

SOLAR PhotoSync v1.0

Автоматическая синхронизация медиафайлов из Telegram в файловую систему SOLAR

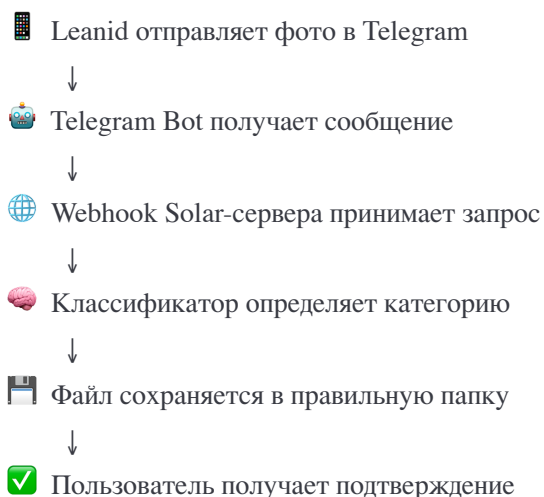
Содержание

1. [Обзор](#)
 2. [Архитектура](#)
 3. [Установка](#)
 4. [Настройка Telegram Bot](#)
 5. [Конфигурация](#)
 6. [Запуск](#)
 7. [Настройка Webhook](#)
 8. [Использование](#)
 9. [API Endpoints](#)
 10. [Структура файлов](#)
 11. [Тестирование](#)
 12. [Troubleshooting](#)
-

Обзор

SOLAR PhotoSync — это модуль для автоматической загрузки медиафайлов (фото, видео, документы) из Telegram в локальную файловую систему с автоматической классификацией по категориям.

Как это работает:



Поддерживаемые типы файлов:

- 📷 Фото (JPG, PNG, HEIC, WebP)
- 🎬 Видео (MP4, MOV, AVI)
- 📄 Документы (PDF, DOC, DOCX, XLS)
- 🎵 Аудио (MP3, WAV, OGG)
- 🔄 GIF-анимации

🏗️ Архитектура

```
SOLAR-PhotoSync/
├── config/
│   └── photosync.config.json  # Конфигурация
├── src/
│   ├── bot.py                # Главный модуль, точка входа
│   ├── webhook_handler.py    # Обработчик Telegram webhook
│   ├── classifier.py         # Классификатор файлов
│   ├── file_saver.py         # Сохранение файлов
│   ├── heic_converter.py     # Конвертация HEIC → JPG
│   └── logger.py             # Логирование
├── tools/
│   └── webhook_setup.py      # Утилита настройки webhook
├── requirements.txt          # Python зависимости
├── start.sh                  # Скрипт запуска
└── SOLAR-Photosync-v1.0.md   # Эта документация
```

Модули:

Модуль	Описание
bot.py	Основной модуль. Web-сервер на aiohttp, маршрутизация
webhook_handler.py	Обработка входящих update от Telegram, скачивание файлов
classifier.py	Классификация по ключевым словам в подписи/чате/имени
file_saver.py	Создание структуры папок, сохранение с правильным именем
heic_converter.py	Автоконвертация HEIC/HEIF в JPG с сохранением EXIF
logger.py	Централизованное логирование с ротацией

🔧 Установка

Требования

- Python 3.9+

- macOS / Linux / Windows
- Telegram Bot Token

Быстрая установка (Mac/Linux)

```
bash

# Клонировем/копируем проект
cd SOLAR-PhotoSync

# Делаем скрипт исполняемым
chmod +x start.sh

# Запускаем (автоматически создаст venv и установит зависимости)
./start.sh --setup-only
```

Ручная установка

```
bash

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

# Устанавливаем зависимости
pip install -r requirements.txt
```

Установка поддержки HEIC

macOS:

```
bash

# sips встроен, но для лучшей поддержки:
brew install imagemagick
```

Linux (Ubuntu/Debian):

```
bash

sudo apt-get install imagemagick libheif-examples
```

Через Python (все платформы):

```
bash

pip install pillow-heif
```

Настройка Telegram Bot

1. Создание бота

1. Откройте Telegram и найдите @BotFather
2. Отправьте команду `/newbot`
3. Введите имя бота: `Solar PhotoSync`
4. Введите username: `solar_photosync_bot` (должен быть уникальным)
5. Скопируйте полученный **Bot Token**

