KPI Engine – Method, Architecture and Implementation Mapping (MVP)

1. Conceptual Overview

The KPI Engine represents the core analytical component of the project performance monitoring system. Its objective is to transform persisted operational data into quantitative performance indicators, classify them using predefined thresholds, and persist historical results for auditability and trend analysis.

The MVP implementation follows a deterministic and reproducible approach. All calculations are executed server-side, without client-side influence, ensuring consistency, traceability, and alignment with project management theory (Earned Value Management principles).

2. KPI Engine Execution Flow

The execution flow of the KPI engine can be described as a pipeline consisting of five sequential stages:

- 1. Data Retrieval – Extraction of baseline, progress and cost data from the database.
- 2. Core Variable Computation – Calculation of PV, EV and AC.
- 3. KPI Formula Evaluation – Derivation of CPI, SPI and Burn Rate.
- 4. RAG Classification – Mapping numeric KPI values to qualitative status.
- 5. Snapshot Persistence – Storage of computed results for historical tracking.

Simplified Component Flow Diagram:

```
Database → Data Services → KPI Engine (PV/EV/AC → KPI Formulas → RAG) → Snapshot Writer → KP
```

3. Mathematical Formulation of KPIs (MVP)

Indicator	Formula (MVP)	Description
PV	BAC × elapsedRatio	Linear planned value progression over project duration
EV	BAC × progressRatio	Earned value derived from aggregated work item progress
AC	SUM(CostEntry.amount)	Actual cost accumulated until asOf date
CPI	EV / AC	Cost efficiency index
SPI	EV / PV	Schedule efficiency index
Burn Rate	AC / elapsedDays	Average daily spending rate

Division-by-zero cases (AC=0 or PV=0) result in null KPI values in the MVP implementation. These are classified as NA in the RAG mapping stage.

4. Data Source Mapping (Implementation Alignment)

KPI Variable	Database Table	Field(s) Used
BAC	Project	plannedBudget
Project Duration	Project	startDate, endDate
Progress Ratio	WorkItem	progressPercent (equal weighted average)
Actual Cost	CostEntry	amount (filtered by projectId and asOf)
Thresholds	KPIDefinition	thresholdGreen, thresholdYellow
Historical Storage	KPISnapshot	value, status, ev, pv, ac

This mapping demonstrates full congruence between theoretical KPI formulas and their technical implementation. Each formula component is directly traceable to a specific database field.

5. RAG Classification Logic

KPI values are interpreted using a configurable threshold model defined per project. The following logic is applied:

- If value is null → Status = NA
- If value ≥ thresholdGreen → Status = GREEN
- If value ≥ thresholdYellow → Status = YELLOW
- If value < thresholdYellow → Status = RED

Threshold configuration is stored in KPIDefinition, allowing project-specific sensitivity tuning.

6. Snapshot Strategy and Auditability

Each recalculation creates new snapshot entries (no overwrite strategy). Snapshots include the KPI value, status, computedAt timestamp, and audit inputs (EV, PV, AC). This enables full traceability and recalculation verification.

Chronological ordering is supported via indexed computedAt fields, enabling trend visualization and dashboard time-series analysis.

7. MVP Assumptions and Limitations

- Planned Value is assumed linear across project duration.
- Work items are equally weighted (no effort-based weighting).
- Burn Rate directional interpretation is not differentiated in MVP.
- No automatic scheduler; recalculation is triggered manually via API.
- No advanced forecasting (EAC, ETC) implemented in MVP.

These constraints are explicitly documented to maintain academic transparency and to define future extension directions.