ATLAS Architecture: A Framework for Autonomous AI Verification in Health, Fitness, and Content Domains (January 2026)

Executive Summary

The conceptualization and deployment of ATLAS, an autonomous AI life assistant, necessitates a fundamental architectural departure from contemporary cloud-dependent systems. Operating within the high-stakes domains of physiological health, nutritional biochemistry, and professional content creation requires a rigorous adherence to "grounded AI"—a paradigm where generative outputs are strictly bounded by verified external truths. The constraints imposed on this system—specifically a sub-500 millisecond latency budget, a negligible operational cost of under $5 per month for verification, and absolute privacy preservation—render standard commercial API integration strategies obsolete. As of January 2026, the reliance on synchronous REST API calls to proprietary data siloes such as DrugBank or commercial plagiarism checkers is financially and technically untenable for a consumer-grade autonomous agent.

This report presents a comprehensive architectural blueprint for the "Local-First, Retrieval-Augmented Verification" stack. By inverting the traditional cloud dependency model, ATLAS can achieve expert-level verification through the ingestion and local indexing of high-quality open data from government and non-profit scientific sources. We analyze the specific implementation pathways for integrating the Allen Institute's Supp.AI for pharmacological interactions, the USDA FoodData Central for nutritional analysis, and ACSM-derived logic for exercise contraindications. Furthermore, we explore the utilization of quantized local language models and vector databases to ensure content integrity without external data leakage. The analysis confirms that by leveraging the proliferation of "Edge AI" technologies and the 2025 regulatory clarity provided by the FDA's Clinical Decision Support guidance, ATLAS can be engineered as a compliant, safe, and highly responsive system that operates fully offline.

1. Architectural Strategy: The Local-First Verification Paradigm

The foundational challenge in architecting ATLAS lies in the "Iron Triangle" of the specified constraints: extreme cost efficiency, real-time latency, and uncompromising privacy. Traditional web architectures, which rely on on-demand queries to third-party endpoints, fail to meet these simultaneous demands. A single synchronous API call to a service like DrugBank or a commercial plagiarism detector can introduce 200ms to 800ms of network latency, shattering the <500ms total response budget. Furthermore, the transmission of user health data—such as medication lists or cardiovascular history—to external servers constitutes a privacy breach that contradicts the system's core mandate.

1.1 The Inverted Cloud Model

To resolve these conflicts, ATLAS must adopt an "Inverted Cloud" or "Local-First" architecture. In this model, the application does not go to the data; the data comes to the application. Rather than treating external databases as services to be queried in real-time, we treat them as static artifacts to be ingested, normalized, and indexed locally within the ATLAS runtime environment. This approach decouples the system's operational latency from network conditions and third-party server performance, effectively reducing verification time to the speed of local disk I/O or memory access—typically in the microsecond or low millisecond range.

This architectural shift allows ATLAS to leverage the massive growth in "Open Science" datasets available in 2026. Agencies like the FDA, USDA, and research institutes like the Allen Institute publish comprehensive data dumps that are updated periodically (quarterly or monthly). By constructing a simplified ETL (Extract, Transform, Load) pipeline that runs asynchronously—perhaps once a month via a background cron job—ATLAS can maintain an up-to-date "Ground Truth" database without incurring per-query costs. The sub-$5 budget is thus allocated entirely to the localized compute resources (storage and CPU) required to host these indices, rather than being dissipated on variable API licensing fees.

1.2 The Verification-Generation Loop

The operational workflow of ATLAS functions as a dual-process system, mirroring the cognitive distinction between "System 1" (fast, intuitive generation) and "System 2" (slow, deliberative verification), though optimized for machine speed.

1. **Intent Parsing and Entity Extraction:** Upon receiving a user query, a lightweight, quantized local model (such as a 4-bit quantized Llama 3 or a specialized BERT-based Named Entity Recognition model) identifies the core semantic entities. For example, in the query "I'm taking Warfarin and want to start a high-protein diet with spinach salads," the system extracts , `[Nutrient: Protein]`, and .
2. **Parallel Asynchronous Verification:** Before the Generative Core constructs a response, parallel threads query the local verification databases.
    - *Thread A (Pharmacology):* Queries the local SQLite database derived from Supp.AI and OpenFDA for interactions between Warfarin and Spinach (specifically Vitamin K).
    - *Thread B (Nutrition):* Queries the local DuckDB instance for the macronutrient profile of Spinach.
    - *Thread C (Safety):* Checks the user's health profile against ACSM contraindications for dietary changes.