2. Настройка токена

Отредактируйте `config/photosync.config.json`:

```
json

{
  "bot": {
    "token": "YOUR_BOT_TOKEN_HERE",
    "webhook_url": "https://your-server.com/api/photosync/webhook"
  }
}
```

Или через переменную окружения:

```
bash

export TELEGRAM_BOT_TOKEN=your_token_here
```

Конфигурация

Файл: `config/photosync.config.json`

```
json
```

```

{
  "bot": {
    "token": "YOUR_TELEGRAM_BOT_TOKEN",
    "webhook_url": "https://your-server.com/api/photosync/webhook",
    "webhook_secret": "optional-secret-for-security"
  },

  "storage": {
    "root_path": "/SOLAR/PhotoSync",
    "allowed_types": ["photo", "document", "video", "animation"],
    "allowed_extensions": [".jpg", ".jpeg", ".png", ".heic", ".pdf", ".mp4"]
  },

  "processing": {
    "convert_heic": true,
    "heic_quality": 85,
    "preserve_exif": true,
    "max_file_size_mb": 100
  },

  "classification": {
    "auto_classification": true,
    "default_category": "Other",
    "categories": {
      "Sprinter": ["sprinter", "спринтер", "sprint"],
      "LDZ": ["ldz", "vagon", "вагон", "wagon", "railway"],
      "Legal": ["court", "суд", "teismas", "legal", "иск"],
      "Documents": ["document", "паспорт", "passport", "справка"]
    }
  },

  "logging": {
    "enabled": true,
    "log_path": "/SOLAR/PhotoSync/logs",
    "log_file": "photosync.log",
    "log_level": "INFO"
  },

  "server": {
    "host": "0.0.0.0",
    "port": 8080
  }
}

```

Параметры классификации

Добавляйте свои категории и ключевые слова:

```
json
```

```
"categories": {  
  "MyProject": ["myproject", "проект", "project-x"],  
  "Finance": ["invoice", "счёт", "payment", "оплата"]  
}
```



Запуск

Быстрый запуск

```
bash
```

```
./start.sh
```

Запуск с параметрами

```
bash
```

```
# На другом порту
```

```
./start.sh --port 3000
```

```
# С другим конфигом
```

```
./start.sh --config /path/to/config.json
```

Ручной запуск

```
bash
```

```
source venv/bin/activate
```

```
python src/bot.py
```

Запуск как сервис (systemd)

Создайте файл `/etc/systemd/system/photosync.service`:

```
ini
```

[Unit]

Description=SOLAR PhotoSync Bot

After=network.target

[Service]

Type=simple

User=your_user

WorkingDirectory=/path/to/SOLAR-PhotoSync

ExecStart=/path/to/SOLAR-PhotoSync/venv/bin/python src/bot.py

Restart=always

RestartSec=10

[Install]

WantedBy=multi-user.target

bash

sudo systemctl enable photosync

sudo systemctl start photosync

Настройка Webhook

Требования для Webhook

- HTTPS URL (Telegram требует SSL)
- Публичный IP или домен
- Открытый порт (443, 80, 88, или 8443)

Способы получить HTTPS URL

1. Ngrok (для тестирования):

bash

ngrok http 8080

Скопируйте https URL

2. Cloudflare Tunnel (бесплатно):

bash

cloudflared tunnel --url http://localhost:8080

3. Свой сервер с SSL:

- Настройте nginx/caddy с Let's Encrypt

- Проксируйте на localhost:8080

Установка Webhook

bash

Через утилиту

```
python tools/webhook_setup.py set --url https://your-domain.com/api/photosync/webhook
```

Проверка

```
python tools/webhook_setup.py info
```

Тест бота

```
python tools/webhook_setup.py test
```

Проверка через curl

bash

Установить webhook

```
curl -X POST "https://api.telegram.org/bot<TOKEN>/setWebhook" \
  -H "Content-Type: application/json" \
  -d '{"url": "https://your-server.com/api/photosync/webhook"}'
```

Проверить статус

```
curl "https://api.telegram.org/bot<TOKEN>/getWebhookInfo"
```



