

ReShorts Auto - Автоматизация создания вирусных видео

![ReShorts Auto](https://img.shields.io/badge/ReShorts-Auto-06B6D4?style=for-the-badge) ![Python](https://img.shields.io/badge/Python-3.8+-3776AB?style=for-the-badge&logo=python) ![React](https://img.shields.io/badge/React-18-61DAFB?style=for-the-badge&logo=react) ![TypeScript](https://img.shields.io/badge/TypeScript-5.0-3178C6?style=for-the-badge&logo=typescript) **Современная full-stack платформа для поиска, анализа и обработки вирусных видео** [Возможности] (#возможности) • [Установка] (#установка) • [Запуск] (#запуск) • [Документация] (#документация)

Описание

ReShorts Auto — это мощная система автоматизации для создания вирусных коротких видео. Платформа позволяет искать популярные видео на YouTube, Instagram и TikTok, скачивать их без API ключей, анализировать с помощью AI и обрабатывать для создания собственного контента.





Ключевые особенности

- 🔍 **Умный поиск** с расширенными фильтрами (просмотры, лайки, engagement rate, viral score)
 - 👁️ **Предпросмотр результатов** с полной информацией о видео перед скачиванием
 - 📦 **Загрузка без API** используя yt-dlp, instagrapi, TikTok scraper
 - 🤖 **AI анализ видео** через множество бесплатных провайдеров (GPT4Free, OpenRouter, Gemini)
 - 🎨 **Современный UI** на React с темной темой и плавными анимациями
 - 📊 **Дашборд** со статистикой и графиками активности
 - ⚙️ **Гибкая настройка** всех параметров через удобный интерфейс
-




Возможности



Backend (Flask API)

- ✅ **Поиск видео** с расширенными фильтрами
- Платформы: YouTube, Instagram, TikTok






- Фильтры: просмотры, лайки, комментарии, длительность, дата, язык, engagement rate
- Исключение нежелательных ключевых слов
-  **Загрузка видео** с автоматическим fallback
- yt-dlp для YouTube
- instagrapі для Instagram
- TikTok scraper для TikTok
-  **AI анализ** с поддержкой множества провайдеров
- GPT4Free (бесплатный)
- OpenRouter (опционально)
- Google Gemini (бесплатный)
- LocalAI (для локального запуска)
-  **Управление фильтрами**
- Сохранение пользовательских пресетов
- Быстрое применение избранных настроек
-  **Статистика и аналитика**
- Количество скачанных, проанализированных, обработанных видео
- График активности за 7 дней
- Журнал всех операций

Frontend (React SPA)

-  **Dashboard** - главная панель
- Статистические карточки с ключевыми метриками
- График активности за неделю
- Последние операции в реальном времени
- Быстрые действия
-  **Search & Preview** - поиск и предпросмотр
- Панель расширенных фильтров
- Красивые карточки видео с миниатюрами
- Информация: название, описание, автор, статистика
- Viral Score с визуальным индикатором
- Пакетное скачивание выбранных видео
-  **Video Manager** - управление скачанными файлами
- Список всех видео с превью
- Статус обработки и AI анализа
- Действия: удалить, анализировать, обработать

-  **Settings** - настройки системы
- Вкладки: Поиск, AI, Загрузка, Обработка, Автоматизация
- Удобное редактирование всех параметров
- Сохранение в реальном времени
-  **System Status** - статус системы
- Состояние всех загрузчиков
- Доступность AI провайдеров
- Использование диска

Дизайн

-  **Modern Tech Dark** тема
 -  Цветовая схема: neutral-950/900/800 + cyan акценты (#06B6D4)
 -  Плавные анимации 250-300ms
 -  Glow эффекты на интерактивных элементах
 -  Полностью responsive (desktop, tablet, mobile)
-



Установка

Требования

- **Python:** 3.8 или выше
- **Node.js:** 16 или выше (для frontend)
- **npm** или **pnpm**: для управления пакетами

Автоматическая установка

```
# Клонировать репозиторий
git clone https://github.com/abdursidovrustam-alt/ReShorts_auto.git
cd ReShorts_auto

# Запустить скрипт установки
chmod +x setup.sh
./setup.sh
```

Скрипт `setup.sh` автоматически:

- Создаст виртуальное окружение Python
- Установит все Python зависимости
- Установит Playwright браузеры

- Создаст необходимые папки (logs, downloads, processed)
- Проверит наличие Node.js для frontend

Ручная установка

Backend

```
# Создать виртуальное окружение
python3 -m venv venv
source venv/bin/activate # Linux/Mac
# или
venv\Scripts\activate    # Windows

# Установить зависимости
pip install -r requirements.txt

# Установить Playwright браузеры
playwright install

# Создать необходимые папки
mkdir -p logs downloads processed
```

Frontend

```
# Перейти в папку client
cd client

# Установить зависимости (npm или pnpm)
pnpm install
# или
npm install
```



Запуск

Вариант 1: Development режим

Backend:

```
# Активировать виртуальное окружение
source venv/bin/activate

# Запустить Flask сервер
python app.py
```

Backend будет доступен на **http://localhost:5000**

Frontend:

```
# В отдельном терминале
cd client

# Запустить dev сервер
pnpm run dev
# или
npm run dev
```

Frontend будет доступен на **http://localhost:5173**

Вариант 2: Production режим

```
# Собрать frontend
cd client
pnpm run build

# Запустить backend (он будет отдавать статические файлы frontend)
cd ..
python app.py
```

