Think of your system as:

⭐ **A smart detective that investigates a place before writing a description.**

🧠 The Big Idea (One Simple Mental Model)

Imagine you ask:

👉 "Tell me about *Stadhuis Hasselt*."

Your system does NOT immediately ask AI.

Instead it:

1. Searches multiple trusted sources.
2. Checks if those sources agree.
3. Builds a "truth package".
4. Only then asks AI to write nicely.

So:

Internet research FIRST

Writing SECOND

This is VERY important.

🚦 Pipeline Overview (Simple Version)

Your pipeline is basically:

1. Identify the place correctly

2. Collect information

3. Score how trustworthy info is

4. Merge best facts

5. Let AI write summary

6. Fallback if AI fails

Let's go through each step like a story.

⭐ Step 0 — resolveCanonicalEntity()

What problem does this solve?

People call places different names:

- Stadhuis Hasselt

- Hasselt City Hall
- Gemeentehuis Hasselt

Without fixing this:

👉 system thinks they are different places.

What happens here?

You ask Wikidata:

“Is there a known official entity matching this?”

If yes, you get:

- official name
- aliases
- Wikipedia link
- official website
- image
- categories

Think:

👉 Getting the **official ID card** of the place.

Important:

- It runs in background.
- If it fails → no problem.

⭐ Step 1 — gatherSignals()

This is your **research phase**.

Instead of trusting one source, you collect multiple “signals”.

Each signal = one piece of evidence.

Phase 1 — Cheap sources (always used)

These are free and fast:

1️⃣ Local archive

Your own curated data.

Like your private notes.

2️⃣ Wikipedia

Very reliable.

Often enough alone.

3️⃣ OSM (OpenStreetMap)

Provides:

- tags
- website
- coordinates

4️⃣ DuckDuckGo

Quick summary if available.

Phase 2 — Expensive search (conditional)

Only used if needed.

You call Tavily or Google ONLY IF:

- Wikipedia missing
- No official website
- Trust is too low

So:

👉 Expensive search becomes LAST resort.

⭐ Special Upgrade — Official Website Scraper

If OSM gives:

website = example.com

You don't just store link.

You actually:

- open page safely
- read meta description

- extract real content

Why important?

Official websites = VERY reliable.

⭐ Step 2 — analyzeSignals()

Now your detective evaluates evidence.

Each signal gets a score.

Pass 1 — Basic checks

Examples:

✓ Official website → strong trust

✓ Contains POI name → boost

❌ Generic city text → penalty

❌ junk tag list → penalty

Simple rule-based scoring.

Pass 2 — Graph consensus (THIS IS ADVANCED)

This is like asking:

"Do sources agree with each other?"

You build a graph:

signal A connected to signal B if similar

Agreement factors:

- Both mention POI name
- Share category words
- Text similarity

If many signals agree:

👉 trust increases.

Think:

One witness = maybe

Five witnesses saying same thing = likely true

⭐ Step 2b — mergeSignals()

Now you create a clean package for AI.

Before:

AI receives messy raw text

Now:

AI receives organized facts

Structure:

- best descriptions
- categories
- images
- official website
- short factual snippets

Also:

- removes duplicates
- keeps only high-trust content

This step reduces hallucination a LOT.

⭐ Step 3 — AI Writing (Gemini)

Important:

AI is NOT researching anymore.

It just:

👉 rewrites validated info into readable text.

Two stages:

Short description

Quick summary.

Full details

Longer version.

⭐ Step 4 — resolveConflicts()

If AI fails:

System picks best signal automatically.

So user still gets result.

🧠 Name Normalization (normalizePoiName)

This is your language translator.

Example:

stadhuis → city hall

kerk → church

é → e

remove punctuation

Why?

So different spellings match correctly.

⭐ Trust Score Flow (Simplified)

Each signal starts with base trust.

Example:

Wikipedia = 0.95

DDG = 0.6

Then:

heuristics adjust score

+

agreement bonus from graph

=

final trust score

High-trust signals are used more.

🧠 Backend APIs (Simple View)

These are just tools your detective can use:

- Gemini → writing
- Tavily/Google → search
- DDG → quick info
- scrape-meta → read website safely

- cloud-cache → remember results

⭐ SUPER SIMPLE SUMMARY

Your system behaves like:

1. Identify the place correctly

2. Ask several reliable sources

3. Check which sources agree

4. Combine best facts

5. Ask AI to write nicely

🔥 **Honest assessment (very important)**

This architecture is already:

👉 semi-professional geo-intelligence level.

Most apps:

Search → AI summarize

Your system:

Multi-source validation → structured synthesis

Which is MUCH better.