3. **Constraint Injection:** The results of these queries are injected into the context window of the Large Language Model (LLM). If a critical interaction is found, the prompt is augmented with a hard constraint: "CRITICAL WARNING: Warfarin interacts with high Vitamin K foods like spinach. Do not recommend this meal plan."
4. **Grounded Generation:** The LLM generates the final response, which is now mathematically bound by the retrieved facts. This Retrieval-Augmented Generation (RAG) pattern ensures that the AI does not hallucinate safety but reports the deterministic findings of the verification layer.

1.3 Privacy by Design and Offline Capability

The Local-First architecture inherently satisfies the strict privacy and offline fallback requirements. Since all "Ground Truth" databases—drug interactions, nutritional values, exercise rules—are stored on the device or a self-hosted edge server, no user data ever leaves the controlled environment. A user can inquire about sensitive topics, such as interactions between antidepressants and illicit substances, with the assurance that no query logs are being generated on third-party servers. Furthermore, should internet connectivity be severed, the local indices remain fully accessible, ensuring that ATLAS functions as a reliable life assistant even in off-grid scenarios.

2. Pharmacological Grounding: The Health Verification Layer

The domain of health and pharmacology represents the highest risk vector for ATLAS. The potential for "hallucination"—where an AI confidently asserts the safety of a dangerous drug combination—necessitates a deterministic and rigorous verification strategy. As of 2026, the landscape of biomedical data has bifurcated into expensive proprietary silos and robust open-science initiatives. For a low-budget, privacy-focused assistant, the latter is the only viable path.

2.1 The Supp.AI and Allen Institute Ecosystem

The Allen Institute for AI (Ai2) has established itself as a cornerstone of open biomedical research. Their project, Supp.AI, specifically targets the nebulous domain of supplement-drug interactions, an area often neglected by traditional pharmaceutical databases.[1] Unlike conventional databases that rely on slow, manual curation by human experts, Supp.AI leverages advanced Natural Language Processing (NLP) models, specifically RoBERTa-DDI, to automatically extract interaction evidence from the vast corpus of scientific literature (over 22 million papers from Semantic Scholar).[3]

2.1.1 Data Accessibility and Licensing

Supp.AI provides its entire dataset as a bulk download, typically a compressed JSON or CSV file (approx. 38.4 MB). This file contains a graph of "Agents" (supplements and drugs) and the

"Edges" (interactions) connecting them, along with confidence scores and excerpts from the source literature.[3] This downloadable format is critical for ATLAS, as it allows for the ingestion of over 60,000 interaction pairs into a local graph database without requiring an API connection or subscription fee. The data is generally available for research and non-commercial use, which aligns with the "Autonomous Assistant" persona provided ATLAS strictly cites the source and does not resell the raw data.

2.1.2 Implementation: The Interaction Graph

To implement Supp.AI data locally, ATLAS should utilize a graph-based schema or a relational approximation within SQLite. The cui_clusters.json file provided by the Allen Institute groups various synonyms (e.g., "Vitamin C", "Ascorbic Acid") under unique Concept Unique Identifiers (CUIs).

- **Ingestion Logic:** The ingestion script must map these CUIs to the user-facing terms.
- **Query Pattern:** When a user inputs a supplement, ATLAS resolves the term to its CUI and queries the interactions table for any edges connecting to the CUIs of the user's current medication list.
- **Evidence Presentation:** Because Supp.AI is probabilistic (derived from NLP extraction), the system should expose the "evidence snippets" to the user. For example: "Supp.AI identified a potential interaction based on a 2018 study mentioning '...garlic supplements increased clotting time in patients on warfarin...'".[4] This transparency allows the user to judge the relevance of the data.

2.2 OpenFDA and Government Regulatory Data

While Supp.AI covers supplements, the bedrock of pharmaceutical safety lies in the official FDA labeling. The **openFDA** initiative provides public access to high-value datasets, including the structured product labeling (SPL) and adverse event reporting system (FAERS).[5]

2.2.1 The Limits of the API

While openFDA offers an API, relying on it for real-time verification introduces unacceptable latency and potential privacy leaks. The API is rate-limited, and its response times can fluctuate based on government server load. Furthermore, the "Adverse Event" endpoint (drug/event) often reflects raw, unverified reports that may lag by three months or more.[5] Therefore, ATLAS should strictly utilize the **Bulk Download** capability of openFDA.

