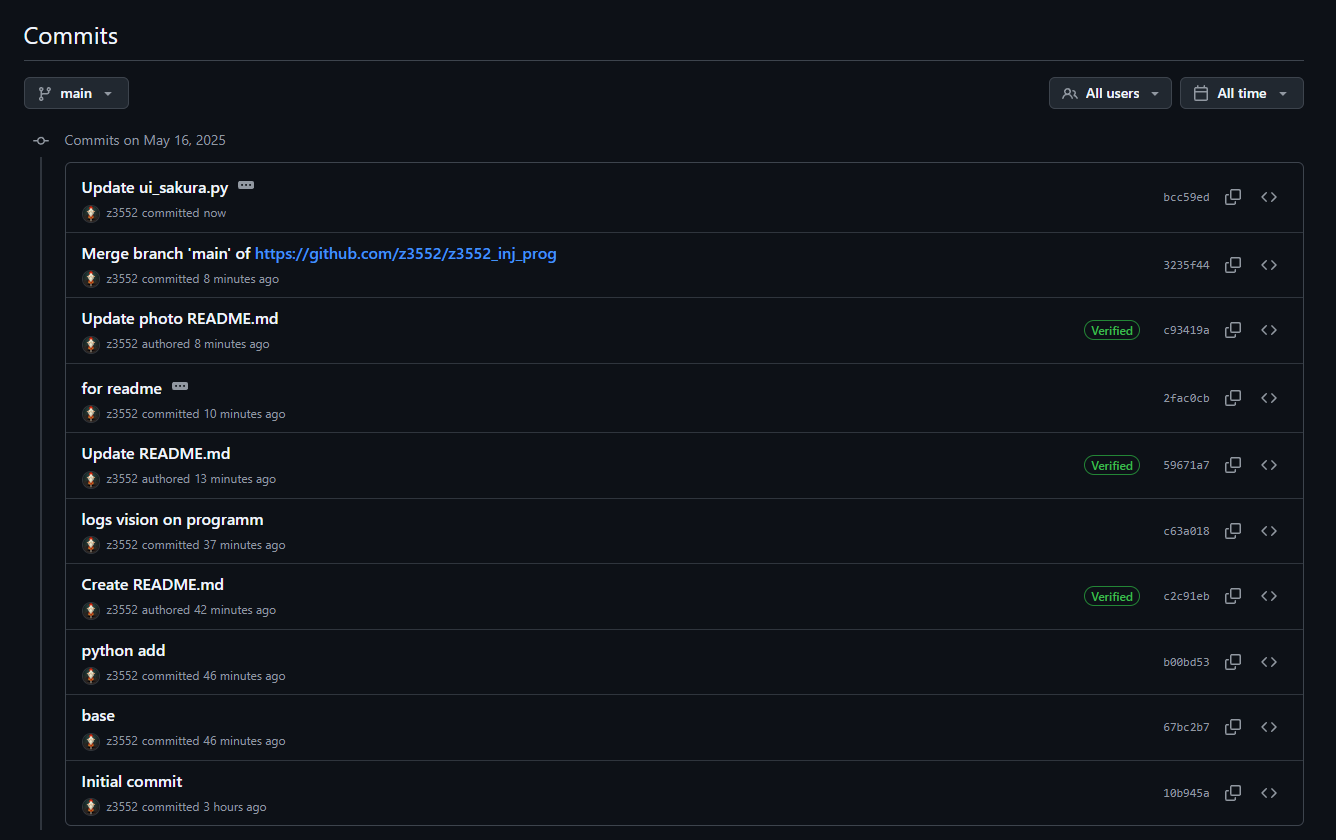


Рисунок 4 – Скриншот коммитов GitHub

Yougile представляет собой инструмент управления проектами, который получил множество положительных отзывов от пользователей. Вот несколько причин, по которым Yougile может быть хорошим выбором для управления проектами:

- удобный интерфейс: пользователи отмечают, что интерфейс Yougile удобен в использовании и разработчики учитывают пожелания пользователей.

- поддержка: сервис предоставляет быструю и качественную поддержку пользователей.

- высокая стабильность: пользователи сообщают о высокой стабильности работы приложения, как настольной, так и веб-версии.

- ценовая политика: для небольших компаний Yougile предлагает бесплатный тариф без ограничений, что является большим плюсом.

- отечественный продукт: Yougile является примером качественного российского софта с активной поддержкой.

Эти аспекты делают Yougile привлекательным выбором для тех, кто ищет эффективное и надежное решение для управления проектами. На рисунке 5 представлена доска Yougile.

