# GovSight Master Notes vNext

**Design & Build Plan for Mike Lane’s AI Lobbying Assistant**  
*Version: July 23, 2025 – Draft prepared by Nyx*

## Document Purpose

You asked for a **single, comprehensive design + working notes document** that you can re‑upload in future sessions to re‑prime me instantly and resume build work without re‑explaining context. This file is that artifact. It consolidates:

* Core mission & strategic differentiators.
* System concept & architecture (local + persistent + data‑rich AI copilot for lobbying).
* Agreed design decisions (data domains, hybrid scope, freshness tiers, authority hierarchy, etc.).
* Environment & cutover strategy (dev → staging → prod).
* Data model outlines (canonical entities, facts, events, watchlists, programs, etc.).
* Retrieval & reasoning flow (DB → vector → web cascade).
* Memory lifecycle, correction handling, and provenance rules.
* Reporting templates (Weekly Client Brief; Meeting Kit).
* Refactor roadmap to break talk.py monolith into modular package.
* Initial bootstrap data plan (authoritative + curated “Mike Brain” overlay).
* Implementation milestones & recommended build sequence.
* Usage examples & workflows.
* Open questions & future expansion hooks.

Keep this document updated as we iterate; think of it as the **GovSight Operating Manual + Developer Spec.**

# 1. North Star & Strategic Context

## 1.1 Who You Are (Operating Persona)

* **Mike Lane** – Federal lobbyist; currently a one‑man shop operating nationally (municipalities, special districts, water/environmental, public safety, infrastructure, and emerging contaminants policy lanes).
* Competes against larger firms by being **faster**, **better informed**, and **able to synthesize massive, messy public sector data** into actionable briefings for local decision‑makers.
* Works across multiple time zones; core business timezone **America/New\_York**.
* Needs repeatable, documented, and auditable outputs (boards, councils, congressional staff).

## 1.2 Core Problem

each client interaction today requires: re‑priming background, digging up current bill status, checking deadlines, cross‑referencing eligibility, and re‑synthesizing talking points. That overhead does not scale.

## 1.3 North Star Statement

**GovSight is a persistent, domain‑tuned AI research teammate that remembers everything relevant to your lobbying practice, continuously ingests federal legislative and funding data, tracks deadlines, and generates client‑ready deliverables on demand—so one person can operate with the reach of a full research team.**

## 1.4 Two Main End Goals

1. **Always‑On Lobbying Intelligence Brain** – Up‑to‑date knowledge of Congress, agencies, programs, and client‑specific priorities; contextual memory across sessions.
2. **Seamless Report & Brief Generation** – Push‑button production of weekly updates, meeting kits, funding eligibility summaries, and board packets without re‑building context.

# 2. Guiding Design Principles

1. **Deterministic First, Generative Second.** Query structured data (DB) and authoritative feeds before invoking large language models for synthesis.
2. **Hybrid Global + Client Overlay.** Shared federal knowledge base; private client layers (strategy, internal docs, sensitive politics).
3. **Geo & Jurisdiction Keys Everywhere.** Every fact and document is tied to normalized location metadata to prevent cross‑jurisdiction contamination (e.g., Grandview, TX ≠ Grandview, WA).
4. **Freshness & Authority Metadata.** Track source date, last verified timestamp, and source authority tier; use them in answer selection.
5. **User Corrections = High‑Signal Labels.** Immediately capture corrections; mark conflicts; verify in background.
6. **Curation Gate.** Not all conversational text becomes permanent fact. Facts enter a pending/curated pipeline unless auto‑trusted (Tier‑1 authoritative sources).
7. **Transparent Confidence & Provenance.** Every answer is internally scored; optional user‑visible indicators (“Verified as of July 23, 2025 from Congress.gov”).
8. **Modular Codebase.** Break the monolithic runtime into testable components (intent, constraints, retrieval, memory, reports, ingestion).
9. **Environment Isolation + Reset Paths.** Develop dirty; stage clean; deploy production curated. Never drag test trash into client data.
10. **Audit & Export Ready.** Everything you brief a client on can be backed by machine‑traceable sources.

# 3. Agreed Strategic Decisions (Snapshot)

| # | Decision Area | Chosen Direction | Notes |
| --- | --- | --- | --- |
| 1 | **Priority Data Domains** | (1) Congress Legislation & Actions; (2) Federal Funding & Program Eligibility Library; (3) Canonical Client/Jurisdiction Layer | Foundation for all downstream use cases. |
| 2 | **Scope Model** | Hybrid Global (federal intel) + Client Private Overlay (strategy, internal docs) | Enables cross‑learning without data leakage. |
| 3 | **Freshness Cadence** | 4h Legislation; Daily Grants; 48h Regulatory; Quarterly Background | Tunable; see §8. |
| 4 | **Initial Report Types** | Weekly Client Update Brief; On‑Demand Meeting Kit | Highest immediate ROI. |
| 5 | **Authority Hierarchy** | Tiered: Official APIs > Official Docs (.gov) > Trusted Associations > News/Trade > User/Unverified | Drives conflict resolution & auto‑trust gating. |
| 6 | **Cost Strategy** | Tiered Models: small local for classify/triage; frontier for deliverable synthesis; caching everywhere | Scales w/ usage & budget. |
| 7 | **Environment Strategy** | dev / staging / prod namespaces in both DB + Pinecone | Enables clean post‑test purge. |
| 8 | **Bootstrap Data Policy** | Hybrid: authoritative seed data + curated “Mike Brain” overlay (contacts, wins, style prefs) | You are sole user (for now); design for multi‑tenant later. |

These baseline selections drive schema, ingestion design, and runtime behavior.

