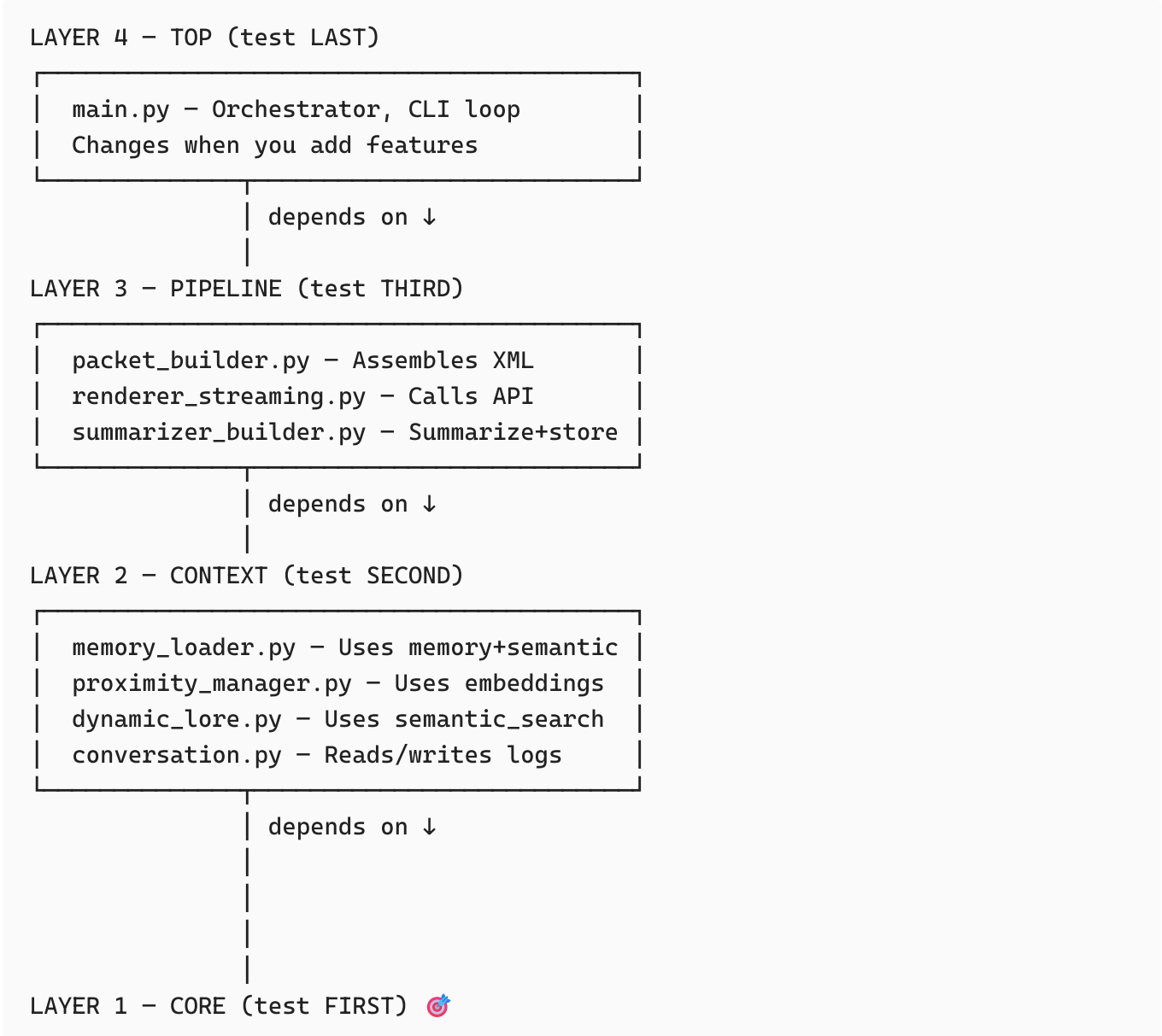
Testing Roadmap — Layer-by-Layer

> Bottom-up approach: test the foundation first, then build upward.
> Derived from HLD.md Section 12 (File Dependency Graph).
>
> **Low-value skipping:** Testing "simple getters" is a waste of time. That is **Smart**.
> **The "Sweet Spot" chart:** This is a real industry concept. Testing has diminishing returns.
>
> **At last generate HTML** coverage report so you can see which parts of your code are currently "unprotected"

Dependency Layers



```
┌──────────────────────────────────────────┐
│  renderer_base.py — clean/validate/parse │
│  temporal.py — time delta logic          │
│  memory.py — SQLite FTS5 operations      │
│  semantic_search.py — FAISS operations   │
│  model_config.py — constants             │
└──────────────────────────────────────────┘
  ↑ Zero project dependencies. Pure utilities.
```

Rule: If Layer 1 is broken → everything above is broken.
If Layer 1 is tested → you KNOW the foundation works.

Testing Order — Checklist

Layer 1 — Core (start here)

- [ ] **renderer_base.py** — Pure functions, zero deps, easiest win (~6-8 tests)
  - [ ] clean_response() strips `[AI]:` prefixes
  - [ ] clean_response() strips punctuation artifacts
  - [ ] validate() rejects empty/short responses
  - [ ] validate() catches user impersonation
  - [ ] parse_sections() extracts XML tags correctly
  - [ ] parse_sections() handles missing tags
  - [ ] build_gemini_payload() produces correct structure
  - [ ] load_api_key() handles missing file
- [ ] **temporal.py** — Only needs json + `datetime` (~4-5 tests)
  - [ ] load_and_update() calculates minutes/hours/days correctly
  - [ ] Handles missing timestamps.json (first run)
  - [ ] Handles corrupted timestamps.json
  - [ ] get_time_block() formats string correctly
- [ ] **memory.py** — Mock with in-memory SQLite (~5-6 tests)
  - [ ] _init_db() creates FTS5 table
  - [ ] add_episode() stores and returns rowid
  - [ ] search() finds matching memories
  - [ ] search() handles special characters / empty query
  - [ ] wipe_memory() clears everything

Layer 2 — Context (after Layer 1 passes)

- [ ] conversation.py — Mock with temp directory (~4-5 tests)
  - [ ] `log_message()` writes to correct file
  - [ ] `get_recent_history()` returns correct count
  - [ ] `buffer_clear()` clears buffer
  - [ ] `buffer_to_raw_text()` formats correctly
- [ ] `memory_loader.py` — Mock Layer 1 deps (~3 tests)
  - [ ] Intent detection ("do you remember" → triggers search)
  - [ ] No intent → no memory section
  - [ ] Returns formatted `<memory_bank>` block

Layer 3 — Pipeline (after Layer 2 passes)

- [ ] packet_builder.py — Integration test (~3-4 tests)
  - [ ] build() produces valid XML with all sections
  - [ ] Proximity block appears on first turn
  - [ ] Memory block appears only on memory intent
- [ ] main.py — is_valid_response() (~3 tests)
  - [ ] Rejects empty string
  - [ ] Rejects fallback message
  - [ ] Accepts valid response

Layer 4 — End-to-End (after everything passes)

- [ ] **Pipeline integration** — Mock the Gemini API (~2-3 tests)
  - [ ] Full chain: input → build → render → validate → commit
  - [ ] Invalid response → discard (nothing logged)
  - [ ] 5-turn cycle triggers summarizer

Total: ~25-30 tests → ~80% critical path coverage

```
Layer 1:  ~15 tests   (foundation)
Layer 2:  ~7 tests    (context glue)
Layer 3:  ~6 tests    (assembly)
Layer 4:  ~3 tests    (full pipeline)
──────────────────────
Total:    ~31 tests
```