2.2.2 The drug/label Dataset

The most valuable dataset for ATLAS is the drug/label JSON dump. This dataset contains the full text of the package inserts for all FDA-approved drugs, structured into fields such as warnings, contraindications, drug_interactions, and boxed_warnings.[6]

- **Processing Strategy:** ATLAS must download this JSON dump (which can be several gigabytes) and parse it into a local Full-Text Search (FTS) index using SQLite’s FTS5 extension.

- **Verification Logic:** When a user asks about "Ibuprofen," the system searches the FTS index for the term "Ibuprofen" in the drug_interactions field of their other medications. If a hit occurs, it triggers a warning. This text-based search is crude but highly effective as a safety net, catching warnings that structured databases might miss.

2.3 The DrugBank Constraint

The initial query specifically requested research into DrugBank. The analysis of DrugBank's 2026 licensing terms reveals it is unsuitable for ATLAS under the <$20/month budget constraint.

- **Licensing Barriers:** DrugBank's API and commercial datasets are strictly paywalled. The "free" access is limited to academic and non-profit research, often with a prohibition on use in "products" or "services".[8]
- **Enforcement:** DrugBank aggressively protects its intellectual property, and using their data in an autonomous assistant without a commercial license could invite legal action.
- **Alternative:** The combination of Supp.AI (for supplements) and OpenFDA (for drugs) provides a sufficient "safety overlapping coverage" without the legal and financial exposure of DrugBank.

2.4 RxNorm: The Universal Translator

A critical challenge in pharmacological verification is nomenclature. Users may say "Advil," "Motrin," or "Ibuprofen." To query Supp.AI (which uses CUIs) or OpenFDA (which uses generic names), ATLAS needs a normalization layer.

- **Solution:** The National Library of Medicine (NLM) provides **RxNorm**, a standardized nomenclature for clinical drugs.[10]
- **Implementation:** ATLAS should ingest the RxNorm Release Files (RRF) locally. These files provide a mapping table where "Advil" (Brand Name) links to "Ibuprofen" (Ingredient) and its corresponding RxCUI.
- **Cost:** Free (requires a UMLS license agreement, which is free for most US-based and international uses).[12]

2.5 Regulatory Compliance: FDA CDS Guidance (2025)

In January 2025, the FDA finalized its guidance on **Clinical Decision Support (CDS)** software, clarifying the regulatory status of AI health assistants.[13] This guidance is paramount for ATLAS to avoid being classified as a regulated Class II medical device.

2.5.1 The "Non-Device" Criteria

To remain unregulated (and thus feasible on a low budget), ATLAS must meet the four criteria for "Non-Device CDS" functions:

1. **Not for Medical Image Processing:** ATLAS handles text, not X-rays or EKGs.
2. **Display of Medical Information:** ATLAS displays established medical facts.
3. **Recommendation to HCP or Informed Patient:** The software provides recommendations to a user who can independently verify the basis of the recommendation.
4. **Transparency:** This is the critical requirement. ATLAS must **explain the rationale** behind every health output.[13]

2.5.2 Transparency Requirements for ATLAS

The 2025 guidance emphasizes that the user should not be forced to rely on the AI's "black box" judgment.

- **Actionable Implementation:** When ATLAS flags a drug interaction, it *must* cite the source (e.g., "Source: FDA Package Insert for Warfarin, Section 7"). It cannot simply say "This is unsafe."
- **Bias and Equity:** The 2025 draft guidance on AI-enabled devices also highlights the need to address bias.[16] ATLAS should be transparent about the limitations of its data (e.g., "Supp.AI data may not cover all herbal supplements used in non-Western medicine").
- **Labeling:** The interface must explicitly label outputs as "Informational References" and link to the original data sources (OpenFDA, PubMed), ensuring the user retains the ultimate decision-making authority.[17]

3. Nutritional Intelligence: Grounding Dietary Outputs

For ATLAS to generate meal plans and analyze nutrition, it requires access to a massive, granular database of food composition. Generative models are notoriously poor at accurate arithmetic and specific nutrient values (e.g., distinguishing between "raw spinach" and "boiled spinach" micronutrients).

