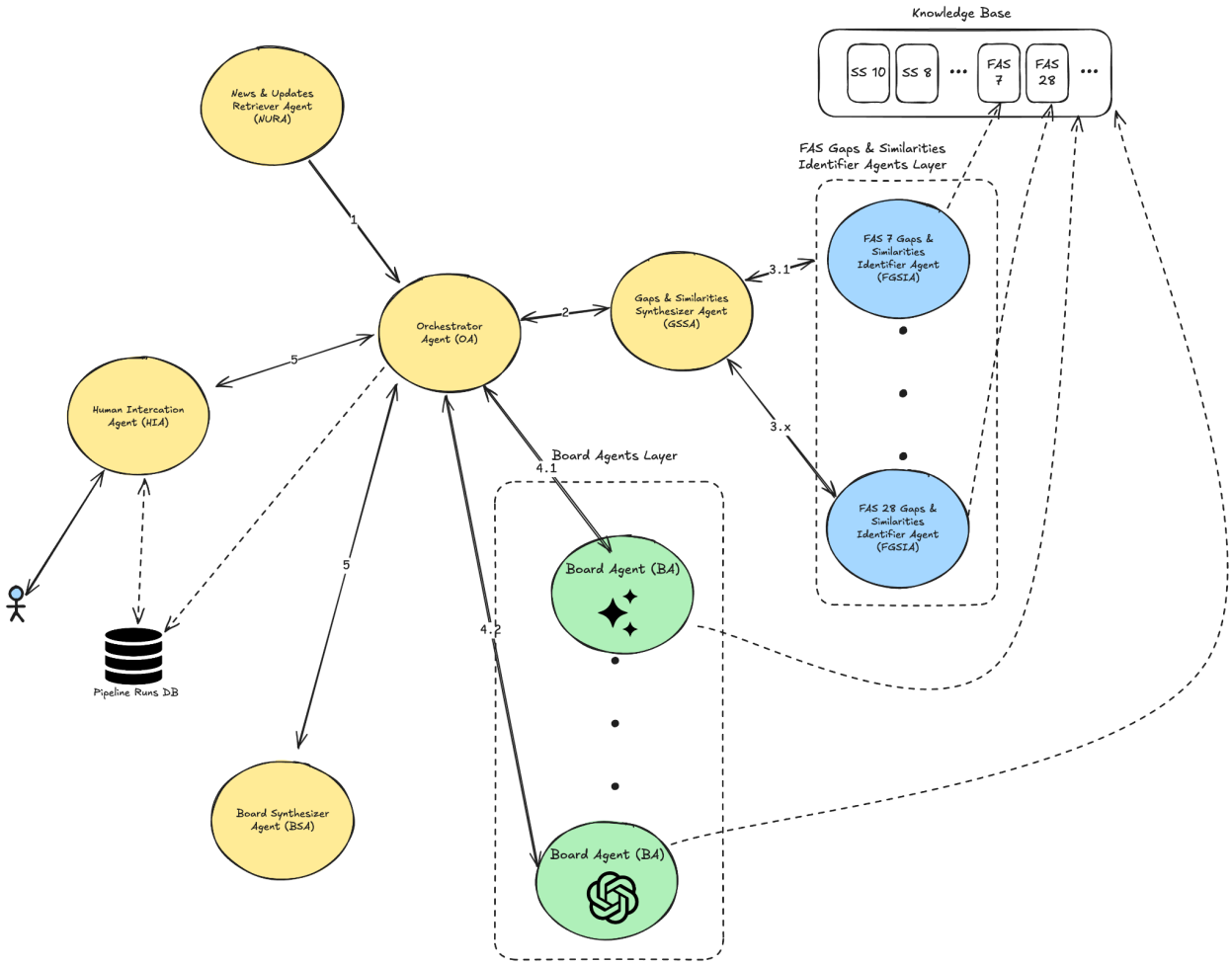
Multi Agents Systems For Standards Enhancement And Improvement

Multi Agent System Diagram

Our system architecture for this challenge was designed with very important criterias in mind:

- Having **multiple expert agents** with a **clear separation of concerns** between each agent and another.
- **Divide and conquer** to optimize the accuracy and coverage of the final output.
- **Scalability and modifiability** that makes the system integrates easily with any newly added standards and principles.
- **Human in the loop** for extra guidance or to handle any inaccuracies

The diagram below gives an overview of the system architecture.



The sequence is as follows:

1. **NURA (News & Updates Retriever Agent)** collects the context from various sources (mainly news sources), and then gives it to the **OA (Orchestrator Agent)**
2. The **OA** gives the context to the **GSSA (Gaps & Similarities Synthesizer Agent)** will will forward it to the **FGSIA Layer (FAS Gaps & Similarities Identifier Agents Layer)**
3. The **GSSA** will receive multiple responses from the **FGSIA Layer** which he will synthesize to get a final verdict of what FASs to update, with justifications, references, and chaine of thoughts.
4. The **OA** will receive the synthesized reply of the **GSSA** and then, for each single FAS, it will give it to the **BA Layer (Bord Agents Layer)**, these **Expert Agents** will come out with various solutions to enhance the given selected FAS, either by updating it or providing a fully new one.

5. The result of **BA Layer** will be given to the **BSA (Board Synthesizer Agent)** that will output its results to the **Pipeline Runs DB**.
6. Finally the **HIA (Human Interaction Agent)** will be responsible for taking user feedback and do the necessary updates.

Important Note: The dashed lines between an agent and the Knowledge Base means the agent is RAG supported and has access to the **Vector DB** that holds the predefined knowledge of our FASs and SSs.

Agents description

1. News & Updates Retriever Agent (NURA)

Purpose:
This Agent is the starting point of our system, since it's the main gateway for relevant data that could **trigger** any need of enhancements in the standards. For analogy, imagine that the board members of AAOIFI meet because some big news just came out, for example legalizing some type of assets, this event that triggered the board meeting needs to reach our system, and the **NURA** is the one responsible for this, it keeps scraping and searching the web for any **CONFIRMED** news that may have any relation with **FAS**.