Приложение будет доступно на **http://localhost:5000**

Документация

Структура проекта

```

ReShorts_auto/
├── app.py # Главный Flask сервер
├── config.json # Конфигурация системы
├── requirements.txt # Python зависимости
├── setup.sh # Скрипт автоматической установки
├── .gitignore # Игнорируемые файлы
├──
├── modules/ # Модули загрузчиков
│   ├── universal_downloader.py # Универсальный загрузчик
│   └── downloaders/ # Реализации загрузчиков
│       ├── ytdlp_downloader.py
│       ├── instagrapi_downloader.py
│       └── tiktok_downloader.py
│
├── ai_analyzer/ # AI модули
│   ├── multi_provider.py # Менеджер AI провайдеров
│   └── providers/ # Реализации провайдеров
│       ├── gpt4free_provider.py
│       ├── openrouter_provider.py
│       ├── gemini_provider.py
│       └── localai_provider.py
│
├── data/ # Данные приложения
│   ├── stats.json # Статистика
│   └── filter_presets.json # Сохранённые пресеты
│
├── logs/ # Логи приложения
├── downloads/ # Скачанные видео
├── processed/ # Обработанные видео
├──
├── docs/ # Документация
│   ├── reshorts-auto-design-spec.md # Дизайн-спецификация
│   └── design-tokens.json # Дизайн-токены
│
└── client/ # React приложение
    ├── src/
    │   ├── api/ # API клиент
    │   │   ├── axios.ts # Axios instance
    │   │   └── endpoints.ts # API эндпоинты
    │   ├── components/ # UI компоненты
    │   │   ├── Button.tsx
    │   │   └── Input.tsx

```

```

|   |   └─ Card.tsx
|   └─ pages/           # Страницы
|   |   └─ Dashboard.tsx
|   |   └─ Search.tsx
|   |   └─ VideoManager.tsx
|   |   └─ Settings.tsx
|   |   └─ Status.tsx
|   └─ layouts/         # Layout компоненты
|   |   └─ MainLayout.tsx
|   |   └─ Sidebar.tsx
|   |   └─ Header.tsx
|   └─ types/           # TypeScript типы
|       └─ index.ts
└─ package.json
└─ vite.config.ts
└─ tailwind.config.js

```

API Endpoints

Поиск видео

```

POST /api/search/preview
Content-Type: application/json

{
  "platform": "youtube",
  "query": "вирусные видео",
  "min_views": 10000,
  "max_views": 1000000,
  "min_likes": 500,
  "duration_min": 10,
  "duration_max": 60,
  "date_range": "week",
  "language": "ru",
  "min_engagement": 5.0,
  "exclude_keywords": ["реклама"]
}

```


Статистика

```
GET /api/stats/dashboard
```

Конфигурация

```
GET /api/config/get  
POST /api/config/save
```

Фильтры

```
POST /api/filters/save  
GET /api/filters/list  
DELETE /api/filters/delete/<preset_id>
```

Загрузка

```
POST /api/download  
POST /api/download/batch
```

Анализ

```
POST /api/analyze
```

Файлы

```
GET /api/files/list  
DELETE /api/files/delete/<filename>
```

Полная API документация: [README_FULLSTACK.md](#)

Конфигурация

Все настройки хранятся в файле `config.json`. Вы можете редактировать их через UI (страница Settings) или напрямую в файле.

Основные секции:

- `search` - параметры поиска по умолчанию
- `ai` - настройки AI провайдеров
- `download` - настройки загрузки видео
- `processing` - параметры обработки
- `scenarios` - автоматизация процессов

Дизайн-система

Полная дизайн-спецификация: <docs/reshorts-auto-design-spec.md>

Основные токены:

- **Цвета:** neutral-950/900/800 (фон), cyan-500 (акцент)
 - **Отступы:** 4px / 8px / 16px / 24px / 32px / 48px
 - **Радиусы:** 4px / 8px / 12px / 16px
 - **Анимации:** 250ms (normal), 300ms (smooth)
-



Технологический стек

Backend

- **Flask 3.0** - веб-фреймворк
- **Flask-CORS** - CORS поддержка
- **yt-dlp** - загрузка с YouTube
- **instagrapi** - загрузка с Instagram
- **TikTokApi** - загрузка с TikTok
- **g4f** - бесплатный доступ к GPT
- **playwright** - автоматизация браузера

Frontend

- **React 18** - UI библиотека
 - **TypeScript 5** - типизация
 - **Vite 6** - быстрый сборщик
 - **Tailwind CSS** - utility-first стили
 - **React Router v6** - клиентский роутинг
 - **Axios** - HTTP клиент
 - **Recharts** - графики и визуализация
 - **Lucide React** - набор иконок
-



Решение проблем

Backend не запускается

Проблема: `ModuleNotFoundError: No module named 'flask'`

Решение:

```
source venv/bin/activate
pip install -r requirements.txt
```

Frontend показывает ошибку подключения

Проблема: `Network Error` при запросах к API

Решение: Убедитесь, что backend запущен на порту 5000:

```
python app.py
```

Playwright ошибки

Проблема: `Executable doesn't exist at ...`

Решение:

```
playwright install
```

Нет результатов поиска

Проблема: Поиск не возвращает видео

Решение:

- Проверьте интернет соединение
- Попробуйте другие ключевые слова
- Уменьшите фильтры (`min_views`, `min_engagement`)
- Проверьте логи: `logs/reshorts.log`



Лицензия

MIT License



Автор

MiniMax Agent

Дата создания: 2025-10-17



Благодарности

- **yt-dlp** за мощный загрузчик видео
 - **instagrapi** за Instagram API
 - **g4f** за бесплатный доступ к GPT
 - **Linear, Vercel, GitHub** за вдохновение в дизайне
-

[ Наверх](#reshorts-auto---автоматизация-создания-вирусных-видео)