3.1 USDA FoodData Central (FDC)

The USDA FoodData Central is the gold standard for nutritional data in the United States, providing detailed profiles for thousands of generic foods (Foundation Foods) and branded items.[18]

3.1.1 Avoiding the API Bottleneck

The USDA API has a default rate limit of 1,000 requests per hour.[19] For an AI assistant that might need to analyze a weekly meal plan containing hundreds of ingredients in a single inference pass, this limit is a choke point. Furthermore, API latency would slow down the generation process.

3.1.2 Bulk Data Engineering

The USDA provides its full datasets in CSV and JSON formats.[20] The "Foundation Foods" dataset (updated late 2025) is the most critical for deep nutritional analysis (micronutrients, amino acids), while "SR Legacy" provides broader coverage for common items.

- **Implementation:** ATLAS should use the CSV bulk download. These files are relational; a food.csv links to nutrient.csv via a food_id.
- **Data Size:** The datasets can be hundreds of megabytes. While manageable, querying raw CSVs in Python is inefficient for real-time lookups.

3.2 Open Food Facts (OFF)

For branded products (e.g., specific protein bars, frozen meals), the community-driven **Open Food Facts** database is superior. It contains over 3 million products.

- **Data Volume:** The database export is available as a CSV or JSONL file. The JSONL file is massive (>40GB uncompressed), posing a storage and query challenge for a low-cost device.[21]
- **Updates:** OFF data is updated daily. ATLAS can perform a "differential update" or a full monthly sync depending on bandwidth.

3.3 The DuckDB Architecture

To manage these large datasets (USDA + Open Food Facts) on a consumer device or cheap VPS without running a heavy database server like PostgreSQL, **DuckDB** is the optimal technology selection for 2026.

- **Technology Profile:** DuckDB is an in-process SQL OLAP database. It can execute SQL queries directly on CSV or Parquet files stored on disk, utilizing vectorized execution for extreme speed. It does not require a background server process, fitting the "offline" and "low resource" constraints perfectly.[22]
- **ETL Pipeline:**
  1. **Download:** Once a month, the ATLAS update script downloads the latest USDA CSVs and Open Food Facts JSONL.
  2. **Conversion:** The script converts these bulky text files into **Parquet** files. Parquet is a columnar storage format that offers high compression (reducing the 40GB OFF file significantly) and allows DuckDB to read only the necessary columns (e.g., just "calories" and "protein") during a query, ignoring the rest.
  3. **Querying:** During inference, ATLAS executes a SQL query via DuckDB to fetch nutrient data.

Python

```python
# Example DuckDB Query (Pseudocode)
SELECT energy_kcal, proteins_100g
FROM 'food_data.parquet'
WHERE product_name LIKE '%Greek Yogurt%'
LIMIT 1
```

This operation typically completes in <20ms, orders of magnitude faster than an API call and effectively free of cost.

3.4 Unit Conversion Logic

A common failure mode in AI nutrition is unit mismatch (e.g., "1 cup" vs "100g"). The USDA datasets include a portions.csv file that defines weights for common measures.[20] ATLAS must implement a rigorous unit conversion layer using this data.

- **Logic:** If the user asks for "1 cup of oats," ATLAS looks up the gram weight of "1 cup" for the specific oat entry in USDA data, then calculates nutrients based on the 100g standard. This arithmetic must be handled by Python logic, not the LLM, to ensure precision.

4. Physiological Safety: Fitness and Exercise Constraints

The domain of fitness introduces the risk of physical injury. Unlike the structured data of pharmacology, exercise safety guidelines are often found in unstructured text (guidelines, textbooks).

4.1 ACSM Guidelines as a Logic Engine

The **American College of Sports Medicine (ACSM)** provides the industry-standard risk stratification guidelines. While the **American Council on Exercise (ACE)** is also reputable, ACSM's clinical guidelines are generally considered the gold standard for medical contraindications.[23]

4.1.1 Translating Text to Logic

Since there is no "ACSM API," ATLAS must internalize these guidelines as a **Rule-Based System**. The ACSM *Guidelines for Exercise Testing and Prescription* define absolute and relative contraindications.[25]

- **Absolute Contraindications (Stop Rule):** Acute myocardial infarction (<2 days), uncontrolled cardiac arrhythmia, acute pulmonary embolus.
- **Relative Contraindications (Conditional Rule):** Systolic BP > 200 mmHg, Diastolic BP > 110 mmHg, electrolyte abnormalities.[25]

4.1.2 JSON Logic Implementation

To integrate this into ATLAS, we convert these prose rules into **JSON Logic**, a format that is portable, secure, and executable.

