🛡️ NeuroChain: Autonomous PR Defense System

NeuroChain is an AI-powered, real-time PR defense system designed to protect brands from viral misinformation. It acts as an automated "Iron Dome," continuously scanning the web for threats, analyzing them using LLMs, and alerting PR teams with verified intelligence.

🌟 Key Features

- **Multi-Source Surveillance:** Automatically scans **NewsAPI**, **Reddit**, **YouTube**, and **Google Search** for brand mentions.
- **AI-Powered Analysis:** Uses **OpenAI GPT-4o/3.5** to extract claims, fact-check against evidence, and determine a verdict (True/False/Misleading).
- **Smart Prioritization:** Scores every post based on engagement and risk keywords. Only high-priority threats trigger alerts.
- **Autonomous Watchdog:** A background bot runs 24/7, analyzing high-risk threats instantly and emailing full reports without human intervention.
- **Immutable Ledger:** Every analysis is hashed and chained, creating a tamper-evident record of all debunked claims.
- **Modern Dashboard:** A sleek Streamlit interface for manual review, company management, and history tracking.

🚀 Getting Started

1. Prerequisites

- Python 3.8+
- API Keys for:
  - **OpenAI** (Required for analysis)
  - **NewsAPI**, **Reddit**, **YouTube**, **SerpAPI** (Optional, for real data)
  - **Gmail App Password** (For email alerts)

2. Installation

Clone the repository and install dependencies:

```
git clone [https://github.com/yourusername/neurochain.git](https://github.com/yourusern
cd neurochain
pip install -r requirements.txt
```

3. Configuration

1. Rename `.env.example` to `.env` (or create a new `.env` file).
2. Add your API keys. **Note:** For Gmail, use an App Password, not your login password.

```
# .env file
OPENAI_API_KEY="sk-proj-..."
OPENAI_ORG_ID="org-..."
OPENAI_PROJECT_ID="proj-..."

# Data Sources
NEWS_API_KEY="your_key"
REDDIT_CLIENT_ID="your_id"
REDDIT_CLIENT_SECRET="your_secret"
YOUTUBE_API_KEY="your_key"
SERPAPI_KEY="your_key"

# Email Alerts
EMAIL_SENDER="your_email@gmail.com"
EMAIL_PASSWORD="your_16_char_app_password"
```

4. Running the System

You can run the entire system (Dashboard + Background Bot) with a single command:

```
python start_all.py
```

- **Dashboard:** Opens in your browser at `http://localhost:8501` .
- **Scheduler Bot:** Runs in the terminal, scanning for threats in the background.

📖 Usage Guide

1. Manage Targets

- Go to the **"Manage Companies"** page in the dashboard.
- Add the **Company Name** (e.g., "Tesla") and the **Alert Email** (e.g., "pr@tesla.com").
- The bot will now track this company.

2. Monitoring (Manual & Auto)

- **Dashboard:** View the **"Alerts Dashboard"** to see a live feed of all mentions (Low, Medium, High).
- **Manual Scan:** Click "Force Network Scan" to trigger an immediate update.
- **Auto-Pilot:** The `scheduler_service.py` runs every 6 hours for a full scan.
- **Watchdog:** The bot checks for **High Risk** threats every 10 seconds. If a high-priority threat is found, it analyzes it and **emails you immediately**.

3. Review & Report

- Go to **"History & Ledger"** to see all past analyses.
- Expand an item to see the extracted claim, verdict, and AI-generated PR response.
- Click **"📧 Send Report"** to manually forward the analysis to the client.

🛠️ Architecture

- **Frontend:** Streamlit (`app.py`)
- **Backend:** Python Scheduler (`scheduler_service.py`)
- **Database:** SQLite (`neurochain.db`)
- **AI Engine:** OpenAI API (`core/analyzer.py`)
- **Ledger:** SHA-256 Hashing (`core/ledger.py`)

🛡️ Security Note

- API Keys are loaded securely from `.env` .
- The system uses a local database by default. For production deployment, switch `db.py` to connect to PostgreSQL.

License

MIT License