Mechanism for Reviewing and Integrating LLM-Collected Facts

Objective: Provide a transparent pipeline where *temporary* facts proposed by the LLM ensemble are:

1. Captured reliably on each request
2. Presented as *pending* suggestions in a dedicated UI panel
3. Persisted in a separate store until the user explicitly chooses to **Integrate** (learn) or **Dismiss** them

---

1. Data Model & Backend Storage

- **Suggestion Entity**:

```
{
  "id": "uuid-v4",
  "fact": "Predicate(args)",
  "source": "RAG|Ensemble",
  "confidence": 0.87,
  "timestamp": "2025-07-13T12:34:56Z",
  "status": "pending"  // pending | learned | dismissed
}
```

- **Separate Store**: Maintain `learning_suggestions` as a persistent in-memory list or lightweight database table (e.g. SQLite), not cleared on `learn` calls. Only update the `status` field.

---

2. API Endpoints

1. **GET /api/learning_suggestions**
2. Returns all suggestions with `status = "pending"`.
3. **POST /api/learning_suggestions/{id}/learn**
4. Marks suggestion `id` as `status = "learned"`
5. Invokes `learn_fact(fact)` to insert into permanentKnowledge
6. **POST /api/learning_suggestions/{id}/dismiss**
7. Marks suggestion `id` as `status = "dismissed"`
8. **GET /api/learning_suggestions/history**
9. Optional: returns all suggestions with their current status for audit

3. Frontend UI

1. **Learning Suggestions Panel** (in Knowledge Base column)
2. Table columns: Checkbox | Fact | Source | Confidence | Timestamp | Actions (Learn / Dismiss)
3. Data source: `GET /api/learning_suggestions`
4. Each row shows **pending** suggestions only.
5. **Action Buttons**:
6. **Learn**: triggers `POST /api/learning_suggestions/{id}/learn` and visually moves the row to a "Learned" section (or removes it).
7. **Dismiss**: triggers `POST /api/learning_suggestions/{id}/dismiss` and removes the suggestion from the pending list.
8. **Persistent View**:
9. Panel keeps suggestions until user explicitly acts.
10. Optionally, add a secondary tab or accordion "History" to review past suggestions and their statuses.
11. **Live Updates**:
12. Use WebSocket or polling (`/api/learning_suggestions`) every 5s to refresh new pending suggestions.

---

4. Data Flow

1. **Capture Phase**:
2. After each RAG/Ensemble operation, call `extract_temporary_facts()`, generate suggestion entities (only those not already in permanentKnowledge or previous suggestions).
3. Insert new entities into `learning_suggestions` with `status = "pending"`.
4. **Review Phase**:
5. User opens panel, reviews facts, clicks Learn or Dismiss per row.
6. **Integration Phase**:
7. **Learn**: Fact is appended to permanentKnowledge; suggestion status → `learned`.

8. **Dismiss**: suggestion status → `dismissed`, no change to permanentKnowledge.

---

5. Implementation Checklist

-

Outcome: Users will see *all* LLM-proposed facts, decide which to integrate, and have a permanent audit trail of accepted and dismissed suggestions, ensuring full transparency and scientific control over the KB enrichment process.