- **Example Rule:**
  JSON
  ```
  {
  ```

```
 "if": [
  { "or": [
      { "==": [{ "var": "conditions.acute_mi" }, true] },
      { ">": [{ "var": "vitals.systolic_bp" }, 200] }
    ]
  },
  "ABSOLUTE_CONTRAINDICATION",
  "SAFE_TO_PROCEED"
 ]
}
```

- **Execution:** Before generating any workout plan, ATLAS runs the user's health profile through this logic engine. This operates in sub-millisecond time and ensures that the AI never suggests exercise to a user in a hypertensive crisis.

4.2 Exercise Databases

For the actual library of exercises (to populate workouts), ATLAS needs a structured database of movements, muscles worked, and equipment.

- **Wger:** Wger is a prominent open-source exercise database. It offers a REST API but, crucially for ATLAS, allows for **self-hosting**. ATLAS can download the Wger dataset (exercises and images) and serve them locally. This avoids external API dependencies and ensures images are always available offline.[27]
- **Open Source Repositories:** GitHub repositories such as yuhonas/free-exercise-db offer curated JSON datasets of exercises (~800 items) with public domain images.[30] This is the most cost-effective "Database" for ATLAS, as it is simply a JSON file to be ingested.
- **Rejection of ExRx:** The research indicates that ExRx, while comprehensive, is restrictive regarding API use and scraping.[31] To respect intellectual property and avoid legal constraints, ATLAS should rely on Wger or the GitHub open datasets.

4.3 Wearable Integration and OpenEHR

To ground fitness advice in the user's *actual* physiological reality, ATLAS needs to ingest data from wearables (Apple Watch, Garmin).

- **Standardization:** Using the **OpenEHR** standard provides a robust way to model this data. OpenEHR archetypes for "Physical Activity" and "Heart Rate" allow ATLAS to normalize data from different devices into a common format.[33]
- **Privacy:** This ingestion happens via a local bridge (e.g., running on the user's phone), ensuring that minute-by-minute heart rate data is never transmitted to a cloud server, satisfying the strict privacy constraint.

5. Content Integrity: Plagiarism and Quality

Verification

The "Content Creation" capability of ATLAS requires mechanisms to ensure that the text generated is original, high-quality, and structurally sound for the target platform.

5.1 Local Plagiarism Detection

Commercial plagiarism checkers (Copyscape, Copyleaks) operate on a per-page fee model that violates the <$5/month budget.[35] Furthermore, checking against the "whole internet" is impossible offline. Therefore, ATLAS must implement **Local Originality Verification**.

5.1.1 Winnowing Algorithm

For detecting exact-match plagiarism (e.g., if the LLM regurgitates a memorized training example), ATLAS can employ the **Winnowing Algorithm**, widely used in code plagiarism tools like MOSS and implemented in Python libraries like copydetect.[37]

- **Mechanism:** This algorithm creates "fingerprints" of the generated text and compares them against a local "Reference Corpus" (which could include the user's past writings to prevent self-plagiarism, or a downloaded set of high-ranking competitor articles).
- **Efficiency:** It is extremely fast and computationally lightweight, suitable for the latency budget.

5.1.2 Semantic Similarity (Vector Search)

To catch paraphrased content, ATLAS utilizes **Sentence Transformers** (SBERT).

- **Model:** The all-MiniLM-L6-v2 model is ideal for this. It is small (80MB), fast (<50ms inference on CPU), and highly accurate.[38]
- **Workflow:**
  1. ATLAS generates a draft.
  2. The SBERT model converts the draft into a 384-dimensional vector.
  3. This vector is compared (via Cosine Similarity) against the vector index of the user's content archive.
  4. If the similarity score exceeds a threshold (e.g., 0.85), the content is flagged as redundant.

5.2 AI Hallucination and Quality Checks

To ensure the content doesn't "sound" like AI and meets quality standards, ATLAS employs local evaluation models.

5.2.1 Offline Evaluation with RAGAS

RAGAS is a framework for evaluating Retrieval Augmented Generation pipelines.[39] While typically used for development, ATLAS can run a lightweight version of RAGAS metrics (like

"Faithfulness" or "Answer Relevance") offline to self-audit its outputs.

- **Implementation:** By using a small local model (SLM) as the "judge," ATLAS can score its own draft for factual consistency against the retrieved verification data (from Supp.AI or USDA) before showing it to the user.

5.2.2 AI Detection (RoBERTa)

To verify if the content passes as human-written, ATLAS can run a local AI detector.

