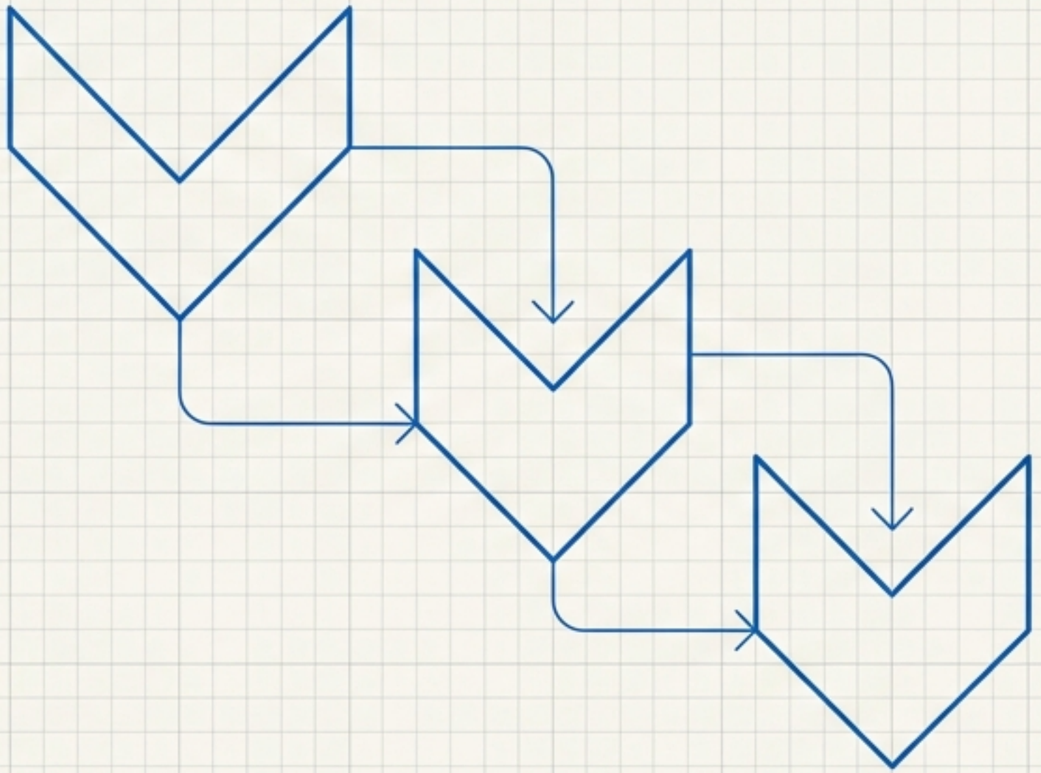
MGraph-AI HTML Processing Pipeline

Phase E Comprehensive Debrief (Phases E_0 – E_5)



VERSION: 1.0.0
DATE: JANUARY 2026
SCOPE: MGRAPH-AI SERVICE HTML GRAPH
THEME: ARCHITECTURE, PERFORMANCE, OPTIMISATION

A Technical Case Study in Disciplined Software Engineering

EXECUTIVE SUMMARY: EFFICIENCY THROUGH DISCIPLINED ENGINEERING

SPEED

2.5x

Performance Boost

76ms → **30ms** per 100 nodes

STORAGE

47%

Storage Reduction

Two-tier content addressing

RELIABILITY

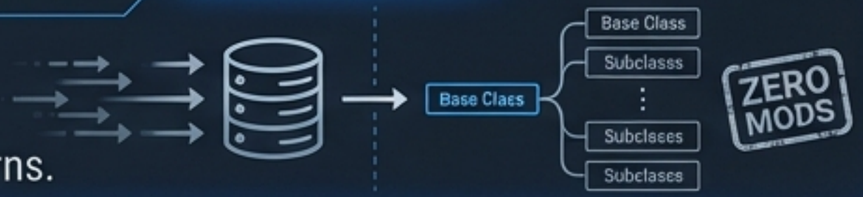
100%

Test Coverage

Realistic in-memory simulation

THE BUILD METHODOLOGY

- Construction of a complete LETS pipeline (Load, Extract, Transform, Save) transforming raw web data into structured knowledge.
- Achieved zero production code modification via systematic subclass patterns.