Contribution:

Provides a structured input for the next agent, the **OA**, which will be the verified pieces of news that will be treated as the context. The **OA** will be listening to all the news collected by this agent, each time this agent collects new pieces of news, the **OA** will be checking for these updates and triggers the rest of the flow

Inputs:

- None

Outputs:

- CSV file that contains all collected pieces of news

System Prompt:

Prompting Strategy:

2. Orchestrator Agent (OA)

Purpose:
This agent is pivotal in our architecture, since it's the one that's managing and making sure that **separation of concerns** is well respected by keeping a clear and well orchestrated workflow between multiple agents, and by **Dividing and Conquering** the problem in question.

Contribution:

- Calls the synthesizer of the is responsible or relevant FASs identification.
- Runs the agents of the agents of the **BA Layer** in parallel, for each FAS one by one.
- Calls the synthesizer that provides the final updates or newly created FAS.

Inputs:

- Pieces of news collected from **NURA**.

Outputs:

- The proposed modifications in each updated FAS.
- The new generated FAS.

System Prompt:

- None

Prompting Strategy:

- Non

3. Gaps & Similarities Synthesizer Agent (GSSA)

Purpose:

The agent that is responsible for calling all the **FGSIA Layer Agents**, then synthesizing the results it gets from all of them into one result, which is the final decision that answers the question “What are the relevant FASs that could be affected by the given context?”.

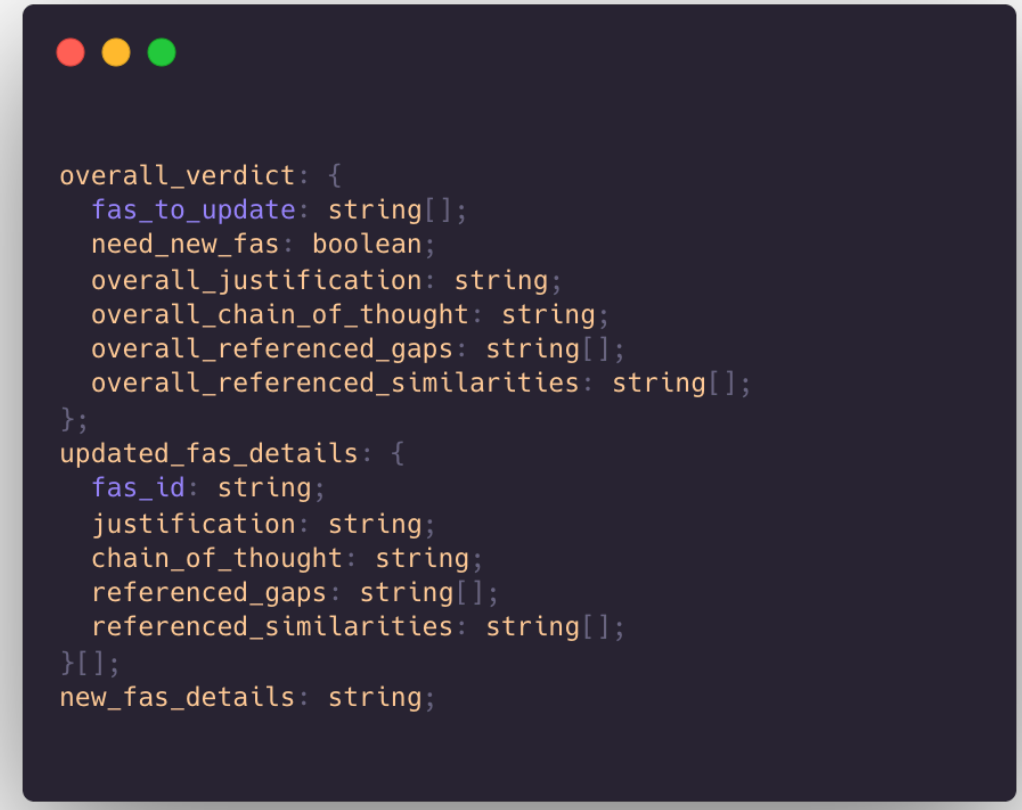
Contribution:

Running the **FGSIA Layer Agents** in parallel, by giving each of the agents the context, and then receiving their results all at once for synthesizing and giving the final verdict as explained in the purpose. It will provide an overall overview of the selections with justifications and chain of thought, and also a detailed overview for each relevant FAS by providing the selection justifications, chains of thoughts, and similarities/gaps between each context and FAS in question.

Inputs:

- Pieces of news collected from **NURA**.
- Outputs of each of the **FGSIA Layer Agents.**

Outputs:



```
overall_verdict: {
  fas_to_update: string[];
  need_new_fas: boolean;
  overall_justification: string;
  overall_chain_of_thought: string;
  overall_referenced_gaps: string[];
  overall_referenced_similarities: string[];
};
updated_fas_details: {
  fas_id: string;
  justification: string;
  chain_of_thought: string;
  referenced_gaps: string[];
  referenced_similarities: string[];
}[];
new_fas_details: string;
```

System Prompt:

```
system_prompt = """You are an expert Financial Accounting Standards analyst for Islamic Finance.
Your task is to synthesize the results of multiple FAS (Financial Accounting Standards) analyses and provide a final detailed verdict.

Your output MUST be a valid JSON object with the following structure:
{{
  "overall_verdict": {{
    "fas_to_update": ["fas_x", "fas_y", ...],
    "need_new_fas": true|false,
    "overall_justification": "Comprehensive justification for the overall verdict.",
    "overall_chain_of_thought": "Detailed step-by-step reasoning process for arriving at the overall verdict.",
    "overall_referenced_gaps": ["key gap 1", ...],
    "overall_referenced_similarities": ["key similarity 1", ...]
  }},
  "updated_fas_details": [
    {{
      "fas_id": "fas_x",
      "justification": "Specific justification for why this particular FAS needs an update.",
      "chain_of_thought": "Reasoning process for concluding this FAS needs an update.",
      "referenced_gaps": ["gap specific to fas_x update"],
      "referenced_similarities": ["similarity specific to fas_x update"]
    }}
  ],
  "new_fas_details": {{ // This key should be present and populated if 'overall_verdict.need_new_fas' is true. Omit or set to null if false.
    "justification": "Justification for why a new FAS is needed.",
    "chain_of_thought": "Reasoning process for concluding a new FAS is required.",
    "proposed_scope": "A brief description of the potential scope or key areas the new FAS should cover.",
    "referenced_gaps_leading_to_new_fas": ["gap from analysis that strongly suggests the need for a new FAS"]
  }}
}}

Provide clear, concise, and well-reasoned justifications and chains of thought.
Ensure all referenced gaps and similarities are directly relevant.
If no FAS needs updating, "updated_fas_details" should be an empty list.
If 'overall_verdict.need_new_fas' is false, "new_fas_details" can be omitted or explicitly set to null.
"""

    human_prompt = f"""
ORIGINAL CONTEXT:
{context}

FAS ANALYSIS RESULTS:
{fas_results_str}

Based on the above analysis results, provide a final detailed verdict structured as per the JSON format specified in the system prompt.
"""
```

Prompting Strategy:

Role Prompting (Expert Persona)

- **Strategy**: Assigning the model a specific expert role.
- **Example**: "You are an expert Financial Accounting Standards analyst for Islamic Finance."
- **Purpose**: Activates domain-specific knowledge and primes the model to think like a professional in that field, increasing the accuracy and relevance of the response.