Использование

Отправка файлов

1. Найдите вашего бота в Telegram
2. Отправьте фото/видео/документ
3. Бот ответит: ☒ Принято → [Категория]

Указание категории

Через команду в подписи:

/sprinter

[прикрепить фото]

Через ключевые слова в подписи:

Фото вагона LDZ

[прикрепить фото]

Автоматическая классификация

Бот анализирует (в порядке приоритета):

- 1. Команду в тексте (/sprinter)
- 2. Ключевые слова в подписи
- 3. Название чата
- 4. Имя файла

Если категория не определена → файл в папку Other

API Endpoints

Endpoint	Метод	Описание
/api/photosync/webhook	POST	Telegram webhook
/api/photosync/health	GET	Health check
/api/photosync/stats	GET	Статистика хранилища
/	GET	Информационная страница

Примеры ответов

Health check:

```
json
{
  "status": "healthy",
  "version": "1.0.0",
  "service": "Solar PhotoSync"
}
```

Stats:

```
json
```

```
{
  "root_path": "/SOLAR/PhotoSync",
  "total_files": 142,
  "total_size_mb": 856.32,
  "categories": {
    "Sprinter": 45,
    "LDZ": 38,
    "Documents": 29,
    "Other": 30
  },
  "dates": ["2025-11-29", "2025-11-28", "2025-11-27"]
}
```

Структура файлов

```
/SOLAR/PhotoSync/
├── 2025-11-29/
│   ├── Sprinter/
│   │   ├── 20251129_143022_IMG_1234.jpg
│   │   └── 20251129_144510_photo.jpg
│   ├── LDZ/
│   │   └── 20251129_150000_vagon_doc.pdf
│   ├── Legal/
│   ├── Documents/
│   └── Other/
├── 2025-11-28/
│   └── ...
└── logs/
    └── photosync.log
```

Формат имени файла

```
YYYYMMDD_HHMMSS_originalname.ext
```

Пример: `20251129_143022_IMG_1234.jpg`

Тестирование

Тест подключения бота

```
bash

python tools/webhook_setup.py test
```

Локальный тест webhook

```
bash
```

```
# Запустите бот
```

```
python src/bot.py
```

```
# В другом терминале отправьте тестовый запрос
```

```
curl -X POST http://localhost:8080/api/photosync/webhook \  
-H "Content-Type: application/json" \  
-d '{  
  "update_id": 123,  
  "message": {  
    "message_id": 1,  
    "chat": {"id": 12345, "title": "Test Sprinter Chat"},  
    "photo": [{"file_id": "test", "file_size": 1000}]  
  }  
'
```

Проверка логов

```
bash
```

```
tail -f /SOLAR/PhotoSync/logs/photosync.log
```

! Troubleshooting

Бот не отвечает

1. Проверьте токен: `python tools/webhook_setup.py test`
2. Проверьте webhook: `python tools/webhook_setup.py info`
3. Проверьте логи: `tail -f /SOLAR/PhotoSync/logs/photosync.log`

Ошибка SSL/Webhook

```
error: "wrong response from webhook: 502"
```

Решение:

- Убедитесь что URL начинается с `https://`
- Проверьте что сервер доступен извне
- Проверьте SSL сертификат

HEIC не конвертируется

```
warning: "No HEIC converter found"
```

Решение:

```
bash

# macOS
brew install imagemagick

# Linux
sudo apt-get install imagemagick libheif-examples

# Python
pip install pillow-heif
```

Файлы не сохраняются

1. Проверьте права на папку `/SOLAR/PhotoSync`
2. Проверьте свободное место на диске
3. Проверьте путь в конфиге `storage.root_path`







Команда


- **Dashka** — Senior, координация, ревью
- **Claude** — Инженер, разработка
- **Leanid** — Архитектор, тестирование, приёмка



Changelog

v1.0.0 (2025-11-29)

-  Начальный релиз
-  Telegram webhook интеграция
-  Автоклассификация по ключевым словам
-  HEIC → JPG конвертация
-  Структурированное хранение файлов
-  Логирование с ротацией

 **SOLAR PhotoSync v1.0** — Храни всё на своих местах!