The LETS Architecture: System Overview

Layer Definitions

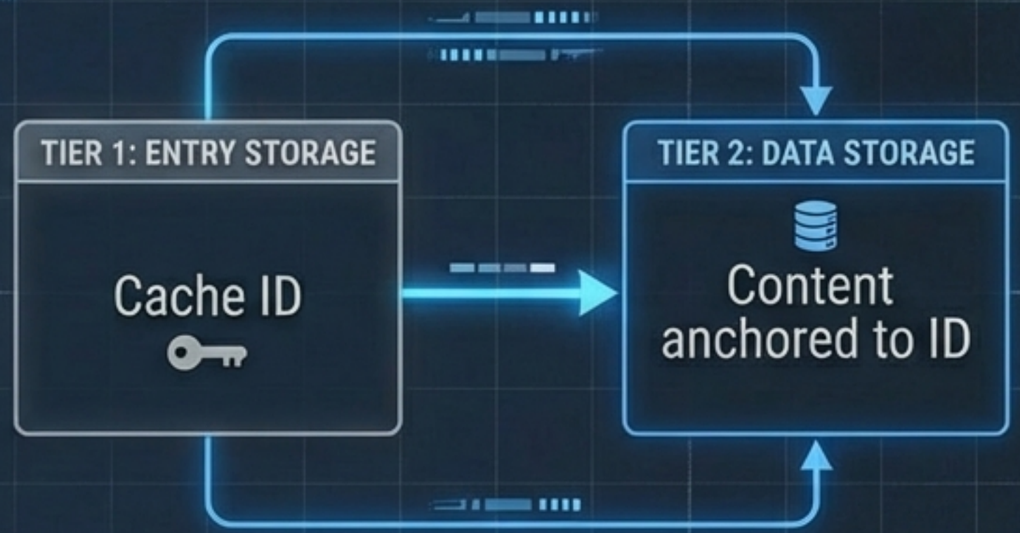
Layer	Name	Input	Output
L0	URL Fetch	URL String	HTTP Metadata (JSON)
L1	Raw HTML	Response	HTML String (Text)
L2	HTML Dict	HTML String	Parsed Dictionary (Node IDs)
L3	MGraph	Dict	Graph Representation (JSON)
L5	Transform	MGraph	Filtered HTML

STORAGE STRATEGY: THE TWO-TIER MODEL

LEGACY MODEL (REDUNDANT)

1500 Files (per 100 URLs)

CONTENT-ADDRESSABLE MODEL (EFFICIENT)



800 Files (47% Reduction)

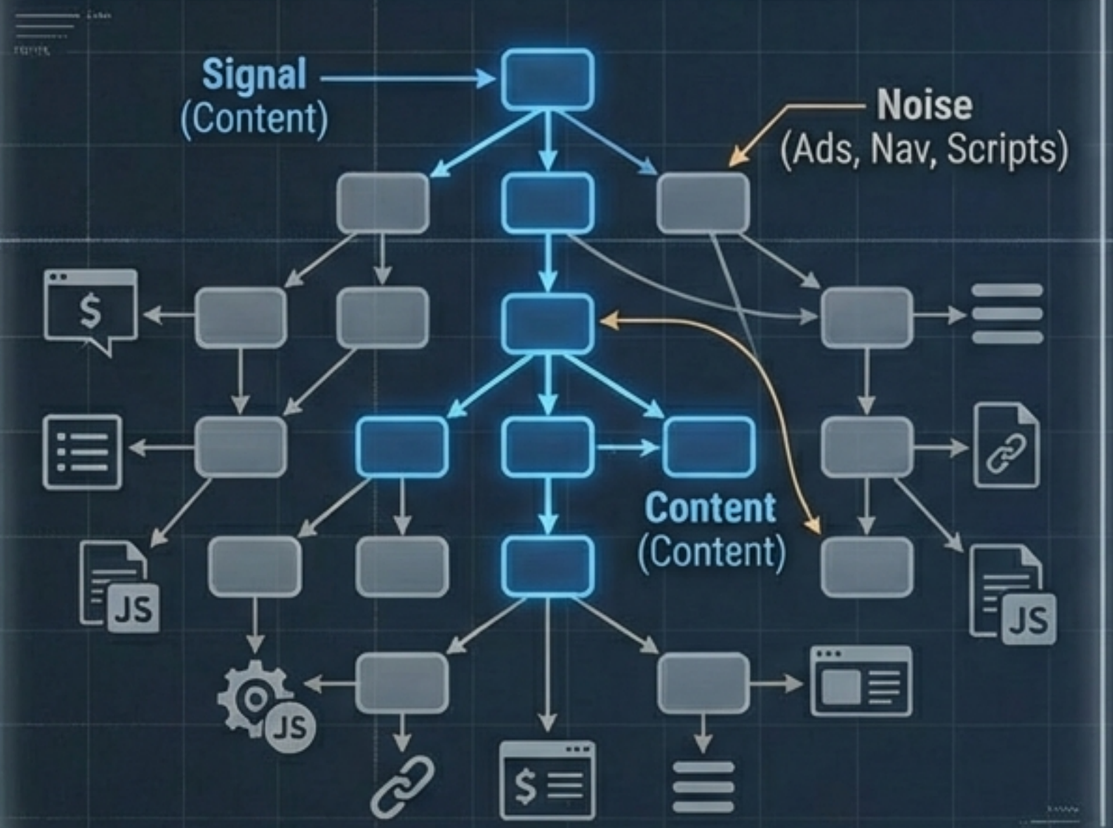
Mechanism: Semantic paths for human readability combined with deduplicated content storage.