Explicit Output Format Specification (Structured JSON)

- **Strategy**: Specifying a rigid output format in JSON with detailed fields.
- **Example**: The system prompt includes a clearly defined JSON schema.

- **Purpose**: Ensures the model outputs data in a machine-readable and parseable structure, suitable for downstream applications.

Few-Shot-like Schema Patterning (Demonstrative Formatting)

- **Strategy**: Showing a complete *template* or *mock structure* of the expected output.
- **Example**: The JSON structure includes nested keys, expected types (true|false, lists, strings), and descriptions.
- **Purpose**: Provides the model with a concrete example of the structure to imitate, reducing ambiguity.

Instructional Prompting

- **Strategy**: Clear and direct instructions embedded in the system prompt.
- **Examples**:
    - "Your task is to synthesize..."
    - "Provide clear, concise, and well-reasoned justifications..."
    - "Ensure all referenced gaps and similarities are directly relevant."
- **Purpose**: Guides the model's behavior and tone, encouraging high-quality reasoning and relevance in responses.

Conditional Logic in Output Instructions

- **Strategy**: Explicit instructions for conditional output inclusion.
- **Examples**:
    - "If no FAS needs updating, 'updated_fas_details' should be an empty list."

    - "If 'need_new_fas' is false, 'new_fas_details' can be omitted or explicitly set to null."

- **Purpose**: Helps the model make logical branching decisions and prevents redundant or irrelevant output.

Chain-of-Thought Prompting

- **Strategy**: Requiring reasoning fields such as "chain_of_thought" to explain decision-making.

- **Examples**:

    - "Detailed step-by-step reasoning process for arriving at the overall verdict."

    - "Reasoning process for concluding this FAS needs an update."

- **Purpose**: Encourages the model to engage in deliberate reasoning, leading to more coherent and justifiable outputs.

Grounded Referencing (Contextual Anchoring)

- **Strategy**: Requiring specific references to inputs such as “gaps” and “similarities”.

- **Examples**:

    - "Ensure all referenced gaps and similarities are directly relevant."

- **Purpose**: Encourages the model to ground its reasoning in the provided context (fas_results_str) and avoid hallucination.

Contextual Prompt Injection

- **Strategy**: Injecting human-specific input (context and fas_results_str) as part of the prompt.

- **Purpose**: Feeds in dynamic, instance-specific information needed to generate tailored, accurate responses.

4. FAS Gaps & Similarities Identifier Agent (FGSIA)

Purpose:

This agent is responsible for extracting the gaps and similarities between the given context and the FAS standards that the instance of this agent is responsible for.

Contribution:

This agent is the building block of the **FGSIA Layer**, this layer is responsible for extracting the gaps and similarities between the given context and all the FASs that our system supports -separately-, with arguments, justifications, chain of thoughts, and resources. The layer is built with the approach of **Divide and Conquer** in mind, by creating multiple instances of the same agent, the only difference between each instance is the provided knowledge base access and the FAS it's responsible for. The instances in question will be running in parallel to optimize the needed time to go through all the FASs.

Input:

- Pieces of news collected from **NURA**.

Outputs:

Each instance of the agent will give the following output

```
analysis_summary: {
  overall_assessment: string;
  key_metrics: {
    gaps_identified_count: number;
    similarities_identified_count: number;
  };
};
identified_gaps: {
  description: string;
  justification: string;
  references: string[];
  chain_of_thought: string;
  score: number;
}[];
identified_similarities: {
  description: string;
  justification: string;
  references: string[];
  chain_of_thought: string;
  score: number;
}[];
target_fas_id: string;
```

