Kushim ML Linking Specification

Document Purpose

This document specifies the machine learning–assisted linking system used by Kushim to infer relationships between artifacts across platforms. It is designed to be directly implementable by ML and backend engineers and explicitly defines how the system evolves from deterministic heuristics to learned semantic linking.

This spec intentionally avoids "black-box AI" in favor of **explainable, feedback-driven intelligence**.

---

1. Design Goals

1. **High Precision First**: Incorrect links destroy trust faster than missing links.
2. **Explainability**: Every link must expose contributing signals.
3. **Progressive Learning**: ML augments—not replaces—deterministic logic.
4. **Human-in-the-Loop**: User behavior is the primary training signal.
5. **Contextual, Not Global Similarity**: Links are scoped by workspace, team, and time.

---

2. Linking Taxonomy

Kushim supports multiple relationship types, each with different confidence requirements:

Relationship Type	Description	Confidence Threshold
Explicit Reference	Direct ID or URL reference	0.95
Strong Contextual	Same work unit, different medium	0.85
Weak Contextual	Peripheral relevance	0.70
Temporal Association	Related by time/actors only	0.60

---

3. Phase 1: Deterministic Linking Engine (Baseline)

3.1 Input

- Two or more KushimStandardRecords (KSRs)

3.2 Feature Signals

Each candidate pair generates a feature vector composed of deterministic signals:

Signal	Description	Weight (Initial)
Explicit ID Match	Jira ID, PR #, ticket ref	0.7
URL Reference	Direct hyperlink	0.6
Shared Branch / Commit	Git metadata	0.5
Actor Overlap	Same author/participants	0.2
Temporal Proximity	<24h window	0.1
Keyword Overlap	TF-IDF cosine	0.1

Weights are configurable per workspace.

3.3 Scoring Function

```
score = Σ (signal_i * weight_i)
link if score ≥ threshold(type)
```

3.4 Output

- Link object with:
- Source artifact
- Target artifact
- Relationship type
- Confidence score
- Signal breakdown (for UI)

---

4. Phase 2: ML-Augmented Semantic Linking

ML is introduced only after deterministic coverage is stable.

4.1 Problem Framing

- Task: Binary classification (link / no link)
- Secondary task: Relationship type prediction

4.2 Candidate Generation (Critical)

To control computational cost and noise, ML operates only on **pre-filtered candidates**: - Same workspace - Overlapping participants OR - Temporal proximity (<7 days)

---

5. Feature Representation (ML Layer)

5.1 Textual Embeddings

Inputs: - Title - Body/content - Comments (summarized)

Models: - Phase 2a: Sentence-BERT–style encoder - Phase 2b: Fine-tuned domain model

Stored in vector store.

5.2 Structural Features

- Platform types (Slack ↔ GitHub)
- Artifact types (PR ↔ Ticket)
- Graph distance
- Prior deterministic score

5.3 Temporal Decay

Similarity score decays exponentially with time distance.

---

6. Model Architecture

6.1 Hybrid Scoring Model

Final confidence score:

```
final_score = α * deterministic_score
            + β * semantic_similarity
            + γ * structural_features
```

Initial values: - α = 0.6 - β = 0.3 - γ = 0.1

---

7. Context Group Formation (Graph Clustering)

7.1 Graph Construction

- Nodes: Artifacts
- Edges: Links (weighted)

7.2 Clustering Algorithm

- Phase 1: Connected components (thresholded)

- Phase 2: Leiden or Louvain community detection

Clusters map directly to Context Groups.

---

8. Feedback & Learning Loop

8.1 Implicit Feedback (Primary)

User Action	Interpretation
Opens group	Reinforce links
Acts within group	Strong reinforcement
Ignores suggestion	Weak penalty
Separates items	Negative label

8.2 Explicit Feedback (Secondary)

- "This doesn’t belong"
- Manual re-linking

---

9. Training Data Strategy

9.1 Cold Start

- Deterministic links treated as weak positives
- High-confidence explicit links as strong positives

9.2 Ongoing Training

- Online learning (preferred)
- Periodic batch retraining

Negative sampling is time- and actor-aware.

---

10. Evaluation Metrics

Offline

- Precision@k
- Recall@k
- ROC-AUC

Online

- User correction rate
- Context group stability
- Action-without-navigation rate

---

11. Explainability Requirements

Every link must expose: - Top contributing signals - Confidence score - Ability to override

Explainability is mandatory for trust.

---

12. Failure Modes & Safeguards

Failure	Mitigation
Over-linking	Conservative thresholds
Concept drift	Time-decayed training
Feedback sparsity	Implicit signals weighting

---

13. Rollout Plan

Stage 1

- Deterministic only
- Logging all features

Stage 2

- Shadow ML scoring (no user impact)

Stage 3

- ML-assisted links above deterministic threshold

---

14. Definition of ML Linking Done

- 85% precision on strong contextual links

- <5% user-rejected links

- Fully explainable link UI

End of ML Linking Specification