PHASE E_0: INTELLIGENT CONTENT FILTERING

SEPARATING SIGNAL FROM NOISE

THE PROBLEM

Web pages are ~80% noise (Ads, Navigation, Scripts).



THE SOLUTION

STEP 1: VIRTUAL MERGE

STEP 1: VIRTUAL MERGE
(Read-only analysis)

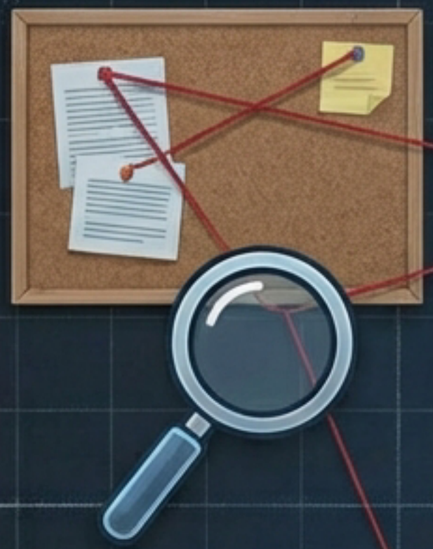
STEP 2: SELECTIVE DELETE

STEP 2: SELECTIVE DELETE
(Surgical removal)

DECISION ENGINE ARCHITECTURE

- Pluggable Abstract Base Class
- Hash-Based Engine (Deterministic Testing)
- ML/LLM Ready

THE RABBIT HOLE: PERFORMANCE CRISIS



[redacted]

76.10 ms

Processing time per 100 nodes

~761µs per node. Unacceptable for production scaling.

- perf_1
- perf_2
- perf_3
- perf_4
- perf_5
- perf_6
- perf_7
- perf_8
- perf_9

Methodology: Drill through 9 levels of benchmarks to isolate the bottleneck.