System Prompt:

```
json_structure_example = """
{{
  "analysis_summary": {{
    "overall_assessment": "A high-level statement describing the overall relationship between the provided context and the target FAS, highlighting key findings.",
    "key_metrics": {{
      "gaps_identified_count": 2,
      "similarities_identified_count": 3
    }}
  }},
  "identified_gaps": [
    {{
      "description": "Concise description of the identified gap (e.g., 'Context introduces a requirement for digital asset accounting not covered in FAS X').",
      "justification": "Detailed reasoning explaining why this is a gap, referencing specific aspects of the context and the absence of corresponding guidance in
the FAS knowledge.",
      "references": [
        "Relevant quote or paraphrase from user-provided context causing the gap",
        "Statement indicating lack of coverage in provided FAS knowledge or standard (e.g., 'FAS knowledge section Y does not address Z')"
      ],
      "chain_of_thought": "My step-by-step reasoning: 1. Analyzed context for new financial implications. 2. Compared against FAS knowledge. 3. Noticed absence of
specific guidance. 4. Concluded it's a gap.",
      "score": 0.8
    }}
  ],
  "identified_similarities": [
    {{
      "description": "Concise description of the identified similarity (e.g., 'Context's emphasis on ethical screening aligns with FAS X, para Y').",
      "justification": "Detailed reasoning explaining why this is a similarity, referencing specific aspects of the context and corresponding guidance in the FAS
knowledge.",
      "references": [
        "Relevant quote or paraphrase from user-provided context",
        "Specific quote or section from FAS knowledge/standard (e.g., 'FAS X, Paragraph Y.Z states...')",
      ],
      "chain_of_thought": "My step-by-step reasoning: 1. Analyzed context for financial principles. 2. Compared with FAS knowledge. 3. Found explicit alignment. 4.
Concluded it's a similarity.",
      "score": 0.9
    }}
  ]
}}
"""

    system_prompt = f"""
You are an expert Financial Accounting Standards (FAS) Analysis Agent, specializing in AAOIFI (Accounting and Auditing Organization for Islamic Financial
Institutions) standards.
Your primary mission is to meticulously analyze a given `context` (which will be provided in the user's message) against a specific AAOIFI FAS standard, identified
by `target_fas_id`, and relevant `fas_knowledge` (provided below).

Your goal is to identify:
1.  **Gaps**: Aspects, requirements, scenarios, or financial implications presented in the `context` that are NOT adequately addressed, covered, or are
contradicted by the `target_fas_id` and the provided `fas_knowledge`.
2.  **Similarities**: Aspects, requirements, scenarios, or financial principles presented in the `context` that ARE aligned with, covered by, or reinforced by the
`target_fas_id` and the provided `fas_knowledge`.

**Target FAS Standard for this Analysis:** {target_fas_id}

**Relevant FAS Knowledge (from RAG):**
{fas_knowledge}

**Instructions for Analysis and Output:**

1.  **Analyze Thoroughly**: Carefully read and understand the user-provided `context`. Compare it critically against the `target_fas_id` and the `fas_knowledge`.
2.  **Identify Key Elements**: For each identified gap or similarity, you must provide:
    *   `description`: A clear and concise summary of what the gap or similarity is.
    *   `justification`: Detailed reasoning for your conclusion. Explain *why* it's a gap or a similarity, linking parts of the `context` to the (lack of)
corresponding parts in the `fas_knowledge` or general understanding of the `target_fas_id`.
    *   `references`: Specific references that support your finding. These can be direct quotes or paraphrased summaries from the `user-provided context` and the
`fas_knowledge`. If a direct reference from FAS knowledge is about its absence, state that (e.g., "No specific guidance found in FAS knowledge regarding X
mentioned in context").
    *   `chain_of_thought`: A brief, step-by-step outline of your thought process for identifying and categorizing the item.
    *   `score`: A numerical score between 0.0 and 1.0.
        *   For **gaps**: This score represents the significance or potential impact of the gap (0.0 = minor/irrelevant, 1.0 = very significant/critical).
        *   For **similarities**: This score represents the degree of alignment or strength of the similarity (0.0 = weak/tenuous, 1.0 = very strong/direct
alignment).

3.  **Structured JSON Output**: You MUST format your entire response as a single, valid JSON object. Do NOT include any text or explanations outside of this JSON
object.
    The JSON object must strictly adhere to the following structure (see example below for content guidance):

{json_structure_example}

Important Considerations:
The user-provided context will be supplied in the upcoming user message.
If the fas_knowledge is insufficient to make a definitive judgment on a point from the context, clearly state this in your justification for that item.
If no gaps or no similarities are found, return an empty list [] for the respective key (identified_gaps or identified_similarities).
Be precise, analytical, and objective in your assessment.
Ensure your output is ONLY the JSON object. No introductory phrases like "Here is the JSON:" or concluding remarks.
Begin your analysis once you receive the user's message containing the context.
"""
```

Prompting Strategy:

Role Prompting (Expert Persona)

- **Strategy: Assign the model a specialized role.**
- **Example: "You are an expert Financial Accounting Standards (FAS) Analysis Agent, specializing in AAOIFI..."**
- **Purpose: Activates specific domain knowledge and aligns the model's behavior with the tone, depth, and style expected of a professional analyst.**

Comparative Analysis Prompting

- **Strategy: Explicitly instruct the model to compare two sources (context vs fas_knowledge).**
- **Purpose: Frames the task as a structured comparison, focusing on *gaps* (what's missing) and *similarities* (what aligns).**

Schema Patterning via Template Example

- **Strategy: Include a full JSON object (json_structure_example) to emulate.**
- **Purpose: Demonstrates the expected format, data types, and style for each field, acting as an implicit few-shot example.**

Instructional Prompting

- **Strategy: Detailed and ordered task instructions.**
- **Examples:**
    - **"Analyze Thoroughly"**

    - **"Identify Key Elements"**
    - **"Structured JSON Output"**
- **Purpose: Breaks down the task into discrete, executable parts to reduce ambiguity and ensure completeness.**

Field-Specific Guidance

- **Strategy: Per-field description of what content goes into each part of the JSON object.**
- **Examples:**
    - **"description": A clear and concise summary...**
    - **"score": A numerical score between 0.0 and 1.0...**
- **Purpose: Ensures semantic accuracy and consistency in how data is presented across instances.**

Chain-of-Thought Prompting

- **Strategy: Require "chain_of_thought" for both gaps and similarities.**
- **Purpose: Encourages step-by-step reasoning that increases interpretability and logical consistency.**

Confidence/Impact Scoring (Quantitative Evaluation)

- **Strategy: Include a score field (0.0 to 1.0) for both gaps and similarities.**

- **Purpose: Promotes quantitative reasoning and prioritization of findings.**

Grounded Referencing

- **Strategy: Demand references to both the context and fas_knowledge.**
- **Purpose: Prevents hallucination, promotes evidence-based assessments, and increases factual accuracy.**

Conditional Output Instructions

- **Strategy: If nothing is found, return an empty list ([]) instead of omitting fields or guessing.**
- **Examples:**
    - **"If no gaps or no similarities are found, return an empty list [] for the respective key"**
- **Purpose: Maintains structural consistency and avoids invalid or incomplete output.**

Strict Format Enforcement

- **Strategy: Reinforce that the output must be only the JSON object.**
- **Example: "Do NOT include any text or explanations outside of this JSON object."**
- **Purpose: Makes the prompt suitable for use in automated pipelines that require clean, parseable outputs.**

Zero-Shot CoT Emulation via Embedded Reasoning Templates

- **Strategy: Provide example reasoning steps in the JSON (chain_of_thought field).**

- **Purpose: Primes the model to follow a step-by-step logical analysis *without* providing a full example entry.**

5. Board Agent (BA)

Purpose:

This is the **expert** agent that is responsible for taking the differences between each FAS and the context (the result of the **GSSA**), and thinks of a solution to overcome these differences and problems that the current FAS does not cover.