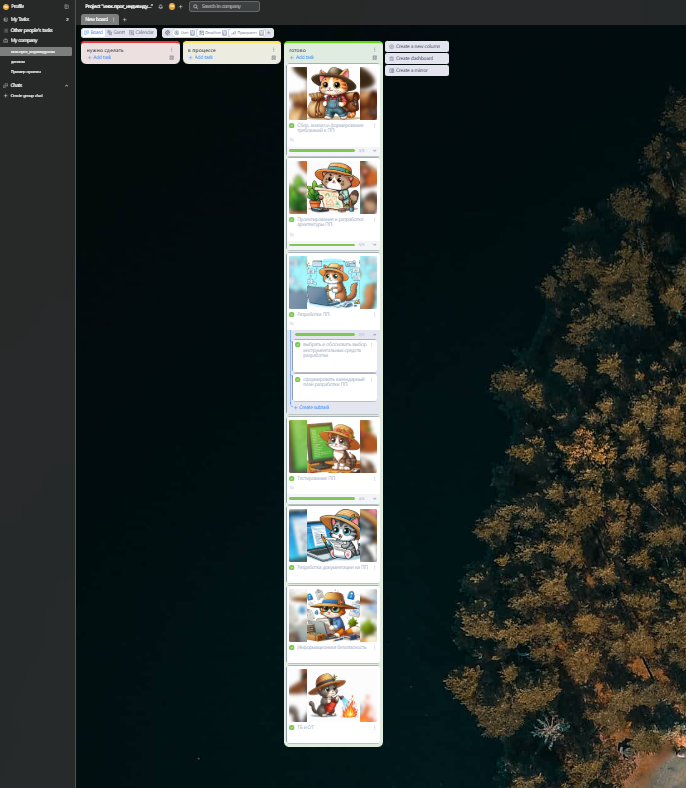


Рисунок 5 -Доска Yougile

4. Тестирование программного продукта

4.1 Выбор метода обеспечения качества

Для гарантии высокого качества отдельных компонентов программного продукта, было необходимо провести тестирование всех возможных вариантов использования и некоторых элементов. Ключевым аспектом является то, чтобы эти компоненты функционировали и вели себя в соответствии с первоначальным замыслом, обеспечивая беспроблемное использование для конечного пользователя.

4.2 Тестирование продукта

Для обеспечения качества и стабильной работы программного продукта Sakura Downloader был выбран метод ручного тестирования. Такой подход позволил эмулировать реальные сценарии использования программы в среде Visual Studio Code с последующим выполнением и отладкой кода на Python. Ручное тестирование особенно эффективно в GUI-приложениях, поскольку позволяет напрямую взаимодействовать с интерфейсом, оценивать корректность отображения данных и поведение при различных пользовательских действиях.

Программа активно использует библиотеку yt-dlp для загрузки мультимедийного контента, а также инструменты работы с сетью (requests) и изображениями (PIL). Особое внимание в ходе тестирования уделялось корректной работе многопоточности, стабильности загрузки, обработке ошибок и взаимодействию с внешними ресурсами (YouTube, файловая система).

Был разработан тестовый план, охватывающий следующие ключевые модули и пользовательские сценарии,которые отображены в таблице 1.

Таблица 1 - Тест-план для проведения тестирования проекта

|  |  |
| --- | --- |
| Вид тестирования | Способ тестирования |
| Тестирование функционала пользователя | Ручное |

Тест-кейсы для функционала владельца представлены в таблице 2.

Таблица 2 -Тест-кейс

|  |  |  |
| --- | --- | --- |
| Название: | Тест получения информации о видео и отображения превью | |
| Функция: | Загрузка метаданных видео с YouTube и отображение изображения превью | |
| Действие: | Ожидаемый результат | Результат теста:  выполнен |
| Предусловие: | Пользователь открыл программу Sakura Downloader | пройден |
| Ввести корректную ссылку на видео YouTube | Отображается название видео, список доступных форматов, загружается изображение превью | пройден |
| Изменить разрешение превью на другое (например, с maxresdefault на hqdefault) | Отображается новое изображение превью соответствующего качества | пройден |
| Ввести ссылку на видео без доступного превью (например, видео в обработке) | Отображается сообщение об ошибке загрузки изображения | пройден |

**Заключение**

В рамках данной работы была реализована программная система Sakura Downloader, предназначенная для загрузки видеоконтента с платформы YouTube с возможностью выбора качества и формата. Основной целью проекта являлось создание интуитивно понятного, функционального и визуально привлекательного интерфейса, способного упростить процесс скачивания видео- и аудиофайлов для конечного пользователя. Процесс разработки включал в себя как техническую реализацию программного кода, так и проектирование пользовательского интерфейса с учетом эргономики и удобства взаимодействия.