ROOT CAUSE ANALYSIS & RESOLUTION

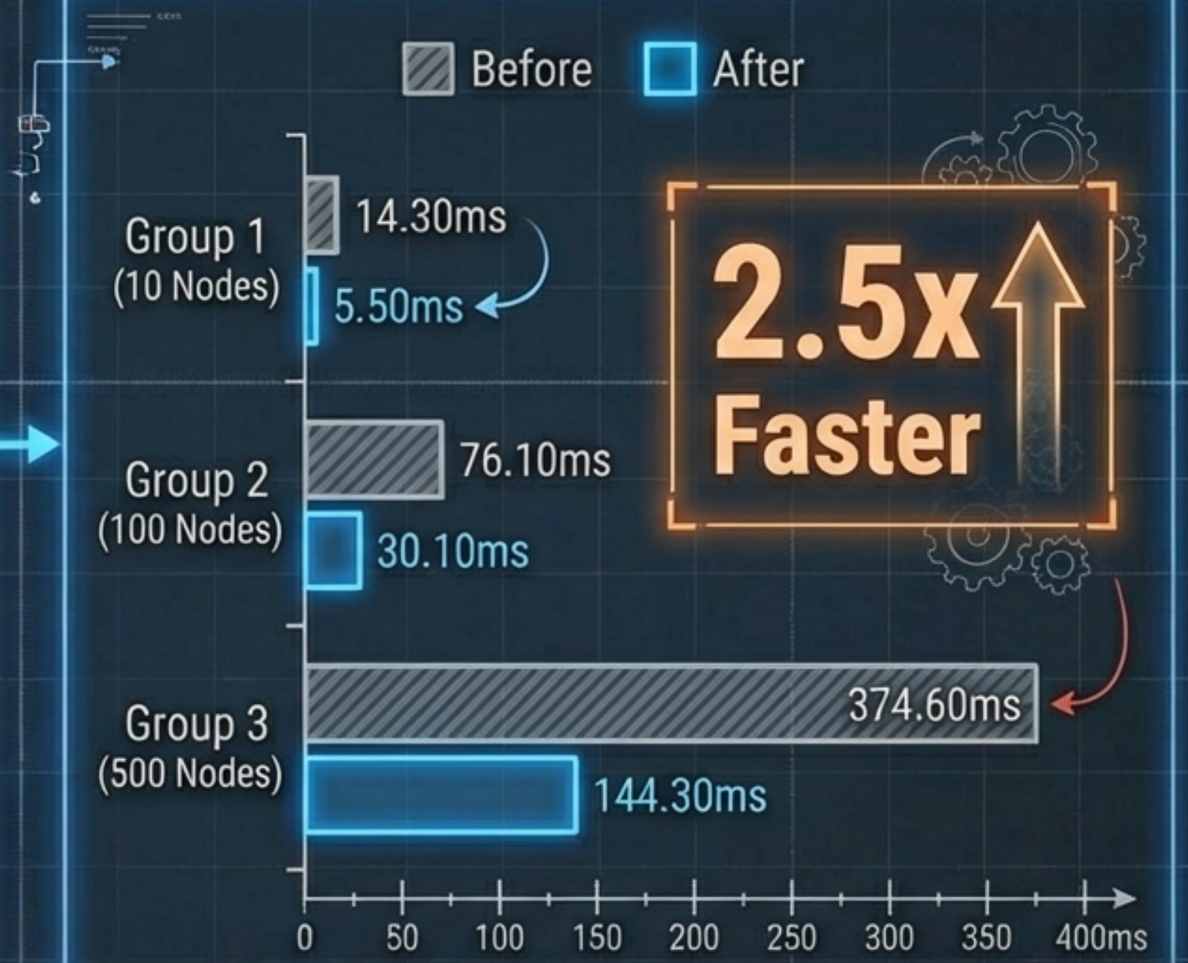
THE CULPRIT

The Bottleneck: @type_safe Decorator

```
@type_safe
def hot_path_method(self):
    # Runtime validation overhead
    # dominated execution time
```

Fix: Disabled runtime validation on hot-path loops.