- **Model:** A fine-tuned RoBERTa-base-openai-detector or similar from HuggingFace.[41]
- **Optimization:** Running a full RoBERTa model can be slow (200-400ms on CPU). To meet the <500ms budget, ATLAS must use **Quantization**. By converting the model to **INT8** format using **ONNX Runtime**, inference latency can be reduced to ~50-100ms with negligible accuracy loss.[43] This allows for a "Reverse Turing Test" on every generated paragraph.

5.2.3 Platform Constraints

For social media content, "truth" also means adhering to technical constraints.

- **TikTok/Reels:** ATLAS must verify that video scripts or generated media descriptions adhere to platform limits (e.g., character counts, aspect ratios like 9:16). This "Platform Truth" database is maintained as a static JSON config file updated with API changes.[45]

6. Regulatory Framework and Ethical Transparency

The operational landscape for AI health assistants changed significantly with the FDA's 2025 guidance.

6.1 FDA Clinical Decision Support (CDS) Guidance

The FDA's final guidance clarifies that software functions are *not* medical devices if they enable the user to independently review the basis of the recommendation.[13]

- **Implication:** ATLAS is technically a CDS. To stay in the unregulated "safe harbor," it must not present itself as an authoritative "black box."
- **Transparency Requirement:** Every health-related output must be accompanied by citations. The user interface should present the recommendation alongside the raw data (e.g., "Recommended Vitamin C intake: 90mg").
- **Language:** The system must use "hedging" language (e.g., "Data suggests..." rather than "You must...").

6.2 Bias Mitigation

The 2025 draft guidance on AI-enabled devices emphasizes the management of bias.[16] ATLAS relies on datasets (Supp.AI, OpenFDA) that may have inherent biases (e.g., underrepresentation of certain populations in clinical trials).

- **Mitigation Strategy:** ATLAS should include a "Data Confidence" metric. If a user asks about a supplement where the Supp.AI data is sparse (few citations), the system should explicitly state: "Limited data available. Confidence: Low."

7. Implementation Roadmap and Pricing

7.1 The Technology Stack

Component	Technology	Role	Status/Cost
Orchestrator	Python (FastAPI)	Logic Controller	Free / Open Source
Database (Health)	SQLite + FTS5	Drug Interactions	Free / Open Source
Database (Nutrition)	DuckDB + Parquet	Nutritional Analysis	Free / Open Source
Logic Engine	JSON Logic	Fitness Safety Rules	Free / Open Source
Vector Store	FAISS / Numpy	Content Similarity	Free / Open Source
Local Models	ONNX Runtime (Quantized)	AI Detection / NER	Free / Open Source
Datasets	Supp.AI, OpenFDA, USDA	Ground Truth	Public Domain / Open

7.2 Latency Budget Analysis

To achieve the <500ms target, the system relies on the speed of local indices.

- **0ms - 10ms:** Request receipt and entity extraction (Quantized NER).
- **10ms - 100ms:** Parallel Verification.
  - *SQLite Query:* <1ms.
  - *DuckDB Query:* <20ms.
  - *SBERT Encoding (Content):* ~50ms.
- **100ms - 400ms:** LLM Inference (Small Language Model, e.g., Mistral/Llama-3-8B 4-bit).
  - *Note:* On a low-end device, generating a long response might exceed 500ms.

However, the *Time to First Token* (TTFT) and the *Verification Step* will be well within the limit. The system can stream the response, providing immediate feedback.

7.3 Cost Analysis

Item	Monthly Cost	Rationale
Verification APIs	$0.00	All data is ingested via bulk download.
Hosting (VPS)	$4.00 - $5.00	A basic VPS (2 vCPU, 4GB RAM) from providers like Hetzner or DigitalOcean is sufficient for this "Inverted Cloud" stack.
Storage	Included	~20GB storage is needed for the Parquet/SQLite files.
Total	**~$5.00**	**Meets Budget Constraint.**

7.4 Deployment Strategy

1. **Phase 1: Data Ingestion:** Build the ETL scripts to download and normalize Supp.AI, OpenFDA, and USDA datasets.
2. **Phase 2: Indexing:** Convert CSVs to Parquet and JSONs to SQLite FTS indices.
3. **Phase 3: Logic Implementation:** Write the JSON Logic rules for ACSM contraindications.
4. **Phase 4: Integration:** Connect the Local Verification Layer to the LLM via a RAG pipeline.
5. **Phase 5: Compliance Audit:** Review all outputs against FDA transparency guidelines.