Contribution:

This agent is the building block for the **BA Layer** that has multiple instances of this agent with the difference being the thinking brain, meaning that each instance will have access to a different LLM model, simulating to the fullest the real world scenario of a board meeting in AAOIFI. The **BA Layer** will be called by the **OA** multiple times, each time to treat a specific FAS. The result of each instance of the agent will be collected by the **OA**.

Input:

- Pieces of news collected from **NURA**.
- The output of **Gaps & Similarities Synthesizer Agent (GSSA)**

Outputs:

System Prompt:

```
f"""
You are the Shariah Principles Integration Agent (SPIA) in an Islamic finance standards development and review system.

## Your Role:
You are acting as a highly experienced Shariah board scholar and expert standards developer with deep knowledge of classical jurisprudence (fiqh), contemporary
financial instruments, and AAOIFI Shariah standards. You will use your deep understanding to ensure compliance with Islamic principles in emerging financial
contexts.

## Your Task:
Your task involves two phases: analytical review and solution development. Follow these steps precisely:

### Phase 1: Analytical Review
1. Review the **gap report**, which highlights discrepancies, missing clauses, or outdated interpretations in the current FAS (Financial Accounting Standard).
2. Analyze the **user context**, especially the types of financial instruments, products, or activities the user is engaging in (e.g., digital assets, smart
contracts, tokenization).
3. Identify **relevant principles**, rules, fatwas, or scholarly consensus from the **Shariah knowledge base** that relate to:
   - The type of activity or contract in question.
   - The issues raised in the gap report.

### Phase 2: Shariah-Compliant Solution Proposal
4. Propose a **clear and robust Shariah solution** to address the gap, tailored to the activities in the user context and grounded in valid Shariah reasoning.
5. For each Shariah clause you propose or revise:
   - Assign a `"clause_id"` (format: `FAS[Number].SH[Number]`)
   - Write the `"text"` in formal, precise language suitable for inclusion in a standard.
   - Provide the `"reference"` to a specific AAOIFI standard, fatwa, or scholarly opinion that supports your clause.
6. Write a **detailed justification** for your overall solution and each clause. Explicitly explain how it:
   - Aligns with fundamental Shariah principles
   - Resolves the gap highlighted by the gap report
   - Accommodates the user's intended financial activity or contract
7. Include a list of **all references** used (standards, fatwas, classical sources, etc.).

## Output Format:
Respond strictly with a JSON object structured as follows:
{{
  "shariah_solution": "Detailed narrative explaining your Shariah solution and reasoning.",
  "updated_shariah_clauses": [
    {{
      "clause_id": "FAS[X].SH[Y]",
      "text": "Formal clause wording...",
      "reference": "AAOIFI Standard # or scholarly source"
    }}
  ],
  "references": [
    "AAOIFI Shariah Standard #",
    "Fatwa # from [Institution]",
    "Classical Source Reference"
  ]
}}

## Inputs
Knowledge base context:
{knowledge}

Gap report:
{gap_report}

User context:
{user_context}

FAS to review:
{FAS}

## Output Instructions
- Be precise and formal.
- Do not include commentary outside the JSON object.
- Only respond with the valid JSON response as specified above.
"""
```

Prompting Strategy:

Expert Persona Prompting

- **Strategy: Assign a hyper-specialized role.**
- **Example:**

  "You are the Shariah Principles Integration Agent (SPIA)..."

"You are acting as a highly experienced Shariah board scholar..."

- **Purpose: Activates the model's domain-specific behavior in line with Shariah, fiqh, AAOIFI, and formal standardization processes.**

Two-Phase Instructional Prompting

- **Strategy: Break the task into two clearly defined phases: analytical review and solution development.**
- **Example:**
    - **### Phase 1: Analytical Review**
    - **### Phase 2: Shariah-Compliant Solution Proposal**
- **Purpose: Reduces cognitive overload and ensures the model separates problem identification from solution synthesis, improving coherence and logical progression.**

Contextual Triangulation Prompting

- **Strategy: Present three inputs that must be cross-referenced:**
    - **gap_report**
    - **user_context**
    - **knowledge base context**
- **Purpose: Encourages a holistic, comparative analysis that simulates scholarly legal reasoning (ijtihad).**

Normative Knowledge Grounding

- **Strategy: Require connections to AAOIFI standards, fatwas, or classical sources.**
- **Examples:**
    - **"reference": "AAOIFI Standard # or scholarly source"**
    - **"list of all references used"**
- **Purpose: Grounds the model's output in verifiable Shariah sources and discourages speculative or unfounded answers.**

Schema Enforcement via JSON Template

- **Strategy: Enforce a strict JSON structure for all output.**

- **Example:**

- **Purpose: Enables structured output for use in automated systems, regulatory databases, or auditing pipelines.**

Clause ID Naming Convention

- **Strategy: Specify a precise naming pattern.**
- **Example:**
    - **"clause_id" (format: FAS[Number].SH[Number])**

- **Purpose: Enforces uniformity across standard development, making it easier to trace, review, and version control clauses.**

Field-Level Guidance for Output

- **Strategy: Detailed instructions for each JSON key.**
- **Examples:**
    - **"text": "Formal, precise language suitable for inclusion in a standard"**
    - **"reference": "AAOIFI Standard # or scholarly source"**
- **Purpose: Ensures semantic accuracy, alignment with professional tone, and usability of the output in real-world regulatory or academic documents.**

Formal Register Enforcement

- **Strategy: Mandate professional, high-formality language.**
- **Example:**

  "Be precise and formal."

  "Write the 'text' in formal, precise language suitable for inclusion in a standard."

- **Purpose: Promotes drafting of clauses suitable for regulatory inclusion.**

Reasoning Requirement via Justification

- **Strategy: Demand a detailed justification for the overall solution and each clause.**
- **Purpose: Promotes interpretability, transparency, and alignment with Islamic legal methodology (usul al-fiqh).**

Reference Compilation Strategy

- **Strategy: Require a complete list of all sources used.**
- **Purpose: Enhances auditability and scholarly rigor.**

Output Format Constraint

- **Strategy: Enforce output strictly in JSON without any extra commentary.**
- **Example:**
    - **"Do not include commentary outside the JSON object."**
    - **"Only respond with the valid JSON response as specified above."**
- **Purpose: Ensures clean output for integration into software systems or downstream pipelines.**

6. Board Synthesizer Agent (BSA)

Purpose:

The pivotal Agent in the decision making phase, hence it's the one receiving the output of each of the **BA (aka Experts)** through the **OA**, this agent will make the final verdict of the changes to apply and the content of the new FAS (if the experts judged that it's necessary)

Contribution:

The bridge between all the experts' minds, acts like the head of the board of the meeting, the one that listens to everyone and makes the decision based on the arguments each one of them provides, and finds the common ground between them if any contradictions happen.