В процессе разработки использовались современные и надёжные программные средства: библиотека yt-dlp для обработки видеоконтента, модуль tkinter для реализации графического интерфейса, а также вспомогательные библиотеки, такие как Pillow, requests и threading, обеспечивающие стабильность, гибкость и высокую производительность приложения. Программа позволяет не только скачивать видео, но и сохранять эскизы (превью), выбирать разрешение обложки, а также следить за ходом выполнения через встроенный лог.

Вся логика приложения была реализована с учетом требований к читаемости, поддерживаемости и расширяемости кода. Благодаря использованию многопоточности пользовательский интерфейс остаётся отзывчивым даже при длительной загрузке, что повышает общее качество взаимодействия и удовлетворённость пользователей.

Проведённое ручное тестирование подтвердило корректность работы всех основных функций. Программа стабильно функционирует при различных вводных данных и условиях, обеспечивая пользователю положительный опыт использования. Были проанализированы возможные ошибки и исключения, и для большинства из них реализована система оповещений в логах, что упрощает процесс устранения неполадок.

Таким образом, можно сделать вывод, что поставленные задачи были успешно выполнены, а цели проекта — достигнуты. Созданный программный продукт соответствует современным требованиям к настольному прикладному ПО, является готовым к использованию и обладает потенциалом для дальнейшего развития.

Несмотря на достигнутый уровень реализации, программный продукт Sakura Downloader не является финальной версией. В процессе использования и тестирования были выявлены возможные направления усовершенствования, каждое из которых способно значительно повысить удобство, функциональность и универсальность системы.

Во-первых, перспективным направлением является реализация функции загрузки по плейлистам или каналам, что особенно актуально для пользователей, работающих с большим объемом видеоматериала. Это позволит скачивать несколько видео автоматически без необходимости ввода каждой ссылки вручную.

Во-вторых, внедрение системы очередей обеспечит более продвинутую логику управления загрузками. Пользователь сможет формировать список задач и отслеживать их выполнение в реальном времени, что значительно улучшит эффективность при массовой загрузке.

Следующим важным шагом может стать интеграция поддержки других видеохостингов — таких как Vimeo, TikTok, Dailymotion и других. Это позволит расширить круг пользователей программы и сделать её более универсальной.

Кроме того, полезной функцией станет хранение истории загрузок, с возможностью открытия, переименования или удаления файлов прямо из интерфейса приложения. Это сделает программу не только средством загрузки, но и полноценным менеджером мультимедийного контента.

Также возможно добавление системы автоматического обновления, что избавит пользователей от необходимости самостоятельно искать и устанавливать новые версии. Это обеспечит своевременное получение новых функций и улучшений безопасности.

С технической точки зрения, интерес представляет возможность портирования приложения на другие операционные системы (Linux, macOS), а также разработка мобильной версии для Android, что позволит использовать Sakura Downloader в любом месте и в любое время.

В целом, проект обладает высокой степенью гибкости, и за счёт открытости к доработкам может развиваться в различных направлениях в зависимости от требований конечных пользователей и актуальных технологических трендов.

**Список литературы**

1. **yt-dlp Documentation.** Современная библиотека для загрузки медиафайлов с YouTube и других платформ, форк youtube-dl. – GitHub, 2024.https://github.com/yt-dlp/yt-dlp
2. **Работа с графическим интерфейсом в Python: библиотека tkinter. –** Хабр, 2022.https://habr.com/ru/articles/548210/
3. **Как скачать видео с YouTube с помощью Python и yt-dlp. –** Хабр, 2023.https://habr.com/ru/articles/722258/
4. **Универсальный парсер превью YouTube на Python. –** Хабр, 2021.https://habr.com/ru/articles/535708/
5. **Как организовать многопоточность в Python: потоки, очереди, таймеры. –** Хабр, 2020.  
   https://habr.com/ru/articles/501046/
6. **Создание кроссплатформенного GUI в Python с помощью Tkinter. –** Pythonist.ru,2022.  
   https://pythonist.ru/gui-v-python-s-pomoshchyu-tkinter/
7. **Requests – HTTP for Humans.** Документация библиотеки для запросов в интернете на Python. – Python Requests Project, 2024.  
   https://docs.python-requests.org/
8. **Pillow (PIL Fork) – работа с изображениями в Python. –** ReadTheDocs, 2023.https://pillow.readthedocs.io/en/stable/
9. **Visual Studio Code как среда разработки Python-проектов. –** Хабр, 2021.https://habr.com/ru/articles/678560/
10. **Git и GitHub для начинающих: как контролировать версии и публиковать проект.** – GitHub Docs, 2023.https://docs.github.com/ru/get-started/quickstart