# 4. Environment & Cutover Strategy

You want to **test aggressively now**, accept dirty data, then wipe clean before serious client onboarding. Rather than a destructive one‑off purge, we formalize environments and simply switch namespaces.

## 4.1 Environments

| Env | Purpose | Data Quality | Reset Policy | Pinecone Namespace | DB Path | Notes |
| --- | --- | --- | --- | --- | --- | --- |
| **dev** | Rapid iteration & debugging | Dirty, mixed | Drop anytime | govsight\_dev | data/dev.db | Default during prototyping. |
| **staging** | Pre‑prod regression & QA | Curated synthetic or scrubbed real | Reset per release | govsight\_stg | data/staging.db | Run test scripts here. |
| **prod** | Real lobbying data | Authoritative & auditable | Never drop (migrate only) | govsight\_prod | data/govsight.db | Used once system trusted. |

## 4.2 Cutover Checklist

1. Tag code baseline (v0\_testfreeze).
2. Finalize DB schema migrations (geo keys, authority tiers, timestamps).
3. Run bootstrap loaders (Congress, programs, geo, client roster seed) into **staging**.
4. Execute regression test pack (Grandview TX mayor, Coachella water programs, etc.).
5. Export curated staging snapshot to versioned artifact (tar/JSON bundle).
6. Provision **prod** namespace; import snapshot.
7. Switch active profile (CLI flag or env var) → prod.

## 4.3 Data Backups & Rollbacks

* Daily sqlite vacuum + gzip snapshot.
* Pinecone export embeddings & metadata to NDJSON (if cost matters, snapshot only metadata + doc sources; recreate embeddings on restore).
* Git‑tracked /bootstrap/seed\_data/ directory contains canonical CSV/JSON loaders for reproducibility.

# 5. System Overview (Conceptual Architecture)

+---------------------------------------------------------------+  
| User (Mike) |  
| CLI / future Web UI / email triggers / scheduled automations |  
+-----------------------+-----------------+---------------------+  
 | |  
 v v  
 +---------------+ +------------------+  
 | Conversation | | Scheduled Agents |  
 | Orchestrator | | (ingest, events) |  
 +-------+-------+ +---------+---------+  
 | |  
 v v  
 +------------------------------+ +-------------------+  
 | Retrieval Planner | | Data Ingestion |  
 | (structured→vector→web) | | Pipelines |  
 +----------+-------------------+ +---------+---------+  
 | |  
 +--------+-------+ +-------+----------+  
 | Structured DB | | Raw Doc Store |  
 | (SQLite/PG) | | (files + meta) |  
 +--------+-------+ +-------+----------+  
 | |  
 v v  
 +--------------+ +--------------+  
 | Fact / Event | | Vector Index |  
 | Tables | | (Pinecone) |  
 +------+-------+ +------+-------+  
 | |  
 +-------------------------------+  
 |  
 v  
 +----------------+  
 | LLM Reasoning |  
 | & Synthesis |  
 +----------------+

**Data Flow Summary:**

1. User asks question.
2. Intent + constraint extraction; resolve to canonical entities.
3. Structured DB queried (facts, events, programs, officials).
4. If insufficient → semantic lookup in vector index.
5. If still thin/stale → targeted web/API fetch.
6. Reasoning layer synthesizes answer w/ provenance & confidence.
7. Facts promoted (pending/curated) into memory; events logged; watchlists updated.

# 6. Data Domain Scoping (Phase 1 Targets)

## 6.1 Federal Congress Data

**Must capture:**

* Bills (number, chamber, title, subject, summary, sponsors).
* Actions timeline (introduced, committee, markup, reported, passed, conferenced, enacted).
* Committee assignments & jurisdictions.
* Voting records.
* CPF/CDS earmark relationships (when present in appropriations bills).

**Sources:** Congress.gov API; GovInfo bulk data; House/Senate Approps committee docs.

**Refresh:** Poll every 4 hrs on business days; incremental by last\_modified.

## 6.2 Federal Funding & Program Eligibility Library

**Programs to seed:**

* EPA: DWSRF, EC‑SDC, SAFER, PFAS‑related.
* DOT: RAISE, INFRA, Safe Streets, bridge/road rehab.
* FEMA: BRIC, HMGP.
* USDA Rural Development utilities grants/loans.
* Interior/Bureau of Reclamation (where applicable to water supply).

**Metadata fields:** assistance listing #; program category; eligibility rules (JSON rules engine); cost share; matching; typical award range; application cycle; links to NOFO docs; historical success rate if available.