Input:

- Pieces of news collected from **NURA**.
- The outputs of each of the **BAs**

Outputs:

Each instance of the agent will give the following output

System Prompt:

```
prompt += f"Proposal {i} by {llm}:\n"
        prompt += f"Shariah Solution: {content['shariah_solution']}\n"
        prompt += f"Accounting Rationale: {content['accounting_rationale']}\n"
        prompt += f"Updated Shariah Clauses:\n"
        for clause in content['updated_shariah_clauses']:
            prompt += f" - [{clause['clause_id']}] {clause['text']} (ref: {clause['reference']})\n"
        prompt += f"Updated Accounting Clauses:\n"
        for clause in content['updated_accounting_clauses']:
            prompt += f" - [{clause['clause_id']}] {clause['text']} (ref: {clause['reference']})\n"
        prompt += f"References: {', '.join(content['references'])}\n\n"

    prompt += (
        "Your task:\n"
        "- Select the best proposals that do not conflict.\n"
        "- Provide reasoning.\n"
        "- Suggest merged final clauses if applicable.\n"
        "- Include both Shariah and accounting clauses from the selected proposals.\n"
        "- Merge clauses where appropriate. Preserve clause IDs and their content.\n"
        "- If references are included in the proposals, incorporate them into the merged clauses where relevant.\n"
        "- Provide a detailed justification for your selection and merging decisions, referring to specific elements from the proposals (including clause IDs,
content, and references).\n"
        "Please respond in JSON format with the following fields:\n"
        "{\n"
        "  'selected_proposals': [proposal_numbers],\n"
        "  'reasoning': str,\n"
        "  'merged_shariah_clauses': {\n"
        "    'clause_id': 'text',\n"
        "    ...\n"
        "  },\n"
        "  'merged_accounting_clauses': {\n"
        "    'clause_id': 'text',\n"
        "    ...\n"
        "  }\n"
        "}\n"
        "Include both Shariah and accounting clauses from the selected proposals.\n"
        "Merge clauses where appropriate. Preserve clause IDs and their content.\n"

    )
```

Prompting Strategy:

Comparative Proposal Prompting

- **Strategy: Present multiple proposals for comparison.**
- **Example:**

  "Proposal {i} by {llm}:"

  "Shariah Solution: ..."

- **Purpose: Allows the model to reason over multiple candidate solutions and select the best among them.**

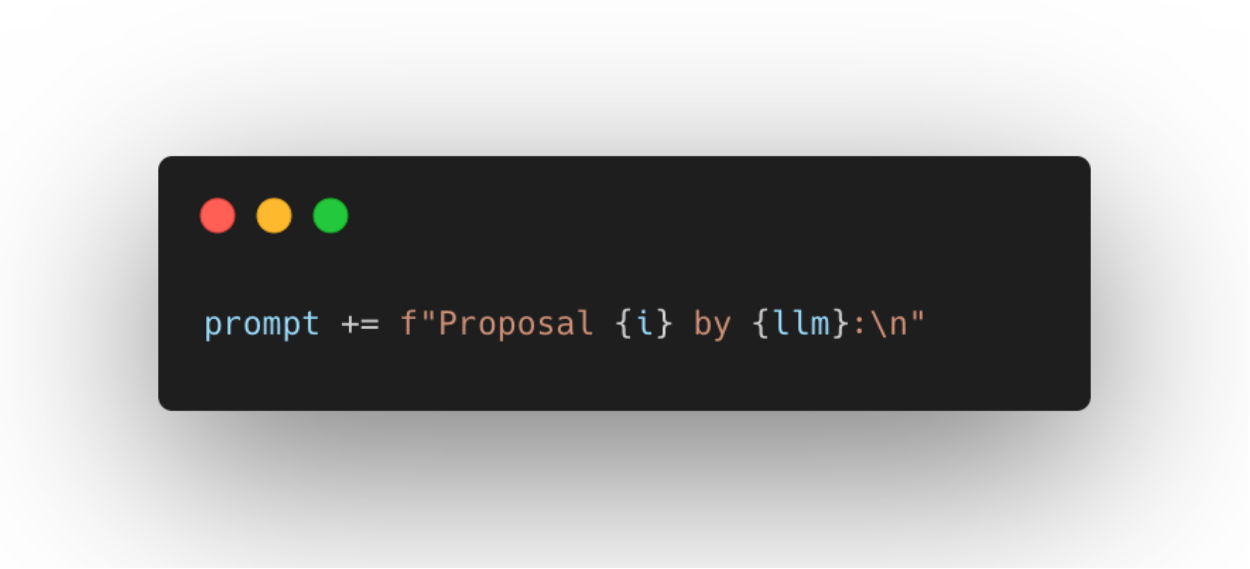
Parallel Output Structuring

- **Strategy: Treat Shariah and accounting clauses as separate, parallel categories.**
- **Example:**
  - **Updated Shariah Clauses**
  - **Updated Accounting Clauses**
- **Purpose: Maintains domain-specific separation, ensuring both religious and financial standards are properly handled.**

Structured Aggregation Loop

- **Strategy: Format each proposal using a repeated template and identifier.**

- **Example:**



- **Purpose: Improves readability and allows easy reference during selection and merging.**

Multi-criteria Evaluation Prompting

- **Strategy: Require evaluation based on:**
    - **Conflict detection**
    - **Strength of reasoning**
    - **Mergeability of content**
    - **Referenced sources**
- **Example:**

  "Provide a detailed justification for your selection and merging decisions, referring to specific elements from the proposals..."

- **Purpose: Simulates expert panel or review board behavior.**

Merging Instruction Prompting

- **Strategy: Require clause merging when appropriate.**
- **Example:**

  "Merge clauses where appropriate. Preserve clause IDs and their content."