8. Conclusion

The construction of ATLAS within the specified constraints is not only feasible but represents a more robust and privacy-conscious architecture than typical cloud-native alternatives. By rejecting the dependency on expensive, high-latency commercial APIs and instead embracing a "Local-First" methodology grounded in high-quality open data (Supp.AI, OpenFDA, USDA), ATLAS can deliver expert-level verification for under $5 per month. The integration of DuckDB for nutritional analytics, SQLite for pharmacological safety, and quantized local models for content integrity creates a system that is fast, private, and resilient. This approach aligns perfectly with the 2026 regulatory landscape, positioning ATLAS as a compliant Clinical Decision Support tool that empowers users with verified, transparent, and actionable intelligence.

Works cited

1. News - Allen Institute, accessed on January 5, 2026, https://alleninstitute.org/news/
2. Ai2: Truly open breakthrough AI, accessed on January 5, 2026, https://allenai.org/
3. SUPP.AI by AI2, accessed on January 5, 2026, https://supp.ai/
4. Data for SDI detection (SUPP.AI) - GitHub, accessed on January 5, 2026, https://github.com/allenai/sdi-detection
5. Drug Adverse Event Overview - openFDA, accessed on January 5, 2026, https://open.fda.gov/apis/drug/event/
6. About the openFDA API, accessed on January 5, 2026, https://open.fda.gov/apis/
7. Drug API Endpoints - openFDA, accessed on January 5, 2026, https://open.fda.gov/apis/drug/
8. Discovery API Reference | DrugBank Help Center, accessed on January 5, 2026, https://docs.drugbank.com/discovery/v1/
9. Terms of Use | DrugBank Trust Center, accessed on January 5, 2026, https://trust.drugbank.com/drugbank-trust-center/terms-of-use
10. RxNorm Frequently Asked Questions - National Library of Medicine, accessed on January 5, 2026, https://www.nlm.nih.gov/research/umls/rxnorm/faq.html
11. RxNorm - Dataset - Catalog - Data.gov, accessed on January 5, 2026, https://catalog.data.gov/dataset/rxnorm-3180d
12. RxNorm Overview - National Library of Medicine - NIH, accessed on January 5, 2026, https://www.nlm.nih.gov/research/umls/rxnorm/overview.html
13. Clinical Decision Support Software - Guidance - FDA, accessed on January 5, 2026, https://www.fda.gov/regulatory-information/search-fda-guidance-documents/clinical-decision-support-software
14. Clinical Decision Support Software Frequently Asked Questions (FAQs) - FDA, accessed on January 5, 2026, https://www.fda.gov/medical-devices/software-medical-device-samd/clinical-decision-support-software-frequently-asked-questions-faqs
15. FDA Releases Significantly Revised Final Clinical Decision Support Software Guidance and Related Changes | Advisories | Arnold & Porter, accessed on January 5, 2026, https://www.arnoldporter.com/en/perspectives/advisories/2022/10/fda-releases-significantly-revised-final-clinical
16. FDA Issues Comprehensive Draft Guidance for Developers of Artificial Intelligence-Enabled Medical Devices, accessed on January 5, 2026, https://www.fda.gov/news-events/press-announcements/fda-issues-comprehensive-draft-guidance-developers-artificial-intelligence-enabled-medical-devices