**Refresh:** Daily automated check for new NOFOs / guidance.

## 6.3 Canonical Client/Jurisdiction Layer

**For each client municipality or district:**

* Name; type (city, county, district, tribe, utility).
* State; county; FIPS.
* Population; MHI (median household income); disadvantaged status flags (e.g., DAC, SDC, rural low income).
* Infrastructure issue tags (water quality, roads, wildfire, stormwater, etc.).
* Project catalog (known capital needs or asks).
* Linked congressional district(s) + current federal delegation.
* Primary contacts: mayor, city admin, public works, finance, outside consultants.

**Refresh:** Manual + periodic (quarterly) data validation; auto updates when underlying census/demographic feed changes.

# 7. Data Model Outlines (Schema Sketches)

*(Represented in portable SQL; adapt to SQLite first, Postgres later.)*

## 7.1 Entities (Universal Registry)

CREATE TABLE entity (  
 entity\_id TEXT PRIMARY KEY,  
 entity\_type TEXT NOT NULL, -- municipality, program, bill, official, committee, etc.  
 name TEXT NOT NULL,  
 alt\_names TEXT, -- JSON array of aliases  
 parent\_entity\_id TEXT, -- hierarchical link if applicable  
 scope TEXT NOT NULL DEFAULT 'global', -- global|client:<id>  
 created\_at DATETIME DEFAULT CURRENT\_TIMESTAMP,  
 updated\_at DATETIME DEFAULT CURRENT\_TIMESTAMP  
);  
CREATE INDEX idx\_entity\_type ON entity(entity\_type);  
CREATE INDEX idx\_entity\_scope ON entity(scope);

## 7.2 Geo Metadata

CREATE TABLE geo (  
 entity\_id TEXT PRIMARY KEY REFERENCES entity(entity\_id) ON DELETE CASCADE,  
 country TEXT,  
 state TEXT,  
 county TEXT,  
 fips\_code TEXT,  
 lat REAL,  
 lon REAL,  
 population INTEGER,  
 mhi INTEGER,  
 disadvantaged\_flags TEXT -- JSON: {"sdac":true,"rural":false,...}  
);  
CREATE INDEX idx\_geo\_state ON geo(state);  
CREATE INDEX idx\_geo\_fips ON geo(fips\_code);

## 7.3 Officials / Electeds

CREATE TABLE official (  
 official\_id TEXT PRIMARY KEY, -- bioguide\_id or synthetic  
 entity\_id TEXT NOT NULL REFERENCES entity(entity\_id), -- municipality/district they represent  
 title TEXT, -- Mayor, Rep., Senator, Councilmember  
 chamber TEXT, -- House, Senate, Local  
 party TEXT,  
 term\_start DATE,  
 term\_end DATE,  
 contact\_email TEXT,  
 contact\_phone TEXT,  
 website TEXT,  
 notes TEXT,  
 scope TEXT NOT NULL DEFAULT 'global'  
);  
CREATE INDEX idx\_official\_entity ON official(entity\_id);

## 7.4 Bills & Legislative Actions

CREATE TABLE bill (  
 bill\_id TEXT PRIMARY KEY, -- e.g., "HR1234-118"  
 congress INTEGER NOT NULL, -- 118, 119, etc.  
 chamber TEXT NOT NULL, -- House, Senate  
 number TEXT NOT NULL, -- HR1234  
 title\_short TEXT,  
 title\_official TEXT,  
 summary\_short TEXT,  
 summary\_long TEXT,  
 status\_code TEXT, -- introduced, committee, passed\_house...  
 status\_date DATE,  
 last\_action\_desc TEXT,  
 last\_action\_date DATE,  
 sponsor\_official\_id TEXT, -- FK -> official  
 scope TEXT DEFAULT 'global'  
);  
CREATE INDEX idx\_bill\_status\_date ON bill(status\_date);  
CREATE INDEX idx\_bill\_congress ON bill(congress);

CREATE TABLE bill\_action (  
 action\_id INTEGER PRIMARY KEY AUTOINCREMENT,  
 bill\_id TEXT NOT NULL REFERENCES bill(bill\_id) ON DELETE CASCADE,  
 action\_date DATE NOT NULL,  
 action\_text TEXT NOT NULL,  
 chamber TEXT,  
 committee TEXT,  
 action\_code TEXT,  
 source\_uri TEXT  
);  
CREATE INDEX idx\_bill\_action\_date ON bill\_action(bill\_id, action\_date);

## 7.5 Programs & Funding Vehicles

CREATE TABLE program (  
 program\_id TEXT PRIMARY KEY, -- e.g., EPA\_DWSRF  
 agency TEXT NOT NULL,  
 program\_name TEXT NOT NULL,  
 assistance\_code TEXT, -- CFDA/Assistance Listing  
 description TEXT,  
 cost\_share\_min REAL,  
 cost\_share\_max REAL,  
 award\_min REAL,  
 award\_max REAL,  
 cycle\_notes TEXT,  
 url\_guidance TEXT,  
 scope TEXT DEFAULT 'global'  
);

### Program Eligibility Rules (JSON rules engine)

CREATE TABLE program\_eligibility (  
 program\_id TEXT REFERENCES program(program\_id) ON DELETE CASCADE,  
 rule\_id TEXT,  
 rule\_json TEXT, -- e.g., {"state\_in": ["CA","TX"], "population\_lt": 50000, "requires\_water\_system": true}  
 PRIMARY KEY (program\_id, rule\_id)  
);