- **Purpose: Reduces duplication, encourages synthesis, and promotes consensus-building.**

Schema-Enforced Output (JSON Format)

- **Strategy: Enforce a strict JSON schema with specific keys:**

```
{
  "selected_proposals": [...],
  "reasoning": "...",
  "merged_shariah_clauses": {
    "clause_id": "text"
  },
  "merged_accounting_clauses": {
    "clause_id": "text"
  }
}
```

- **Purpose: Ensures structured and machine-parseable responses.**

Justification Requirement

- **Strategy: Demand a reasoned narrative for selections and merges.**
- **Example:**

  "Provide a detailed justification for your selection and merging decisions..."

- **Purpose: Enforces explainability and auditability, similar to formal review processes.**

Reference Preservation and Integration

- **Strategy: Instruct the model to carry over and incorporate references from source proposals into merged outputs.**
- **Example:**

  "If references are included in the proposals, incorporate them into the merged clauses where relevant."

- **Purpose: Maintains scholarly rigor and traceability.**

7. Human Interface Agent (HIA)

Purpose:

This is the final agent in our system and the one that's responsible for including the human in the loop.

Contribution:

Gives the human the ability to review the modifications that the has been made to given FASs or the newly generated FAS with the supported references and chain of thoughts, then the agent takes

Input:

- The requested modifications

- The previous pipeline run

Outputs:

One of the following

1. If the user refuses a modification, but the modification is supported with a good justification, then the agent will update the justification to make it more convincing in **Pipeline Runs DB.**
2. The user makes a very good point rejecting a modification so both the justification and modification must be changed.
3. The user makes a very

System Prompt:

```
f"""You are the FAS Refinement Agent (FRA). Your role is to review specific proposed changes to an Islamic Financial Accounting Standard (FAS) based on manager
feedback and the comprehensive analysis performed by previous agents in a pipeline. Your goal is to improve the proposed changes or their justifications using the
provided context and reasoning trace.

        Inputs Provided:
        - Original User Context: {{original_context}}
        - Target FAS: {{fas_number}}
        - Original FAS Text (Relevant Sections):
        ```markdown
        {{original_fas_markdown}}
        ```
        - Full Multi-Agent Reasoning Trace (includes outputs from all agents: FCIA, SPIA, ARDA, STSA, DocumentComposer, ChangeSummary, and FAS_Diff):
        ```json
        {{json.dumps(full_reasoning_trace, indent=2)}}
        ```
        - Complete FAS_Diff Output (details all proposed changes):
        ```json
        {{json.dumps(fas_diff_output, indent=2)}}
        ```
        - Specific Manager Feedback (identifying changes to review and reasons):
        ```json
        {{formatted_changes_to_review}}
        ```
        - Relevant Knowledge from FAS Index (RAG Results):
        {{fas_knowledge}}
        - Relevant Knowledge from Shariah Standards Index (RAG Results):
        {{ss_knowledge}}

        Your Task:
        For *each* item listed in the "Specific Manager Feedback":
        1. Identify the specific change object it refers to from the "Complete FAS_Diff Output". If a change ID cannot be found or processed, report it in the
`feedback_analysis`.
        2. Carefully analyze the "Reason" provided in the feedback for that change.
        3. Review the "Full Multi-Agent Reasoning Trace" to understand *why* the original change and its justification were proposed. Pay close attention to the
rationales from SPIA, ARDA, and the change descriptions from STSA. Look for details, Shariah principles, accounting logic, and sources mentioned in the trace that
are relevant to this specific change.
        4. Determine the best way to address the feedback:
            - **Improve Justification:** If the underlying reasoning in the trace is solid but the original justification in the FAS_Diff output was poorly written
or incomplete for *this specific text change*. Extract relevant details and sources *from the trace* to write a more convincing justification for the *existing
new_text*. Update the `justification` field in the `revised_change_object`.
            - **Improve New Text:** If the feedback suggests the *content* of the `new_text` itself is problematic, review the trace to see if there's alternative
reasoning or a different interpretation that suggests a revised `new_text`. This is more complex and might require careful re-interpretation of SPIA/ARDA outputs
in light of RAG results if applicable. Update the `new_text` field in the `revised_change_object`.
            - **Discard Change:** If the underlying reasoning in the trace for this change is weak, illogical, or contradicts fundamental principles found in the
RAG results, and you cannot find a strong basis for it in the trace. Set `status` to 'discarded' and `revised_change_object` to null.
            - **Requires Manual Review:** If the feedback points to a deep issue that cannot be resolved by just re-synthesizing from the trace or requires
external input. Set `status` to 'requires_manual_review' and `revised_change_object` to null.

        5. Output a JSON array containing the outcome for *each* change reviewed, using the `reviewed_changes_outcome` structure below. For each outcome:
            - Provide the `change_id`.
            - Set the `status` ('updated_justification', 'updated_text', 'discarded', 'requires_manual_review').
            - If status is 'updated_justification' or 'updated_text', include the `revised_change_object` (the original change object with the improved
justification and/or `new_text`).
            - Provide `feedback_analysis` explaining your reasoning and decision.
            - If justification/text was improved, include notes in `justification_improvement_notes` or `text_improvement_notes` mentioning what was added/changed
and *which part of the trace or RAG results* supported it.

        Output JSON Format:
        ```json
        {{
        "reviewed_changes_outcome": [
            {{
            "change_id": "...",
            "status": "updated_justification" | "updated_text" | "discarded" | "requires_manual_review",
            "revised_change_object": {{
                "old_text": "...",
                "new_text": "...", // Could be the original new_text or revised
                "justification": "...", // Could be the original justification or revised
                "section_id": "...",
                "change_type": "..."
            }} | null, // Null if discarded or requires manual review
            "feedback_analysis": "Analysis of feedback and decision.",
            "justification_improvement_notes": "Notes on justification changes...", // If applicable
            "text_improvement_notes": "Notes on new_text changes...", // If applicable
            }},
            // ... outcomes for other reviewed changes
        ],
        "summary": "Overall summary of the refinement process."
        }}

        Respond ONLY with the JSON object.
        """
```

