# Kenyan News Monitoring Web Crawler for Social Harm Detection

Author & Developer: Mr. Patrick

Date: November 2025

## 1. Overview

- Scraping: Built a robust, multilingual (English + Swahili + Kikuyu-aware) news scraper targeting 15 major Kenyan news sites

- ML/NLP: Integrated Named Entity Recognition (NER) and multilingual sentiment analysis with keyword-based social-harm categorization (GBV, Cyberbullying, Scams)

- Backend: Designed and implemented a Supabase (PostgreSQL) backend with real-time dashboard auto-refresh via triggers

- Deployment: Created a fully automated nightly GitHub Actions pipeline running at 00:00 EAT every day

## 2. Objectives

The main goal of this project is to automatically monitor Kenyan online media every single day and detect articles related to three critical social harm areas:

1. Gender-Based Violence (GBV) & Femicide

2. Cyberbullying & Online Harassment

3. Scams & Financial Fraud

Why this matters:

- These issues are severely under-reported in structured data in Kenya

- Early detection from news can inform NGOs, researchers, law enforcement, and policymakers

- The system runs completely autonomously

## 3. Scraping / Data Collection

### Libraries Used

selenium + webdriver-manager → for JavaScript-heavy sites

newspaper3k → best-in-class article extraction (title, text, date, authors)

beautifulsoup4 + lxml → link extraction from homepage

pandas → data structuring

nltk → sentence tokenization for keyword matching

transformers + torch → Hugging Face NER & sentiment models

supabase-py → direct database uploads

### Scraping Workflow

1. Headless Chrome launches (stealth mode to avoid detection)

2. Visits each of the 15 target sites (Tuko, Citizen, Nation, The Star, etc. + Swahili & Kikuyu sections)

3. Waits for page to fully load (60-second timeout)

4. Extracts up to 10 latest article links per site using strict path filters (`/news/`, `/article/`, `/kenya/`, `/swahili/`, etc.)

5. For each link → `newspaper3k.Article()` downloads and parses clean text, title, and publish date

6. Filters:

   - Only articles < 30 days old

   - Minimum 150 characters of body text

7. Runs categorization + NLP analysis

8. Stores everything in a Pandas DataFrame → CSV + Supabase

### Challenges & Solutions

| Challenge | Solution Implemented |
|---|---|
| Heavy JavaScript sites | Selenium with headless Chrome + long waits |
| Bot detection | Disabled automation flags, excluded switches, spoofed navigator.webdriver |
| Duplicate articles | `article_url` is UNIQUE in DB + upsert on conflict |
| robots.txt compliance | Respectful delays (2–10 sec), only public news sections, no aggressive crawling |
| Dynamic "infinite scroll" | Limited to top 10 visible links (most sites show latest on homepage) |

## 4. Data Preprocessing / ML Pipeline

Categorization (Rule-Based + Multilingual Keywords)

I chose a hybrid keyword approach instead of full classifier training because:

- Very limited labeled training data in Swahili/Kikuyu

- Need for immediate interpretability and low latency

### Machine Learning Components

| Task | Model Used | Reason Chosen |
|---|---|---|
| Named Entity Recognition \| ` | `Davlan/bert-base-multilingual-cased-ner-hrl` | Excellent performance on African names/locations |
| Sentiment Analysis | `cardiffnlp/twitter-xlm-roberta-base-sentiment` | Trained on multilingual tweets, understands code-switching |

Both models run locally (cached via GitHub Actions) – no external API costs.

## 5. Backend & Storage

**Database**: Supabase (PostgreSQL as a service) – chosen for:

- Free tier sufficient

- Built-in REST API & Auth

- Real-time subscriptions

- Easy SQL editor & triggers

**Main Tables**

1. `scraped_articles` – raw article data (one row per article)

2. `dashboard` – one row per day with aggregated statistics (auto-refreshed)

**Smart Dashboard System**

I built a self-updating dashboard using PostgreSQL triggers:

- Every time an article is inserted/updated/deleted → trigger fires

- `refresh_dashboard_for_date(CURRENT_DATE)` recalculates ALL metrics for today

- Uses `ON CONFLICT (date) DO UPDATE` → always exactly one row per day

This means your dashboard is always 100% up-to-date without running separate jobs.

## 6. Deployment / Automation

Fully automated nightly run via GitHub Actions (`scraper.yml`)

- Schedule: Every day at 00:00 East Africa Time (21:00 UTC)

- Manual trigger available (`workflow_dispatch`)

- Caches: pip, Chrome, Hugging Face models → cold start < 2 min, warm < 40 sec

- Secrets: `SUPABASE_URL` and `SUPABASE_KEY` stored securely

- Artifact: Daily CSV downloadable from GitHub Actions UI

## 7. References / Links

- GitHub Repository: https://github.com/Muchai10/safeguard_crawler

- Main Script: `WebCrawler_V2.py`

- Database Schema & Dashboard Logic: `Supabase.sql`

- CI/CD Pipeline: `.github/workflows/scraper.yml`

- Requirements: `requirements.txt`

- Daily CSV Artifacts: Available in GitHub Actions → "scraped-articles.csv"