## 7.6 Client Overlay (Link Clients to Entities & Programs)

CREATE TABLE client (  
 client\_id TEXT PRIMARY KEY, -- e.g., GRANDVIEW\_TX  
 display\_name TEXT NOT NULL,  
 primary\_contact TEXT,  
 email TEXT,  
 phone TEXT,  
 notes TEXT  
);

CREATE TABLE client\_entity\_link (  
 client\_id TEXT REFERENCES client(client\_id) ON DELETE CASCADE,  
 entity\_id TEXT REFERENCES entity(entity\_id) ON DELETE CASCADE,  
 role TEXT, -- owner, interest, region\_overlap, etc.  
 PRIMARY KEY (client\_id, entity\_id)  
);

CREATE TABLE client\_program\_priority (  
 client\_id TEXT,  
 program\_id TEXT,  
 priority INTEGER DEFAULT 3, -- 1=low,5=high  
 notes TEXT,  
 PRIMARY KEY (client\_id, program\_id),  
 FOREIGN KEY (client\_id) REFERENCES client(client\_id),  
 FOREIGN KEY (program\_id) REFERENCES program(program\_id)  
);

## 7.7 Facts (Curated Knowledge Claims)

CREATE TABLE fact (  
 fact\_id TEXT PRIMARY KEY,  
 entity\_id TEXT, -- optional: who this fact is about  
 program\_id TEXT, -- optional  
 bill\_id TEXT, -- optional  
 fact\_text TEXT NOT NULL,  
 fact\_type TEXT, -- e.g., official\_name, mayor, deadline, funding\_award, free\_text  
 value\_json TEXT, -- normalized structured data when extractable  
 confidence REAL DEFAULT 0.5,  
 authority\_tier INTEGER, -- 1=API,2=.gov doc,3=trusted assoc,4=news,5=user  
 source\_uri TEXT,  
 source\_date DATE,  
 last\_verified\_ts DATETIME,  
 scope TEXT DEFAULT 'global',  
 created\_at DATETIME DEFAULT CURRENT\_TIMESTAMP,  
 updated\_at DATETIME DEFAULT CURRENT\_TIMESTAMP  
);  
CREATE INDEX idx\_fact\_entity ON fact(entity\_id);  
CREATE INDEX idx\_fact\_bill ON fact(bill\_id);  
CREATE INDEX idx\_fact\_program ON fact(program\_id);

## 7.8 Events (Change Feed)

CREATE TABLE event (  
 event\_id TEXT PRIMARY KEY,  
 event\_dt DATETIME NOT NULL,  
 event\_type TEXT NOT NULL, -- bill\_action, deadline\_open, deadline\_close, official\_change, funding\_award, correction, etc.  
 entity\_id TEXT,  
 bill\_id TEXT,  
 program\_id TEXT,  
 summary TEXT,  
 details\_json TEXT,  
 source\_uri TEXT,  
 scope TEXT DEFAULT 'global'  
);  
CREATE INDEX idx\_event\_dt ON event(event\_dt);  
CREATE INDEX idx\_event\_type ON event(event\_type);

## 7.9 Watchlists

CREATE TABLE watchlist (  
 watch\_id TEXT PRIMARY KEY,  
 client\_id TEXT, -- null = global  
 entity\_id TEXT, -- municipality, official, program, bill  
 watch\_type TEXT NOT NULL, -- bill\_status, grant\_window, official\_change, reg\_update  
 freq\_hours INTEGER DEFAULT 168, -- default weekly (7\*24)  
 active INTEGER DEFAULT 1,  
 created\_at DATETIME DEFAULT CURRENT\_TIMESTAMP  
);  
CREATE INDEX idx\_watch\_client ON watchlist(client\_id);

# 8. Freshness & Authority Policy (Operational)

## 8.1 Authority Tiers (1 = Highest Trust)

1. **Tier 1:** Official APIs / machine‑readable primary data (Congress.gov API, Grants.gov structured feed, USASpending, SAM.gov).
2. **Tier 2:** Official primary documents from .gov/.mil/.state sources (PDF NOFOs, federal register notices, state water board guidance).
3. **Tier 3:** Trusted sector associations & technical consortia (NACo, NLC, AWWA, SWRCB digests, state municipal leagues).
4. **Tier 4:** High‑reputation media / trade publications.
5. **Tier 5:** User‑provided, meeting notes, hearsay (valuable but unverified).

## 8.2 Freshness Windows (Defaults)

| Data Class | Check Cadence | Stale After | Auto Alert Behavior |
| --- | --- | --- | --- |
| Legislative actions | 4 hrs workdays | 12 hrs | Immediate alert to watchlist subscribers. |
| Funding program NOFO/updates | Daily | 72 hrs | Include in next daily digest; escalate urgent if <30d deadline. |
| Regulatory changes | 48 hrs | 5 days | Alert tagged clients. |
| Demographic / static municipal data | Quarterly | 6 months | Soft reminder only. |
| User‑asserted facts | Verify next use | 60 days | Prompt: “Last confirmed 60+ days ago—refresh?” |

Tunable per client & per data type.

