WeeklyAI App Summary (Repo-Based, One Page)

What It Is

WeeklyAI is a global AI product discovery app that combines crawler pipelines and LLM-based scoring to surface rising stars (2-3) and dark horses (4-5).
It serves a web experience and REST API backed by curated JSON datasets in crawler/data.

Who It's For

Primary persona: Product Managers (PMs) who want fast signal on emerging AI products across regions and categories (software + hardware).

What It Does

- Runs multi-region discovery with weighted coverage and rotating keyword pools.
- Routes providers by region (China -> GLM, others -> Perplexity) for search and extraction.
- Applies quality gates (required fields, URL checks, why-matters quality) and deduplication.
- Scores and tiers products into dark horses vs rising stars, then syncs to products_featured.json.
- Collects social first-hand signals from YouTube/X and enriches featured products with latest news.
- Exposes product APIs (dark horses, rising stars, search, blogs, industry leaders, related).
- Renders a three-part homepage flow: weekly dark horses, swipe discovery, and more recommendations.

How It Works (Architecture)

- Ingestion: crawler/main.py orchestrates spiders (Product Hunt, Hacker News, Tech news, hardware, YouTube, X, etc.).
- Discovery/Scoring: crawler/tools/auto_discover.py searches, extracts, validates, dedupes, scores, and saves products.
- Storage: crawler/data/*.json (products_featured.json, blogs_news.json, dark_horses/, rising_stars/, industry_leaders.json).
- Serving: backend Flask app (backend/app/__init__.py) exposes /api/v1 endpoints via ProductService/ProductRepository.
- Frontend: Express + EJS serves UI; frontend/public/js/main.js fetches backend APIs and renders sections/cards.
- Authentication/user accounts and formal RBAC: Not found in repo.

How To Run (Minimal)

- 1. Install deps: backend pip requirements, crawler pip requirements, and frontend npm install.
- 2. Create .env from env.example and set PERPLEXITY_API_KEY (plus ZHIPU_API_KEY if using China GLM routing).
- 3. Start backend API: cd backend && python run.py (http://localhost:5000).
- 4. Start frontend UI: cd frontend && npm start (http://localhost:3000).
- 5. Optional data refresh: cd crawler && python3 tools/auto_discover.py --region all.

Config note: crawler/.env.example is referenced in README but Not found in repo.

Source basis: README.md, crawler/, backend/, frontend/, ops/ files in this repository only.