clients.teknolepassword: 316h7y$!x-71ck13-516n41!

PSAI_1

“ Build You are an expert automation engineer standing up “Culture Current — Lite” within a single evening, operating as 1.2 FTE. Build everything inside a free IDE/ notebook using only zero-cost services (Google Sheets, GitHub Actions, Render free cron, Ollama local models, lightweight storage in JSON/CSV). Target output: weekly auto-brief with citations and analyst approval loop.”

Scope & Constraints:

Data: choose one source family (RSS, Reddit via Pushshift-like mirrors, or YouTube Trending via rapidapi-free alternatives).

Processing: orchestrate with makefiles + GitHub Actions (scheduled weekly + manual dispatch).

Models: run Ollama (llama3.1 or mistral) locally via container; prompts reference shared JSON contract.

Storage: cache harvested items in Google Sheets or flat files; maintain evidence snippets.

Deliverables: Markdown + DOCX brief saved to Drive/SharePoint equivalent (use rclone + free tier).

Governance: enforce citation checks, dedupe by URL hash, log reviewer decisions.

Timeline: <5 days to MVP, <60 minutes weekly runtime.

Extensions: Document upgrade hooks for ChatGPT Pro / MS Copilot Pro (swap Ollama summarizer + use Pro for manual override).

Deliverables:

1. Repo layout with scripts (`harvest.py`, `extract.py`, `report.py`, `approve.ipynb`), workflow YAML, config templates.

2. Prompt pack (harvest/extract/report) tuned for Ollama models, plus Pro-tier variants.

3. README quickstart for non-technical analyst, including verification checklist and rollback.

clients.teknolepassword: 316h7y$!x-71ck13-516n41!

PSAI_2

“ Act as lead architect using Dust.tt to launch “Culture Current — Lean Alpha” in under 6 weeks, with only 1.2 FTE total effort. Build a semi-autonomous, no-cost-to-run pipeline that polls daily across News, Reddit, YouTube, Spotify, normalizes into a light knowledge graph, and pushes alerts.”

Key Requirements:

Dust workspace: define agents (Harvest-News, Harvest-UGC, Normalizer, KG-Builder, PS-Scorer, Reporter, QA).

Schedules: Dust recurring jobs (daily scan, weekly brief) + Render free cron pings for backups.

Connectors: RSS (custom fetch), Reddit via free search endpoints, YouTube/ Spotify public APIs, Upstash Redis free tier for dedupe cache.

KG Storage: lightweight graph via Memgraph community Docker or sqlite edge table; expose via Dust data source.

Predictions: simple trend momentum scoring + PS adjacency rules (D0–D2).

Monitoring: Dust evals for citation presence, Slack-compatible webhooks via n8n.cloud free.

Human-in-loop: Dust task queue for analyst review <30 min/week.

Output: automated briefs (Markdown, DOCX) + JSON feed served on Vercel free tier.

Prepare upgrade notes: integrate ChatGPT Pro / Copilot Pro as premium summarizer or fallback agent.

Deliverables:

1. Dust flow definitions (YAML/JSON) ready to import, with agent prompts and tool wiring.

2. Setup guide covering zero-cost deployment (Dust + Render + Vercel + Upstash + local Ollama for heavy LLM runs).

3. Runbook for daily alerting, evaluation dashboards, and path to expand KG or swap in paid LLMs.

clients.teknolepassword: 316h7y$!x-71ck13-516n41!

PSAI_3

Design a fully autonomous “Culture Current — Temporal KG” solution built in short order by a multi-agent overseer, with zero-cost infrastructure (Dust.tt orchestrator, Vercel, Render free tier, Neo4j Aura free, Ollama clusters). You are the 1.2 FTE responsible for setup and ongoing operations.

Architecture Goals:

Overlord Agent manages dynamic prompts, spawns specialist agents (per-source harvesters, canonicalizer, graph-writer, temporal modeler, PS adjacency, reporter, bias auditor).

Data sources: 6–8 free/public feeds (News, Reddit, YouTube, Spotify, Google Trends via SERP API-free alternative, Letterboxd RSS, fashion blogs).

Temporal KG: store in Neo4j Aura free tier with weekly buckets, provenance ledger, and API published via Vercel Edge functions.

Forecasting: temporal embeddings using free notebooks + Ollama time-series models; predictions surfaced in watchlist cards.

Autonomy: agents schedule themselves (Dust orchestrations + Render worker) while respecting cost guardrails; include evaluation suite (precision@k, forecast lift) and RBAC-style approval.

Client portal: lightweight Next.js app on Vercel pulling from Builder control plane (Dust API) + Neo4j.

Deliverable timeline: phased release within 12 weeks, each increment shippable; include rollback plans and hotfix playbooks.

Upgrade path: optional ChatGPT Pro / Copilot Pro for higher-quality reporting or complex audits.

Deliverables:

1. Comprehensive prompt kit (Overlord + specialists) with dynamic context injection, tool schemas, and self-healing instructions.

2. Infrastructure-as-code snippets (Docker compose for local Ollama cluster, Render deployment configs, Vercel edge handler templates).

3. Operational handbook covering monitoring, alerting, evaluation, and forecast verification.

clients.teknolepassword: 316h7y$!x-71ck13-516n41!

PSAI_4

Evaluate how Dust.tt can act as a single control plane when paired with a managed graph database (Neo4j Aura Free or AWS Neptune sandbox). Map out integration patterns for Culture Current across all tiers (Small/Medium/Large), detailing:

Dust data sources vs. graph connectors (read/write patterns, caching strategies).

Agent orchestration for harvesting, KG updates, semantic search, and forecasting.

Migration story: start with Dust internal tables → introduce vector store → attach external graph without rework.

Governance: versioned prompts, eval pipelines, human approvals, provenance storage in graph.

Cost management: stay within free tiers (Dust basic, graph free tier, Render/Vercel free plans, Ollama self-hosted).

Expansion hooks: when and how to layer ChatGPT Pro / Copilot Pro for selective workloads (reporting, anomaly review).

Produce a blueprint summarizing flows, agents, storage bindings, and phased roadmap from prototype to temporal KG with forecasting.