## 8.3 Answer Selection Logic (Pseudo)

if structured\_answer\_found(entity) and not stale:  
 respond(structured)  
else:  
 if structured\_found\_but\_stale:  
 run\_refresh\_fetch(); merge\_updates(); respond\_with\_staleness\_flag()  
 else:  
 semantic\_search(); if weak -> web\_fetch();

## 8.4 Staleness Flags in Responses

* “Verified from Congress.gov as of 2025‑07‑23.” (fresh)
* “Last verified 2025‑06‑11; no newer official update found.” (aged)
* “User‑reported; verification pending.” (unverified)

# 9. Correction Handling & Conflict Resolution

## 9.1 User Correction Detection

Patterns: “that’s wrong”, “actually”, “not X”, “should be Y”, “correction:”, “update:”.

## 9.2 Correction Flow

1. Parse correction statement → candidate structured update (entity, field, value).
2. Insert or update **fact** row w/ authority\_tier=5, confidence=0.9 (weight high because user = domain expert), source\_uri='user://mike'.
3. Scan DB for conflicting facts (same entity + fact\_type) w/ higher or equal authority; mark confidence decreased; add conflict note.
4. Queue **verification task**: poll authoritative feeds; cross‑check.
5. On verification result:
   * Confirmed → promote correction: raise authority tier or mark verified\_at.
   * Rejected → message user; keep both w/ conflict indicator.

## 9.3 Conversation Behavior

On correction, respond immediately reflecting update: “Got it—recorded Bill Houston as Mayor of Grandview, TX. I’ll verify against official records and let you know if that changes.”

# 10. Retrieval & Reasoning Flow (Detailed)

## 10.1 High‑Level Steps

1. **Intent Classification** – chat / info / compare / recall / command / action.
2. **Constraint Extraction** – entities (names, geo, topics, time windows) + verbs (track, compare, generate report).
3. **Entity Resolution** – match extracted strings to canonical entity IDs using geo hints.
4. **Retrieval Planner** chooses step order:
   * Structured DB exact lookups (officials, bill status, deadlines).
   * Fact table query by entity/topic.
   * Vector similarity search (Pinecone) constrained by metadata (state=TX, entity=Grandview).
   * Web/API fallback (only if structured + vector insufficient or stale).
5. **Evidence Pack Assembly** – collect structured rows + doc snippets.
6. **Reasoning / Synthesis** – LLM composes answer; attaches provenance metadata.
7. **Memory Update** – extracted new facts → pending/curated; events → event log; watchlist triggers.

## 10.2 Retrieval Planner Pseudocode

def retrieve(query, constraints):  
 # 1. Structured  
 s = structured\_lookup(constraints)  
 if s.is\_authoritative and s.fresh:  
 return s  
  
 # 2. If structured found but stale, schedule refresh & continue collecting evidence  
 if s and s.is\_authoritative and s.stale:  
 schedule\_refresh(s.keys)  
  
 # 3. Semantic search constrained by geo/entity  
 v = vector\_search(query, constraints, filters={'state': constraints.state, 'entity\_id': constraints.entity\_id})  
 if v.strong():  
 return merge(s, v)  
  
 # 4. Web/API fallback  
 w = web\_fetch(query, constraints)  
 return merge(s, v, w)

## 10.3 Evidence Scoring

Score factors:

* Authority tier
* Freshness recency
* Directness (explicit vs inferred)
* Scope match (client vs global)
* Semantic relevance to question

## 10.4 Answer Confidence Buckets

* **High:** Tier≤2, <7d old, direct match.
* **Medium:** Tier≤3 or verified synthesis across ≥2 sources.
* **Low:** Tier≥4, conflicting, or >90d old no re‑verify.

# 11. Memory Lifecycle

## 11.1 What Goes Into Memory?

* Confirmed structured data (officials, program rules, deadlines).
* Extracted facts from authoritative docs.
* User corrections / priorities / strategy notes (client scope).
* Session summaries: what was discussed; action items; watchlist commands.

## 11.2 What Does NOT Automatically Persist?

* Offhand casual chat unrelated to clients/work (unless flagged).
* Speculative brainstorming unless you say “save this idea.”
* Raw scraped text (store doc ref; chunk to vector index; don’t bloat fact table).

## 11.3 Memory Aging & Pruning

* Auto‑downgrade confidence for unverified Tier‑5 facts older than 6 months.
* Suggest archive of stale client projects not referenced in 12 months.
* Purge orphan facts pointing to deleted entities.

## 11.4 Session Summaries & Rehydration

At session end:

* Summarize key decisions, corrections, new watchlists, open TODOs.
* Store in session table; embed summary for quick retrieval on next run. At session start:
* Load last N session summaries for currently active client context.

# 12. Watchlists & Event Engine

## 12.1 Watchlist Creation Triggers

* Explicit user commands: “track bill HR1234,” “monitor EC‑SDC deadlines for Coachella,” “watch mayoral change in Grandview TX.”
* Implicit detection: repeated queries + high interest flag (“You’ve asked about Chromium‑6 compliance 5 times—add to watchlist?”).

