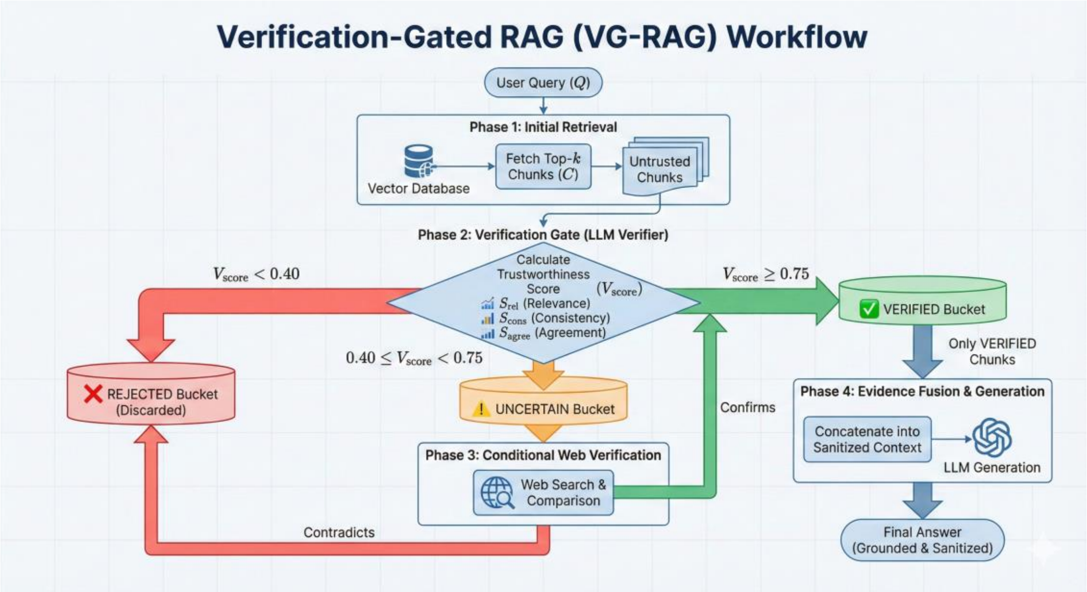
Verification-Gated RAG (VG-RAG)


Verification-Gated RAG (VG-RAG) Workflow
User Query (Q)
Phase 1: Initial Retrieval
Vector Database
Fetch Top-k Chunks (C)
Untrusted Chunks
Phase 2: Verification Gate (LLM Verifier)
Calculate Trustworthiness Score (V_{score})
S_{rel} (Relevance)
S_{cons} (Consistency)
S_{agree} (Agreement)
$V_{\text{score}} < 0.40$
$V_{\text{score}} \geq 0.75$
$0.40 \leq V_{\text{score}} < 0.75$
✗ REJECTED Bucket (Discarded)
✓ VERIFIED Bucket
Only VERIFIED Chunks
⚠ UNCERTAIN Bucket
Phase 3: Conditional Web Verification
Web Search & Comparison
Confirms
Contradicts
Phase 4: Evidence Fusion & Generation
Concatenate into Sanitized Context
LLM Generation
Final Answer (Grounded & Sanitized)

1. Core Philosophy

Standard Retrieval-Augmented Generation (RAG) systems operate on a **"Retrieve-then-Generate"** basis. This approach implicitly assumes that all retrieved documents are accurate, up-to-date, and relevant. In practice, this leads to **"Hallucination Propagation,"** where the Large Language Model (LLM) incorporates outdated, noisy, or conflicting data from the vector database into its final answer.

VG-RAG introduces a **"Retrieve-Verify-Generate"** paradigm. It operates on the principle of **Input Sanitization**: strictly preventing unverified information from ever reaching the generation model. We replace the standard "soft attention" mechanism of LLMs (which attempts to weigh conflicting data implicitly) with a **hard verification gate** that explicitly filters content before generation.

2. System Workflow (Step-by-Step)

Phase 1: Initial Retrieval

- **Input:** User Query (Q).
- **Action:** The system performs dense vector retrieval from the internal Knowledge Base to fetch the top-k chunks (C = {c1, c2, ..., ck}).
- **Status:** At this stage, chunks are **untrusted**.

Phase 2: The Verification Gate (The Novelty)

This is the core innovation. Before generation, every chunk c_i passes through a lightweight **LLM Verifier** that calculates a **Trustworthiness Score (V_{score})** based on three distinct metrics:

1. **Semantic Relevance (S_{rel}):** Does this chunk actually answer the user's specific question?
2. **Internal Consistency (S_{cons}):** Is the chunk factually self-contained, clear, and unambiguous?
3. **Cross-Chunk Agreement (S_{agree}):** Does this chunk align with the majority of other retrieved chunks? (This detects outliers and contradictions).

Based on the final V_{score}, chunks are sorted into three strict buckets:

- ✅ **VERIFIED ($V_{score} \geq 0.75$):** Passed directly to the final context.
- ⚠️ **UNCERTAIN ($0.40 \leq V_{score} < 0.75$):** Passed to Phase 3 (Web Verification).
- ❌ **REJECTED ($V_{score} < 0.40$):** Discarded immediately.

Phase 3: Conditional Web Verification (The Safety Net)

This step is **only** triggered for chunks in the **UNCERTAIN** bucket. This makes the system computationally efficient compared to agentic workflows that always search the web.

- **Action:** The system generates a targeted search query based on the uncertain chunk.
- **Comparison:** External web results are compared against the chunk.
  - If Web confirms Chunk ---> Promoted to **VERIFIED**.
  - If Web contradicts Chunk ----> Downgraded to **REJECTED**.

Phase 4: Evidence Fusion & Generation

- **Context Construction:** The system concatenates *only* the **VERIFIED** chunks (from Phase 2 and Phase 3).
- **Generation:** The LLM generates the final answer using this sanitized context.
- **Result:** A response grounded strictly in verified evidence, with conflicting or hallucinated data removed *before* generation.