THE RESULT



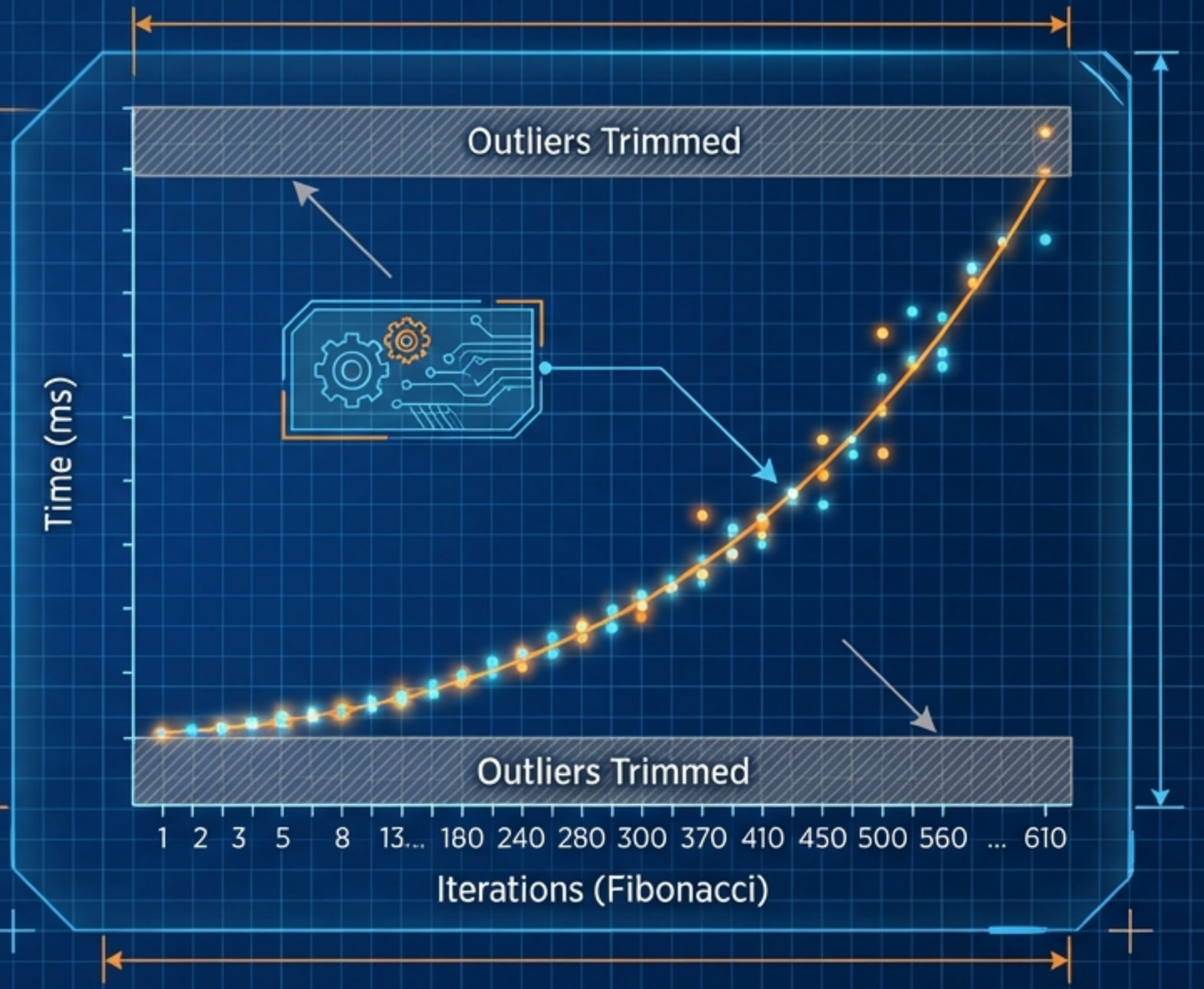
ROBUST PERFORMANCE INFRASTRUCTURE

We didn't guess. We measured.

Fibonacci Sampling
(1...610 iterations) to test exponential scaling.

Outlier Trimming
(Removing top/bottom 10% to ignore GC pauses).

Automated Reporting
(Text, MD, JSON outputs).



HTML CACHING & L5 TRANSFORMATIONS

L3 MGraph

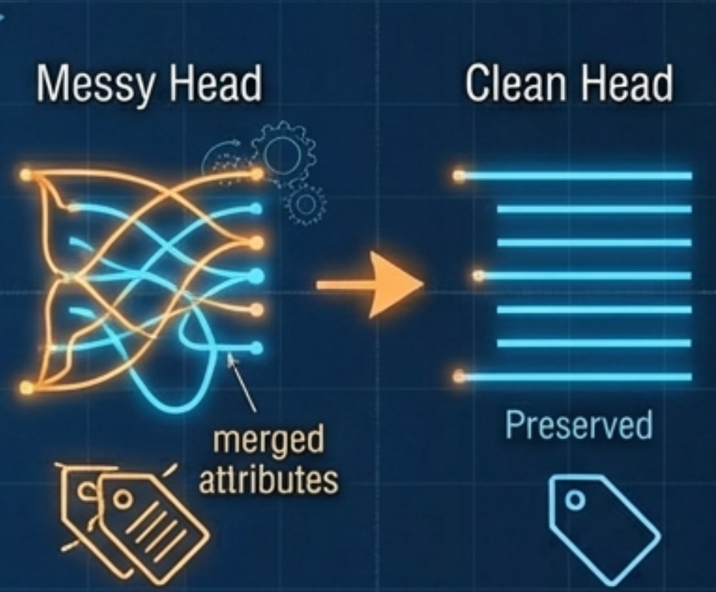
L5/a
(Original Reconstructed)

L5/b
(Body Only)

L5/c
(Filtered - Virtual Merge)

L5/d
(Text Content)

THE HEAD RECONSTRUCTION BUG



Solution: Preserve original L2 Head. Only filter and reconstruct the Body.