## 12.2 Event Detection Sources

* Congress.gov action feed diff.
* Grants.gov NOFO publish/close dates.
* EPA/State water board program updates.
* Official roster changes (mayor, city manager, utility director).
* User confirmation events (“City approved us to submit CPF”).

## 12.3 Event Handling Pipeline

1. Ingestion job polls sources.
2. Normalize to canonical IDs.
3. Compare to last snapshot; detect new/changed.
4. Insert event row.
5. Match watchlists; queue alert.
6. On next user session (or immediate push if flagged urgent) present delta summary.

## 12.4 Alert Modes (initial CLI, later email)

* **Immediate** (critical legislative move; impending deadline <7d).
* **Digest** (daily/weekly depending on watchlist freq).
* **Session Banner** (“Since your last visit: 3 updates for Coachella, 1 for Grandview TX”).

# 13. Reporting Layer (Initial Templates)

## 13.1 Weekly Client Update Brief (Markdown → PDF/DOCX)

**Sections:**

1. Header (Client name, week ending date, author).
2. Top 3 Federal Actions Relevant to Client.
3. Funding Windows – 30/60/90 Day Outlook.
4. Active Projects & Status vs Federal Pathways.
5. Legislative Tracking Table (Bill | Status | Next Step | Recommended Action).
6. Recommended Advocacy Actions (bullets: call, submit pre‑app, gather data).
7. Appendix: Source log + last verified dates.

**Data Sources:** client priorities; bill table; program deadlines; fact table; events since last report.

## 13.2 On‑Demand Meeting Kit

**Trigger:** “Generate meeting kit for Coachella w/ EPA Region 9 re: Chromium‑6.”

**Sections:**

* Meeting Context (who, when, objective).
* Participant Bios (EPA officials, client delegation, congressional ties).
* Client Snapshot (population, water system, disadvantaged status).
* Project Overview (treatment upgrades, cost estimate, stage, match funds).
* Relevant Programs (EC‑SDC, DWSRF, SAFER) w/ eligibility & deadlines.
* Legislative Alignment (bills, authorizations, appropriations asks).
* Past Engagement History (last contact, commitments).
* Talking Points & Asks.
* Leave‑Behind One Pager (auto‑generated summary block).

## 13.3 Report Export Pipeline

1. Gather structured data.
2. Pull supporting text blocks from vector store (summaries only; not full docs).
3. Render Markdown template.
4. Convert to DOCX/PDF (python‑docx, weasyprint, or pandoc pipeline).
5. Save to /exports/<client>/<date>/ and log event.

# 14. Codebase Refactor Plan (Monolith → Modular)

You currently have a large talk.py script doing everything. We’ll refactor in controlled phases while preserving working behavior.

## 14.1 Target Package Layout

govsight/  
 \_\_init\_\_.py  
 config.py # env, paths, keys, profile mgmt  
 logging\_utils.py  
 models.py # dataclasses / pydantic schema objects  
  
 canonical/  
 congress\_ingest.py # API pull + normalize  
 program\_ingest.py # funding program loaders  
 geo\_loader.py # FIPS/census mapping  
  
 memory/  
 db.py # DB init + migrations + connections  
 fact\_store.py # insert/update/query facts  
 watchlist\_store.py # CRUD watchlists  
 bootstrap.py # seed loaders; env resets  
  
 retrieval/  
 structured.py # DB lookups (officials, bills, programs)  
 semantic.py # Pinecone wrapper + filters  
 web\_search.py # SERP + scrape fetchers  
 planner.py # cascade logic & staleness checks  
  
 nlp/  
 intent.py  
 constraints.py  
 correction\_detector.py  
 summarizer.py  
  
 reasoning/  
 answer\_compose.py # merge evidence into response  
 confidence.py # authority + freshness scoring  
 curation.py # fact gating rules  
  
 reports/  
 weekly\_client\_brief.py  
 meeting\_kit.py  
 export.py  
  
 automation/  
 scheduler.py # periodic jobs  
 event\_diff.py # detect changes & create events  
  
 cli/  
 main.py # entrypoint; replaces talk.py gradually

## 14.2 Refactor Stages

**R0 – Config & Logging Extract**

* Move env parsing + path resolution out of talk.py.
* Add --profile flag (dev/staging/prod).

**R1 – Memory Manager Separation**

* Lift SQLite + Pinecone operations into memory/ modules.
* Standardize Fact + Event inserts.

**R2 – NLP Modules**

* Intent + constraints functions return typed objects; unit testable.

**R3 – Retrieval Planner**

* Single function: plan\_and\_fetch(query, constraints, profile) returns evidence pack.

**R4 – Answer Composer + Curation**

* Centralize synthesis & fact promotion rules; add correction handling hook.

**R5 – CLI Thin Wrapper**

* talk.py becomes shim calling govsight.cli.main(); deprecate once stable.

### Refactor Safety Nets

* Golden output fixtures for common questions.
* Regression harness comparing pre/post refactor answers.
* Trace logging in debug mode.

# 15. Bootstrap Data Strategy (Initial Load)

You asked me to decide what’s best for you as a single user (with future multi‑user potential). Here’s the recommended hybrid bootstrap plan.

