Below are precise, P1-style instructions for preparing **config/thresholds.yml**. This file defines how the evaluator converts raw metrics into a single pass/fail decision with reasons and penalties.

Purpose

Standardize evaluation so pieces are accepted or rejected **consistently**. The thresholds file: - declares metric **minimums** and **target ranges**, - assigns **weights** for a weighted score, - encodes **penalties** and **critical violations**, - sets an **acceptance rule** used by `/evaluate` and CI checks.

What to Prepare (Creator-Curated)

1) **Metric floors** (min acceptable values 0..1): frame fit, schema coverage, metaphor diversity, attention discipline, explosion timing. 2) **Target ranges** where applicable (e.g., explosion timing ideal window). 3) **Hard caps** for form (max lines/chars) and trace requirements. 4) **Weights** (0..1, sum $\leq$ 1) that build a weighted score. 5) **Penalties** (0..1) for violations (e.g., banned pairs, frame mismatch, over length). 6) **Critical violations** that auto-fail regardless of score. 7) Final **acceptance rule** (boolean expression) combining weighted score, floors, and criticals.

Validation Rules

- Header includes `version`.
- All metric names must match evaluator outputs.
- Numeric values in `[0,1]` except line/char caps.
- `weights` keys must be a subset of metrics; recommended `sum(weights)=1.0` (not required).
- `acceptance_rule` must reference defined keys only.

Minimal YAML Template

```yaml
version: "0.1.0"

metrics:
  # Minimum floors for individual metrics (0..1)
  frame_fit_min: 0.70
  schema_cov_min: 0.60
  metaphor_diversity_min: 0.40
  attention_discipline_min: 0.65
```

```yaml
  # Ranges (inclusive) for time-critical events
  explosion_timing_range: [0.75, 0.95]
# desired expectation value at the TURN beat

form:
  max_lines: 8
  max_chars: 700
  trace_required: true

weights:                          # contribution to weighted score (sum ≈ 1.0)
  frame_fit: 0.35
  schema_cov: 0.25
  metaphor_diversity: 0.15
  attention_discipline: 0.15
  explosion_timing: 0.10

penalties:                        # deducted AFTER weighted sum (clamped to
[0,1])
  frame_violation: 0.25           # using schemas/metaphors disallowed by frame
  banned_pair: 0.30               # disallowed metaphor/schema pair fired
  over_length: 0.15               # exceeds max_lines or max_chars
  missing_trace: 0.40             # no trace attached when required
  weak_turn: 0.10                 # turn beat misses min expectation by small
margin

critical_violations:              # any true → automatic FAIL
  disallowed_content: true        # safety/policy hit flagged by guardrails
  explosion_outside_turn: true    # explosion fires outside configured TURN beat
(unless overridden)
  empty_output: true              # produces zero non-whitespace characters

# Optional per-frame/metaphor overrides
overrides:
  frames:
    journey:
      weights: { frame_fit: 0.40, schema_cov: 0.25, metaphor_diversity: 0.10,
attention_discipline: 0.15, explosion_timing: 0.10 }
      penalties: { over_length: 0.10 }
  metaphors:
    raw_cooked:
      penalties: { weak_turn: 0.15 }

# Final decision rule (evaluated by the evaluator)
# Available symbols:
#   score = weighted_sum(weights × metrics_realized)
#   floors_ok = all(metric >= *_min)
#   in_range(x, [a,b]) returns boolean
```

```
#   total_penalty = sum(applied penalties, capped at 0.9)
#   pass = (score - total_penalty) >= accept_threshold AND floors_ok AND NOT
any(critical)
accept_threshold: 0.70
acceptance_rule: "(score - total_penalty) >= accept_threshold AND floors_ok AND
in_range(explosion_timing, explosion_timing_range) AND NOT any_critical"

# Optional reporting bins for dashboards
reporting:
  buckets:
    score: [0.0, 0.5, 0.7, 0.85, 1.0]
    frame_fit: [0.0, 0.6, 0.75, 0.9, 1.0]
```

Spreadsheet (Optional) → YAML Columns

If curating thresholds in a sheet, use one row with these columns (keys on the left; values on the right):

```
frame_fit_min,schema_cov_min,metaphor_diversity_min,attention_discipline_min,explosion_timing_low
```

- JSON columns hold objects, e.g., `{"frame_fit":0.35,"schema_cov":0.25,...}`.

Curation Workflow

1) Start conservative: higher floors, modest weights for diversity/timing. 2) Run evaluator on 20–30 samples; adjust floors/weights until **precision** (few false passes) is acceptable. 3) Tighten penalties slowly; keep `critical_violations` rare but meaningful. 4) Freeze version and commit: `config(thresholds): seed v0.1`.

Quality Checklist (Before Handoff)

- [ ] All numbers within valid ranges; weights sum ≈ 1.0.
- [ ] Floors and ranges reflect your taste; explosion window aligns with beats.
- [ ] Acceptance rule references defined keys only and reads clearly.
- [ ] Overrides are minimal and justified; cross-refs exist (frames/metaphors).
- [ ] Version set and commit message prepared.

Notes to Engineering

- Validator must check numeric ranges, weight sum, key existence, and expression safety for `acceptance_rule`.
- Evaluator computes: `score = Σ(weights[k] * metrics[k])`, clamps to `[0,1]`, subtracts `total_penalty`, then applies the rule.
- API `/evaluate` should return: raw metrics, weights, penalties applied, final `score_pre_penalty`, `score_final`, and boolean `pass` with `reasons`.