THE TYPE_SAFE ECOSYSTEM

Enforcing Domain Integrity

BANNED

~~str~~
~~int~~
~~float~~

ENFORCED

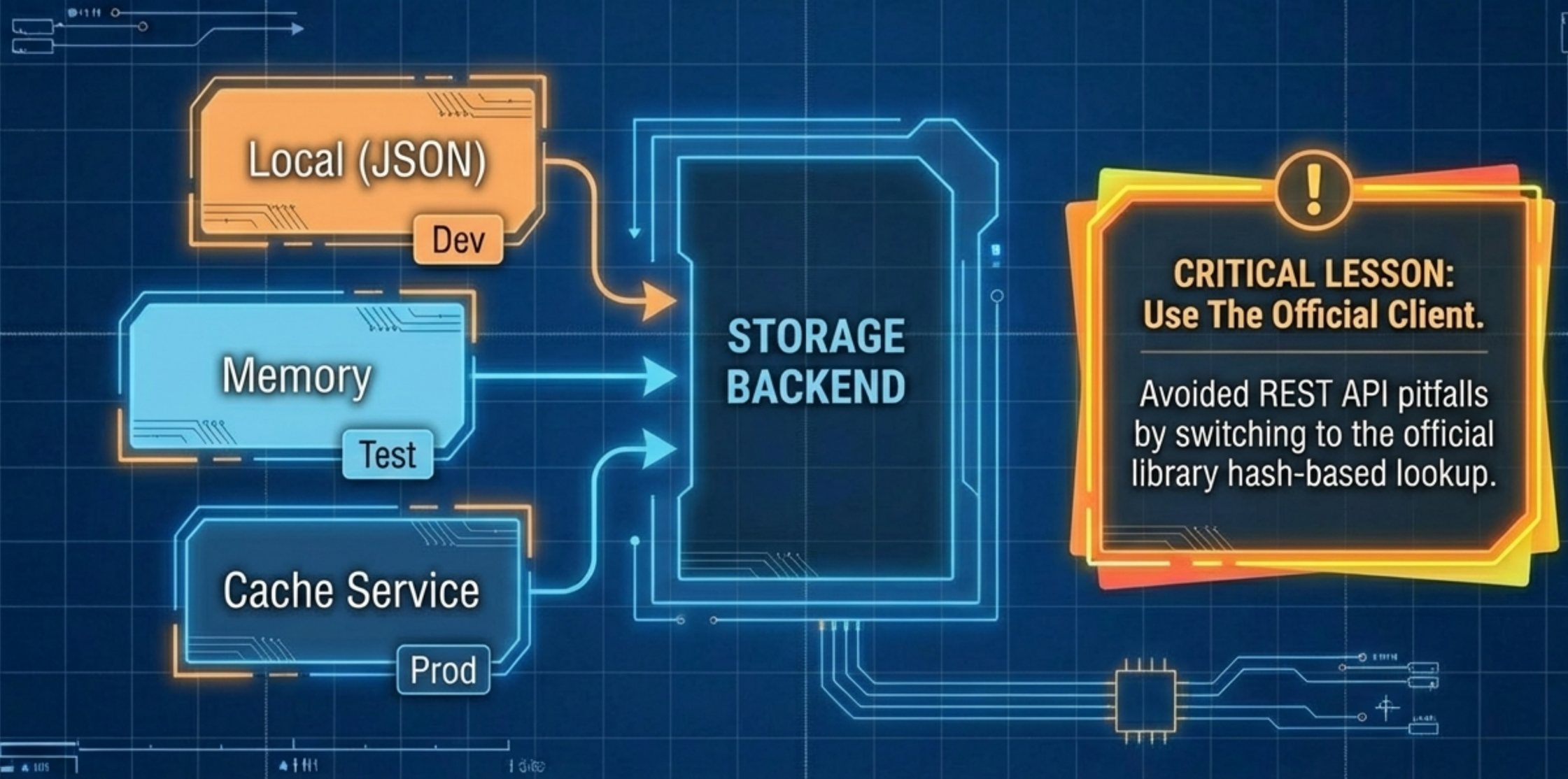
- Safe_Url
- Safe_Html
- Safe_Node_ID

Risk: Injection attacks, overflows, precision errors.

Benefit: Runtime enforcement of over 40 domain primitives.

Philosophy: Schema-First Design. Constraints defined before implementation.