## 15.1 Authoritative Baseline Pack (Global Scope)

* Latest Congress (current session) bill list + actions (topline fields + summary short).
* Core federal programs (EPA DWSRF, EC‑SDC, SAFER; FEMA BRIC; DOT RAISE; etc.).
* Federal elected roster (House + Senate) w/ committees.
* Census geo reference for states + major cities (FIPS + population buckets).

## 15.2 Curated “Mike Brain” Overlay (Client Scope = per‑client)

* Client list (Grandview TX, Coachella CA, Lakewood CA, etc.).
* Known project asks & amounts (include wins list for credibility decks).
* Key contacts (emails, phone, notes).
* Policy priority tags per client (water, transportation, public safety).

## 15.3 Developer Sandbox Data (Dev Only)

* Noisy test cases intentionally colliding names (Grandview WA, Grandview TX; Springfield multiples).
* Synthetic conflicting facts to test correction flow.

## 15.4 Import Order

1. Entity registry (global).
2. Geo records.
3. Programs.
4. Bills.
5. Officials.
6. Clients + links.
7. Facts (baseline mayors, deadlines, known wins).
8. Watchlist seeds.

# 16. Usage Workflows (Day in the Life)

## 16.1 Morning Sweep

**Command:** govsight --profile prod morning-digest  
**System:** Pulls 4h legislative deltas, daily grant updates, last 24h events for tracked clients. Produces Markdown summary; optional email.

## 16.2 Quick Client Check (Phone Ring)

City calls: “Where’s our Chromium‑6 funding path?”

* You: gs> status chromium6 coachella (alias).
* Structured DB returns projects + relevant programs; if stale, background refresh; immediate bullet answer.

## 16.3 Prep for Congressional Staff Call

* gs> meeting-kit grandview\_tx rep-elser (example) → generate binder: city snapshot, active asks, bill linkages, last commitments.

## 16.4 Add New Watchlist On the Fly

* “Track AB338 progress in CA Legislature; notify weekly.” → watchlist row created; event engine polls state feed.

## 16.5 Capture Correction

* You: “Correction: Mayor is Bill Houston, not Ashley Lara; that’s Grandview WA.”
* System updates fact, marks conflict, queues verify.

# 17. Implementation Milestones & Timeline (Rough)

| Milestone | Scope | Duration Est. | Dependencies | Exit Criteria |
| --- | --- | --- | --- | --- |
| **M0** | Stabilize current prototype; fix glaring bugs; env profiles | 1 wk | existing code | CLI profile switch; dev DB load works |
| **M1** | Geo keys + correction handling + location gating | 1–2 wks | M0 | Grandview TX test passes every time |
| **M2** | Canonical schema + migration tool; memory modules | 2 wks | M1 | Entities & facts stored in new schema; old data migrated |
| **M3** | Congress ingest + program library loaders | 2–3 wks | M2 | 95% coverage of top programs; incremental sync works |
| **M4** | Watchlist + event engine | 1–2 wks | M3 | Auto alerts fired on test deltas |
| **M5** | Reporting engine (Weekly Brief, Meeting Kit) | 2 wks | M3 | PDF export validated; sample client deck generated |
| **M6** | Refactor completion; talk.py deprecated | 2 wks | M5 | All regression tests green; docs updated |

(Weeks = effort blocks; calendar time depends on availability.)

# 18. Security, Privacy, and Multi‑Tenant Readiness

## 18.1 Current Use Case

* Single user (you) running locally; risk is low inter‑client leakage but still important.

## 18.2 Future Multi‑Client

* Add tenant\_id or scope=client:<id> everywhere.
* Row‑level filtering in queries; retrieval planner must respect scope.
* Redaction layer when generating cross‑client comps.

## 18.3 Secrets Management

* Use .env in dev only.
* Transition to OS keyring or environment‑injected secrets for prod.
* Do not log API keys.

## 18.4 Data Protection

* Encrypt client contact tables at rest if stored on shared systems.
* Option for offline mode (no web calls) when handling sensitive strategy sessions.

# 19. Development Conventions & Style Guide

**Python Version:** 3.11+ recommended.

**Type System:** Pydantic models or dataclasses for all structured payloads (constraints, entities, facts, events). Enforce mypy in CI.

**Logging:** logging\_utils.py w/ structured JSON logs (timestamp, env, turn\_id, intent, retrieval\_plan, token\_cost, error trace).

**Prompt Templates:** Stored in /prompts/\*.jinja (Jinja or simple f‑string safe). Never .format() raw curly‑brace JSON.

**Unit Tests:** pytest; fixtures for entity resolution, correction handling, and retrieval gating.

**Regression Test Data:** YAML scripts describing input → expected structured retrieval decisions (not always full answer text—test logic, not wording).

