Farysene AI Telegram Bot

Development & writer is my name
Telegram Bot: `@Faryseneaidownloder_bot`

Feature-rich Telegram bot for downloading and processing media from multiple platforms, delivering daily music, keeping per-user history/statistics, and built with a modular handler architecture for easy extension.

This README documents the **current internal behavior** of the bot and serves as both **user** and **developer** documentation. AI-specific features and future filters are documented separately in `AI_UPDATE_IDEAS.md`.

Project name

- **Repository / Package name:** `@Faryseneaidownloder`
- **Display name:** Farysene AI Telegram Bot

Overview & purpose

Farysene AI Telegram Bot is a Python-based Telegram bot created to experiment with and develop modern bot features, with a long-term focus on **AI-powered capabilities**.

Current functionality includes:

- Multi-platform media handling
- Daily music delivery
- Per-user history and statistics
- A modular architecture that allows new platforms and AI features to be added with minimal effort

Supported platforms include **Spotify, Pinterest, Instagram, TikTok, YouTube, Twitter (X), and Threads**.

Features (current)

Multi-platform link handling (beta)

- Supported platforms:
  - Spotify
  - Pinterest
  - Instagram
  - TikTok
  - YouTube
  - Twitter (X)
  - Threads
- Automatic link detection and routing to the correct handler
- Multi-link handling:
  - Spotify: 2–3 links per message
  - Pinterest: up to 5 links per message

Daily songs delivery (opt-in)

- Sends **3 curated songs per day**:
  - Morning
  - Noon
  - Evening
- Enabled via `/daily_on`
- Delivered messages include an unsubscribe option

History & statistics

- `/history` — shows recent downloads per user
- `/stats` — per-user download counts, platform breakdown, and last 7 days overview

Task control & job management

- `/status` — platform availability check
- `/retry` — best-effort retry of the last operation
- `/cancel` — attempt to cancel the current task

Access control (development)

- `ALLOWED_USERS` list restricts usage before public deployment

Caching & persistence

- SQLite database (default)
- Alembic scaffold included for future schema migrations

Logging

- File logging (`bot.log`)
- Console logging

Architecture & layout

- **Bot framework:** aiogram v3 (long polling, no webhook)
- **Entrypoint:** `bot.py`
- **Handlers:** `handlers/`
    - One module per platform
    - `handlers/detector.py` handles link detection and routing
- **Task manager:** `task_manager.py` (retry, backoff, rate control)
- **Database utilities:** `db_manager.py`
- **Utilities:** `utils.py` (URL parsing, validation, helpers)
- **Scheduler:** APScheduler (Asia/Tehran timezone)
- **Migrations:** `alembic/`

Suggested repository layout

```
.
├─ bot.py
├─ requirements.txt
├─ .env.example
├─ handlers/
│   ├─ detector.py
│   ├─ spotify.py
│   ├─ pinterest.py
│   └─ ...
├─ task_manager.py
├─ db_manager.py
├─ utils.py
```

```
├── daily_songs/
├── migrations/
├── bot.log
└── README.md
```

Requirements & prerequisites

- **Python:** 3.10+ (3.11 recommended)
- **FFmpeg:** Available on PATH or configured via `FFMPEG_PATH`
- **Internet connectivity** (optional proxy support)
- **SQLite** (default database)
  - Optional GUI: SQLite Expert Professional
- **OS:** Windows, Linux, or macOS

> Recommended: use a Python virtual environment.

Installation

1. Clone repository

```
git clone <your-repo-url> @Faryseneaidownloder
cd @Faryseneaidownloder
```

2. Create & activate virtual environment

Windows (PowerShell)

```
py -3 -m venv .venv
.venv\Scripts\Activate.ps1
```

Linux / macOS

```
python3 -m venv .venv
source .venv/bin/activate
```

3. Install dependencies

```
pip install -r requirements.txt
```

Configuration (env)

Create a `.env` file in the project root:

```
BOT_TOKEN=your_telegram_bot_token
PROXY=
FFMPEG_PATH=ffmpeg

SPOTIPY_CLIENT_ID=
SPOTIPY_CLIENT_SECRET=
GENIUS_API_TOKEN=
SPOTIFY_MARKET=US

ALLOWED_USERS=123456789,987654321
```

Notes:

- Never commit `.env` to version control
- Remove or update `ALLOWED_USERS` before public deployment

Run

```
python bot.py
```

The bot uses **long polling**. Ensure the bot token is valid and no conflicting webhook is set.

Commands (user-facing)

- `/start` — welcome message
- `/help` — usage and limits
- `/playlist` — placeholder (under development)
- `/history` — recent downloads
- `/stats` — download statistics

- `/status` — platform status
- `/retry` — retry last task
- `/cancel` — cancel active task
- `/daily_on` — enable daily songs

Daily songs — file naming & delivery

Place MP3 files in `daily_songs/` using the following naming format:

daily_morning-<Title> - <Artist>.mp3
daily_noon-<Title> - <Artist>.mp3
daily_evening-<Title> - <Artist>.mp3

Songs are delivered automatically based on Asia/Tehran timezone.

Database & persistence

- SQLite database stores:
  - Download history
  - User preferences
  - Statistics
- Alembic is used for schema migrations
- Media and cache files are ignored by git

Logging

- File: `bot.log`
- Console output for runtime diagnostics

Security & privacy notes

- Never commit tokens, cookies, or session data
- Restrict access with `ALLOWED_USERS` during development

- Respect platform Terms of Service and copyright laws

Troubleshooting

Bot does not start

- Check `BOT_TOKEN`
- Verify Python version (3.10+)
- Ensure dependencies are installed

No media processing

- Verify FFmpeg installation and path

Daily songs not delivered

- Check file naming in `daily_songs/`
- Ensure users enabled daily songs

Platform errors / rate limits

- Retry later
- Consider proxy usage

See `bot.log` for detailed errors.

Development notes (how to extend)

Add a new platform

1. Create a new handler in `handlers/`
2. Update `handlers/detector.py`
3. Add config options to `.env.example` and README

Database changes

- Add Alembic migration in `alembic/versions/`
- Run `alembic upgrade head`

Scheduler

- APScheduler uses Asia/Tehran timezone

Roadmap

- Unified media processing module
- AI features:
    - Transcription & subtitles
    - Translation & summaries
    - Personalized recommendations
- Advanced rate limiting
- Cookie/session management
- Admin/status dashboard
- Tests & CI pipeline

Contributing & license

Contributions are welcome via issues and pull requests.

License: Public domain

Contact

Development & Writing is my name
Telegram