# **Pre-Coding Essentials (Component: BUILD\_RESULT, Version/FormulaID: VM-ENGINE v0)**

## **1) Goal & Success**

Goal: Compose the **Result** artifact from prior pipeline outputs (allocations, aggregates, gates, ties, label), ready for canonical serialization and hashing.

Success: Fields match DB spec (top-level IDs; per-unit blocks; gates; TieLog; Label; optional frontier\_map\_id); deterministic ordering; integer/rational values copied without re-computation.

## **2) Scope**

In scope: Merge **LegitimacyReport**, **DecisivenessLabel**, **TieLog**, **AggregateResults**, **UnitScores/UnitAllocation** into **Result**; attach input IDs (REG/TLY/PS) and optional frontier\_map\_id.

Out of scope: I/O & hashing (vm\_io), RunRecord persistence (next step).

## **3) Inputs → Outputs (with schemas/IDs)**

Inputs:

reg\_id, ballot\_tally\_id, parameter\_set\_id (from ctx).

AggregateResults (totals/shares/turnout/weighting).

LegitimacyReport (gate values & pass/fail).

DecisivenessLabel.

TieLog (may be empty).

FrontierMap pointer if produced.

Output:

**Result (RES:…)** with: top-level input IDs; per-unit blocks (tabulation, allocation, turnout, flags); aggregates; gates; TieLog; Label; optional frontier\_map\_id.

## **4) Entities/Tables (minimal)**

## **5) Variables (only ones used here)**

None computed here. Values already decided upstream; we **copy** gate results, label, and flags. (Gate semantics reference VM-VAR-020/022/023/025/026/027, but not recalculated here.)

## **6) Functions (signatures only)**

fn build\_result(ctx: &PipelineCtx, agg: &AggregateResults, gates: &LegitimacyReport, label: &DecisivenessLabel, ties: &[TieEvent], frontier\_id: Option<FrontierId>) -> Result<Result>  
 Purpose: Compose Result. Ordering/determinism enforced.

Helpers:

fn write\_unit\_blocks(..) -> Vec<UnitBlock> (IDs/order canonical).

fn attach\_gate\_panels(result: &mut Result, gates: &LegitimacyReport)

fn attach\_ties\_and\_label(result: &mut Result, ties: &[TieEvent], label: &DecisivenessLabel)

## **7) Algorithm Outline (bullet steps)**

Initialize Result with input IDs (REG/TLY/PS).

Emit **UnitBlocks** from unit-level tabulation/alloc data; set flags (unit\_data\_ok, unit\_quorum\_met, unit\_pr\_threshold\_met, protected\_override\_used, mediation\_flagged).

Attach **Aggregates** and **weighting**.

Attach **gates** (values + pass/fail + denominators).

Append **TieLog**; append **Label**.

If present, set frontier\_map\_id.

Return Result (ready for canonical JSON + hashing downstream).

## **8) State Flow (very short)**

Previous steps: LABEL\_DECISIVENESS → **BUILD\_RESULT** → BUILD\_RUN\_RECORD.

Stop/continue: Always build a Result; if VALIDATE failed upstream, label is **Invalid** and gates/frontier may be absent. (Packaging still occurs.)

## **9) Determinism & Numeric Rules**

Ordering: Units by **Unit ID**; Options by **Option.order\_index** then ID.

Values: Use exact integers; ratios copied from gates; no float recomputation.

TieLog: policy/seed/order recorded verbatim.

## **10) Edge Cases & Failure Policy**

Missing per-unit tallies for a unit: emit unit\_data\_ok=false; keep allocations/gates if present; totals may be zeroed. (Still a valid Result.) Spec allows packaging even after validation fail.

No FrontierMap produced: frontier\_map\_id omitted. Reports read only available artifacts.

Ensure seats sum to Unit.magnitude (PR) or 100% power (WTA).

## **11) Test Checklist (must pass)**

Top-level IDs set; unit blocks present and ordered; aggregates match inputs; gates copied with denominators; TieLog & Label preserved; optional frontier pointer set. (Report depends on these fields.)