# 20. Open Design Questions (Track & Resolve)

| ID | Question | Current Thinking | When Decide |
| --- | --- | --- | --- |
| Q01 | How granular should geo resolution be (city vs county vs metro)? | City + State required; county helpful | Before M1 |
| Q02 | Accept fuzzy city name if only one match in tracked clients? | Yes w/ confirmation | M1 |
| Q03 | Multi‑source conflicting deadlines—who wins? | Highest authority tier; else latest doc date | M3 |
| Q04 | Should client strategy notes ever sync upstream into global models? | No; remain private overlay | M2 |
| Q05 | Push notifications (email vs console only)? | Console now; email later | M4 |
| Q06 | Billing metrics (usage by client)? | Capture token & time per scope row | M5 |

# 21. Quick Reference: Command & Intent Patterns

| User Pattern | Intent | Action |
| --- | --- | --- |
| “who is the mayor of <city, state>?” | fact\_lookup | structured lookup → fallback |
| “track” | command\_watchlist | create watchlist |
| “generate weekly brief for” | report\_gen | run report template |
| “update: mayor is” | correction | memory update + verify |
| “compare vs funding” | compare | multi‑entity structured query |

# 22. Example Answer Styles (Client‑Facing vs Internal)

## 22.1 Internal Debug Mode

[INTENT] fact\_lookup  
[ENTITY] Grandview, TX (entity\_id=GRANDVIEW\_TX)  
[STRUCTURED] Mayor=Bill Houston (verified 2025‑07‑20)  
[STATUS] Fresh; no web fetch needed.  
ANSWER: Bill Houston is the current Mayor of Grandview, Texas (confirmed as of July 20, 2025).

## 22.2 Client‑Facing (Short Email Snip)

Bill Houston currently serves as Mayor of Grandview, Texas. Confirmed with the city roster (last verified July 20, 2025). Let me know if you need contact details or a meeting intro.

# 23. Daily Ops Checklist (You)

**Morning (auto or manual):**

* Run digest → scan for client‑relevant deltas.
* Flag urgent deadlines (<30 days) & schedule outreach.

**Before Client Calls:**

* Generate meeting kit; skim contact notes.

**After Calls:**

* Speak corrections/updates directly into GovSight: “Record that Tehama Co RCD asked for Mendocino access permit follow‑up; due Friday.”

**Weekly:**

* Generate weekly briefs for each active retainer client.
* Review watchlist noise; tune frequencies.

# 24. Minimal Bootstrap Checklist (What To Prepare Before Next Session)

You can start gathering these now so ingestion work goes fast once we resume:

### 24.1 Client Roster CSV

Columns: client\_id,display\_name,state,primary\_contact,email,phone,priority\_tags.

### 24.2 Client Projects CSV

Columns: client\_id,project\_name,sector,est\_cost,status,notes.

### 24.3 Funding Wins CSV (for credibility slides)

Columns: client\_id,project\_name,award\_amount,fiscal\_year,program\_id,notes.

### 24.4 Program Seed CSV (start w/ top 10 you use most)

Columns: program\_id,agency,program\_name,assistance\_code,match\_req,award\_range,cycle\_notes,url\_guidance.

### 24.5 Mayor/Official Quick Table

Columns: entity\_name,state,official\_name,title,email,phone,term\_start,term\_end,verified\_date.

Drop these into /bootstrap/seed\_data/ and we’ll import.

# 25. Future Extensions (Phase 2+ / Nice to Have)

* **State Legislature Tracking** (CA, TX, FL) with dual‐chamber bill mirroring.
* **AI Email Draft Assist** that reads last meeting summary + latest events and drafts follow‑up for you to edit.
* **Comparative Benchmarking Engine** (“Where does Coachella stand vs similar desert municipalities on EC‑SDC uptake?”).
* **Voice Input Mode** (dictation → action parsing) for driving or travel.
* **Document QA Chat on Upload** (ask a PDF directly; responses grounded in doc chunks tagged to entity).
* **Dashboard UI** w/ client tiles, red/yellow/green action bars.
* **Mobile SMS Alerts** for urgent deadlines.

# 26. Appendix A – Abbreviation & Term Glossary

| Term | Meaning |
| --- | --- |
| CPF | Community Project Funding (House) – member‑directed funds (earmarks). |
| CDS | Congressionally Directed Spending (Senate earmarks). |
| EC‑SDC | Emerging Contaminants – Small or Disadvantaged Communities (EPA funding lane). |
| DWSRF | Drinking Water State Revolving Fund. |
| SAFER | (CA) Safe and Affordable Funding for Equity and Resilience in drinking water (state program; confirm context). |
| Cr‑6 | Hexavalent Chromium (drinking water contaminant of concern). |
| Watchlist | Subscribed topic for change alerts (bill status, deadline, official change). |
| Authority Tier | Trust level of source (1 highest). |
| Scope | global vs client‑scoped data partition. |

# 27. Appendix B – Example Config File (.toml Sketch)

[global]  
timezone = "America/New\_York"  
model\_tier\_high = "gpt-4.2" # example placeholder  
model\_tier\_mid = "gpt-4o-mini"  
model\_local\_small = "llama3.2-8b" # if running local  
  
[profiles.dev]  
db\_path = "data/dev.db"  
pinecone\_namespace = "govsight\_dev"  
auto\_web = true  
  
[profiles.staging]  
db\_path = "data/staging.db"  
pinecone\_namespace = "govsight\_stg"  
auto\_web = true  
  
[profiles.prod]  
db\_path = "data/govsight.db"  
pinecone\_namespace = "govsight\_prod"  
auto\_web = false # require explicit when