# ☑️ Mantis-to-Game SDK Integration: Summary & Action Plan

## 🎯 Goal

Use the Mantis SDK to power interactive experiences (games, simulations, workshops) by transforming **document collections** into **semantic landscapes** that include:

- XY coordinates of ideas
- Cluster names
- Hierarchical questions
- Tree-based structure of those questions

## 🧠 Use Cases

The input will define the **context** for framing the questions and map structure. Supported contexts include:

- 👫 **Team building**
- 🎓 **Student learning**
- ❤️‍🩹 **Relationship strengthening**
- 💌 **Romantic date enrichment**

These contexts guide how Mantis interprets clusters and crafts progression paths (mountains, quests, conversations).

---

## 🔄 System Flow

### 1. Input

- Document collection (PDFs, text files, CSVs, etc.)
- Context type (e.g., `"team-building"`)

### 2. Mantis SDK Processing

- Embedding and dimensionality reduction
- Clustering into semantic groups
- Generation of:
  - XY coordinates
  - Cluster name
  - **Tree of questions** for each cluster (root = general, leaves = specific)

### 3. Output

- One CSV or JSON per cluster, each including:

```
{
  "x": 12.5,
  "y": -3.7,
  "cluster": "Team Communication",
  "tree": {
    "question": "How do we collaborate?",
    "children": [
      {
        "question": "What tools do we use to stay in sync?",
        "children": [
          { "question": "Are those tools effective?" },
          { "question": "How often do we use them?" }
        ]
      },
      {
        "question": "Where does miscommunication tend to happen?",
        "children": []
      }
    ]
  }
}
```

**Document for Game-Building Team: API Usage Spec – Mantis SDK Semantic Map + Question Trees**

## Overview

You will use the Mantis SDK to generate **semantic maps** from document collections. These maps include spatial positions (XY), cluster names, and a tree of context-specific questions per cluster. These are designed to drive interactive environments like games or simulations.

---

## How to Use the SDK

Use the function:

```
mantis.export_landscape_with_questions(input_documents, context="team-building")
```

---

## Input Parameters

- `input_documents`: List of file paths (PDFs, text files, etc.)
- `context`: One of the following scenarios:
  - "team-building"
  - "student-learning"
  - "relationship-strengthening"
  - "romantic-date-enrichment"

The `context` determines how Mantis generates questions.

---

## Output (JSON recommended)

Each cluster has:

- `x`, `y`: spatial coordinates (for layout/visualization)
- `cluster`: name or label
- `tree`: a hierarchy of questions (for narrative flow or mission design)

**Example**

```
{
  "x": 15.2,
  "y": -7.3,
  "cluster": "Collaboration",
  "tree": {
    "question": "How do we collaborate?",
    "children": [
      {
        "question": "What tools do we use to stay in sync?",
        "children": [
          {"question": "Are they effective?"},
          {"question": "How often do we use them?"}
        ]
      },
      {
        "question": "Where do communication breakdowns occur?",
        "children": []
      }
    ]
  }
}
```

---

## Integration Guidance

- Use `x`/`y` for visual layout of clusters/points
- Use `cluster` as a theme label
- Use `tree` to construct nested decision trees or dialogue trees in your game UI

---

## Coming Soon

- Alternative output: CSV (with flattened tree)
- Optional: API agent wrappers for direct HTTP call (if needed)

---

## Questions?

Ask Luca (SDK), or reply to this document for game-specific implementation guidance.

**Document for Luca: Mantis SDK API Specification – Semantic Landscape Export with Questions**

## Objective

Implement a new Mantis SDK function to support external applications (e.g., games, simulations, educational tools) that need semantic maps, clusters, and question trees derived from documents and guided by user context.

---

## Function Signature

```
mantis.export_landscape_with_questions(
    input_documents: List[str],
    context: str,
    output_format: Literal["json", "csv"] = "json"
) -> Union[List[Dict], str]
```

## Parameters

- `input_documents`: List of paths to documents (PDF, TXT, CSV, etc.)
- `context`: One of the supported scenario types:
  - "team-building"
  - "student-learning"
  - "relationship-strengthening"
  - "romantic-date-enrichment"
- `output_format`: Either "json" (default) or "csv"

---

## Output Structure (JSON example)

```
[
  {
    "x": 12.5,
    "y": -3.7,
    "cluster": "Team Communication",
    "tree": {
      "question": "How do we collaborate?",
      "children": [
        {
          "question": "What tools do we use to stay in sync?",
          "children": [
            { "question": "Are those tools effective?" },
            { "question": "How often do we use them?" }
          ]
        },
        {
          "question": "Where does miscommunication tend to happen?",
          "children": []
        }
      ]
    }
  }
]
```

---

## Notes

- **Contours are not required** in this version.
- **Question generation is mandatory**, structured as a tree per cluster.
- The output is to be used for interactive UI generation (e.g., game landscapes).
- Ensure compatibility with existing Mantis space creation pipeline.
- Prefer integration as a callable method from the SDK, but could be an extension of `createSpace()` if preferred.

---

## Deliverables

- SDK method implementation
- README usage example
- Unit test with fixture document input and mocked tree output