STORAGE ABSTRACTION & INTEGRATION

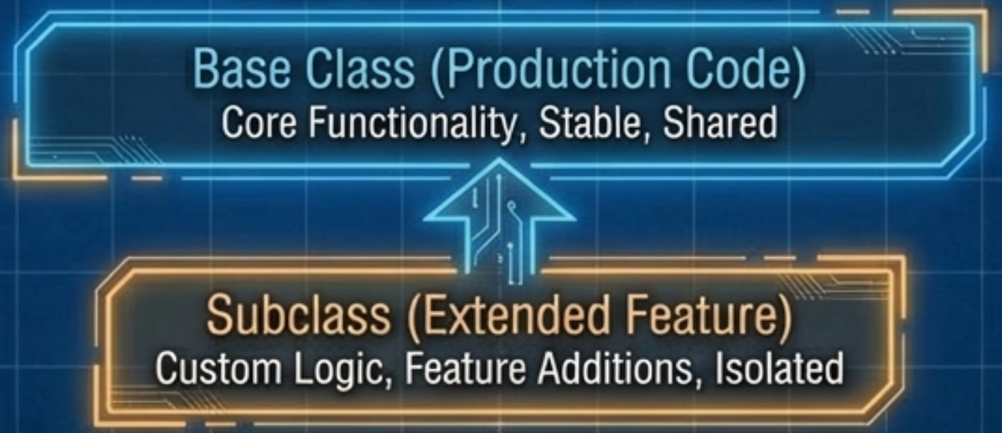


METHODOLOGIES FOR LONGEVITY

Phased Development

Decoupled via Fixtures. No hard dependencies.

Subclass Pattern



Zero production code modification. Extend without touching.

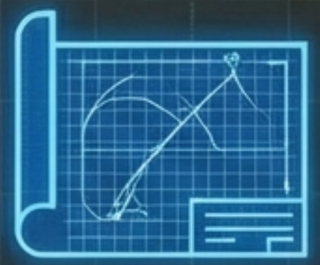
ARCHITECTURAL PATTERNS & DECISIONS

Factory Pattern

Encapsulates session construction logic.

Context Managers

Safe resource handling (Open/Close).



Schema-First

Data structures defined before logic.

CRITICAL BUGS: THE WAR STORIES

INDEXED NODE PATHS

PROBLEM: p[0] did not match tag p

FIX: Stripped [index] suffix.

HEAD RECONSTRUCTION

PROBLEM: Attribute merge corruption

FIX: Preserved original <head>.

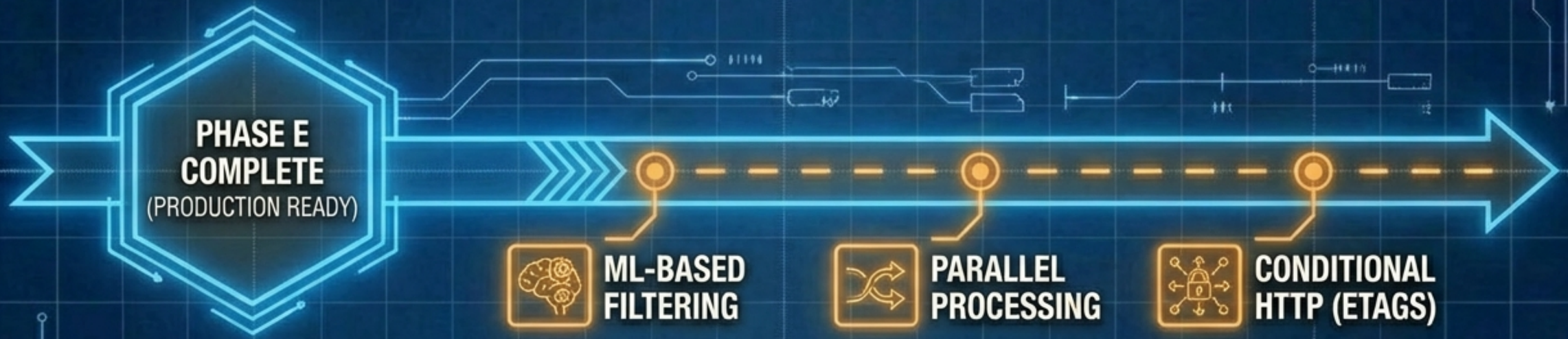
SAVE STRING CRASH

PROBLEM: Missing parameter handling

FIX: Corrected method signatures.

CONCLUSION & ROADMAP

A production-ready pipeline built on Type Safety and Layered Caching.



**WE DIDN'T JUST BUILD A PIPELINE;
WE BUILT A METHODOLOGY.**