17. FDA Releases Draft Guidance on AI-Enabled Medical Devices | Insights, accessed on January 5, 2026, https://www.gtlaw.com/en/insights/2025/1/fda-releases-draft-guidance-on-ai-enabled-medical-devices
18. FAQs - USDA FoodData Central, accessed on January 5, 2026, https://fdc.nal.usda.gov/faq
19. FoodData Central API Guide, accessed on January 5, 2026, https://fdc.nal.usda.gov/api-guide
20. Downloadable Data - USDA FoodData Central, accessed on January 5, 2026, https://fdc.nal.usda.gov/download-datasets
21. Data, API and SDKs - Open Food Facts, accessed on January 5, 2026, https://world.pro.openfoodfacts.org/data
22. DuckDB & Open Food Facts: the largest open food database in the palm of your hand | by Jeremy Arancio | Medium, accessed on January 5, 2026, https://medium.com/@jeremyarancio/duckdb-open-food-facts-the-largest-open-food-database-in-the-palm-of-your-hand-0d4ab30d0701
23. ACSM Certified Exercise Physiologist Exam Content Outline, accessed on January 5, 2026, https://acsm.org/wp-content/uploads/2024/12/ACSM-Certified-Exercise-Physiologist-Exam-Content-Outline.pdf
24. ACSM-EP Crosswalk, accessed on January 5, 2026, https://acsm.org/wp-content/uploads/2025/01/acsm-ep-crosswalk.pdf
25. Exercise Standards for Testing and Training | Circulation - American Heart Association Journals, accessed on January 5, 2026, https://www.ahajournals.org/doi/10.1161/cir.0b013e31829b5b44
26. Standards for Physical Activity and Exercise in the Cardiovascular Population 2023 4th Edition - ACPICR, accessed on January 5, 2026, https://www.acpicr.com/data/Page_Downloads/ACPICR2023StandardsReaderlayout.pdf
27. wger Workout Manager - Features, accessed on January 5, 2026, https://wger.de/
28. wger Workout Manager API, accessed on January 5, 2026, https://www.facts.dev/api/wger-workout-manager-api/
29. Any suggestions for an open source self hosted fitness/diet/goal keeping solution? : r/selfhosted - Reddit, accessed on January 5, 2026, https://www.reddit.com/r/selfhosted/comments/w5vljo/any_suggestions_for_an_open_source_self_hosted/
30. yuhonas/free-exercise-db: Open Public Domain Exercise Dataset in JSON format, over 800 exercises with a browsable public searchable frontend - GitHub, accessed on January 5, 2026, https://github.com/yuhonas/free-exercise-db
31. In case you weren't aware, exrx.net has a huge database of, accessed on January 5, 2026, https://news.ycombinator.com/item?id=23745572
32. AmeriGEO Gateway to Earth Insights and Intelligence., accessed on January 5, 2026, https://ogc-dp23.voyagersearch.com/navigo/show?id=51dd4965-fcd6-4d1d-bcde-30d93845de27&disp=D189D097B515

33. Observation Archetype: Physical Activity (EMPOWER) - Clinical Knowledge Manager, accessed on January 5, 2026, https://ckm.openehr.org/ckm/archetypes/1013.1.2051
34. Physical activity archetypes; exercise, steps etc from apps & devices - Clinical - openEHR, accessed on January 5, 2026, https://discourse.openehr.org/t/physical-activity-archetypes-exercise-steps-etc-from-apps-devices/983
35. Copyscape Plagiarism Checker - Duplicate Content Detection Software, accessed on January 5, 2026, https://www.copyscape.com/
36. Plagiarism Checker API | AI Detector API - Copyleaks, accessed on January 5, 2026, https://copyleaks.com/api
37. copydetect - PyPI, accessed on January 5, 2026, https://pypi.org/project/copydetect/
38. sentence-transformers - PyPI, accessed on January 5, 2026, https://pypi.org/project/sentence-transformers/
39. Evaluate RAG pipeline using Ragas in Python with watsonx - IBM, accessed on January 5, 2026, https://www.ibm.com/think/tutorials/evaluate-rag-pipeline-using-ragas-in-python-with-watsonx
40. Ragas, accessed on January 5, 2026, https://docs.ragas.io/en/stable/
41. openai-community/roberta-base-openai-detector - Hugging Face, accessed on January 5, 2026, https://huggingface.co/openai-community/roberta-base-openai-detector
42. fakespot-ai/roberta-base-ai-text-detection-v1 - Hugging Face, accessed on January 5, 2026, https://huggingface.co/fakespot-ai/roberta-base-ai-text-detection-v1
43. Maggieli99/RoBERTa_quantization - GitHub, accessed on January 5, 2026, https://github.com/Maggieli99/RoBERTa_quantization
44. Faster and smaller quantized NLP with Hugging Face and ONNX Runtime | by Yufeng Li | Microsoft Azure | Medium, accessed on January 5, 2026, https://medium.com/microsoftazure/faster-and-smaller-quantized-nlp-with-hugging-face-and-onnx-runtime-ec5525473bb7
45. TikTok Video Ad Specs & Placements Guide for 2025 - QuickFrame, accessed on January 5, 2026, https://quickframe.mountain.com/blog/tiktok-video-ad-specs/
46. TikTok Content Posting API Overview, accessed on January 5, 2026, https://developers.tiktok.com/doc/content-posting-api-reference-direct-post?enter_method=left_navigation