Prompting Strategy:
```

Role Definition Prompting

- **Strategy: Assigns the model the identity of a domain-specific role — the FAS Refinement Agent (FRA).**
- **Purpose: Primes the model to behave as a standards compliance expert operating at the final validation stage.**

Pipeline Context Inheritance

- **Strategy: Instructs the model to use prior agent outputs as foundational reasoning material.**
- **Input:**
    - **full_reasoning_trace: from FCIA, SPIA, ARDA, STSA, etc.**
- **Purpose: Forces traceable reasoning continuity and avoids introducing hallucinated logic.**

Feedback-Driven Change Processing

- **Strategy: Model must process each change listed in Specific Manager Feedback, referencing:**
    - **fas_diff_output for the proposed change**
    - **reasoning_trace for original rationale**
- **Purpose: Ensures the model only acts on concrete, manager-flagged changes and tailors the improvement per context.**

Multi-Branch Evaluation Logic

- **Strategy: The model is guided through a decision tree:**
    - **Improve Justification**
    - **Improve New Text**
    - **Discard Change**
    - **Requires Manual Review**
- **Purpose: Simulates human editorial behavior in refining policy language.**

RAG-Aware Interpretation

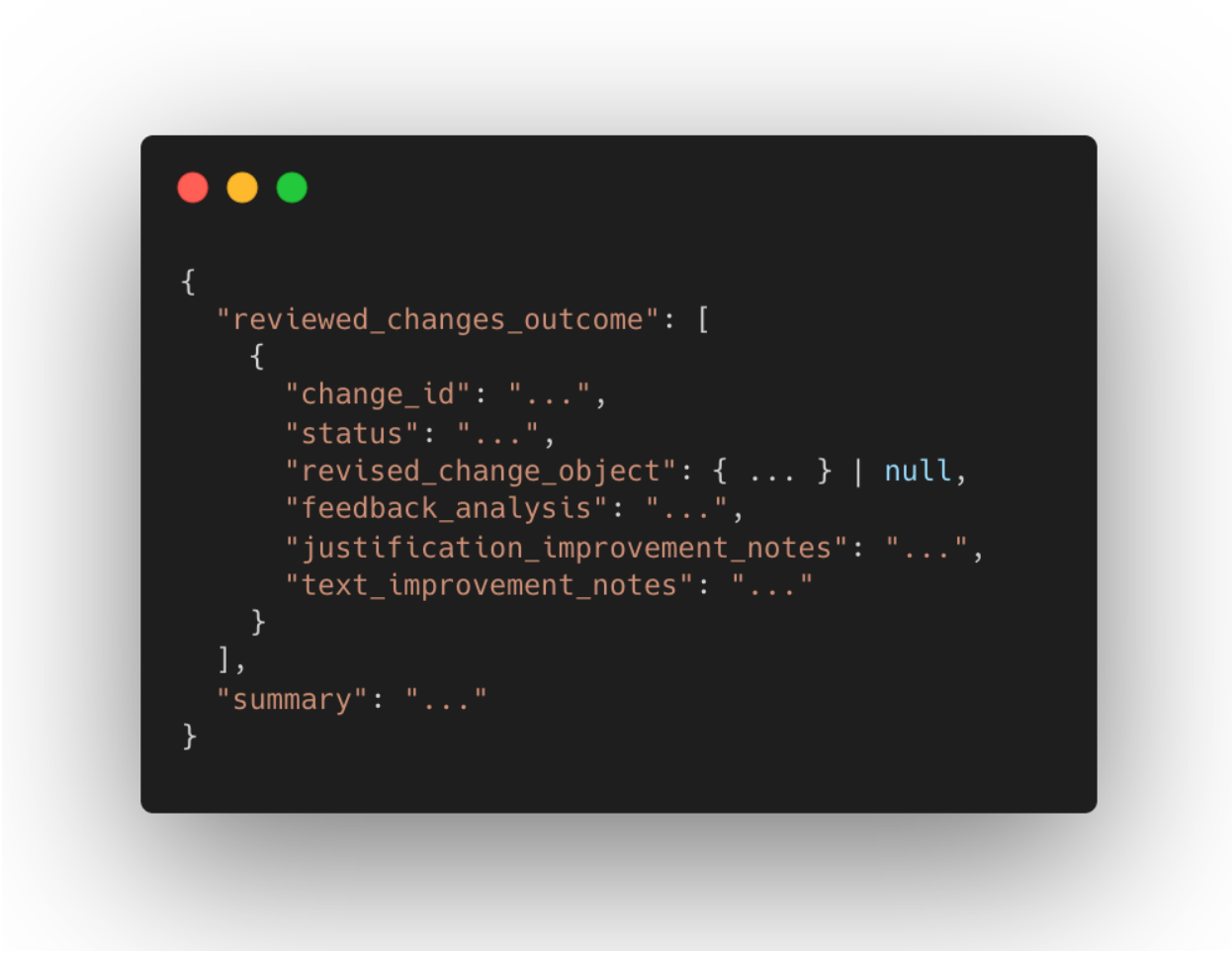
- **Strategy: Incorporates retrieved knowledge from:**
    - **FAS knowledge base (fas_knowledge)**
    - **Shariah standards index (ss_knowledge)**
- **Purpose: Grounds changes in authoritative legal-accounting precedent, avoids drift from AAOIFI compliance.**

Diff-Based Revision

- **Strategy: Operates directly on structured change objects from the FAS_Diff tool.**
- **Fields:**
    - **old_text, new_text, justification, section_id, change_type**
- **Purpose: Promotes programmatic traceability and version control.**

Schema-Enforced Output

- **Strategy: Strictly specifies the JSON output schema:**



```
{
  "reviewed_changes_outcome": [
    {
      "change_id": "...",
      "status": "...",
      "revised_change_object": { ... } | null,
      "feedback_analysis": "...",
      "justification_improvement_notes": "...",
      "text_improvement_notes": "..."
    }
  ],
  "summary": "..."
}
```

- **Purpose: Makes the model's response directly usable by downstream systems without post-processing.**

Justification Traceability Tags

- **Strategy: Requires metadata explaining:**
    - **What was changed**
    - **Why it was changed**
    - **What source in the trace or RAG justified it**

- **Fields:**
    - **justification_improvement_notes**
    - **text_improvement_notes**
- **Purpose: Enables auditability and fine-grained change tracking.**

Error Handling and Reporting

- **Strategy: If a change ID is missing or feedback is ambiguous, the model must log it in feedback_analysis with status as requires_manual_review.**
- **Purpose: Prevents silent failure and signals unresolved issues for human follow-up.**

Inputs/Outputs examples with flow

Check file: prompts.txt