

DAY-7 TASK REPORT

Finalization & Handover of MCQ Generation System

Name: Arya

Role: AI / LLM Content Generation

Status: Completed

Date: December 13, 2025

1. Work Summary

The objective of Day-7 was to finalize all deliverables produced during the MCQ generation workflow and prepare them for backend integration. This phase focuses on locking prompt logic, confirming the strict JSON schema used for API validation, and documenting representative sample outputs to support quality assurance and system handover.

2. Final Deliverables Overview

The following components were finalized and are included in this handover package:

- **Final Optimized LLM Prompt (Version-2):**
Tuned for Bloom's Taxonomy compliance, strong distractor logic, and strict JSON-only output.
 - **Locked JSON Schema:**
The final validation contract that backend responses must strictly adhere to.
 - **Sample Output 1:**
Standard MCQ generation demonstrating Recall and Application-level questions.
 - **Sample Output 2:**
Optimized, high-difficulty MCQs demonstrating true Analysis-level reasoning.
 - **Integration Guidelines:**
Operational notes covering API temperature settings and error-handling strategy.
-

3. Final Prompt (Version-2 Reference)

(Note: The complete prompt text is provided in the attached file: *Prompt_V2_Final.txt*)

Summary of Optimization

The optimized Version-2 prompt enforces the following constraints to ensure production stability and pedagogical quality:

- **Bloom's Taxonomy Alignment:**
Enforces a fixed distribution of **1 Recall, 2 Application, and 1 Analysis** question per transcript chunk.
 - **Cognitive Load Balance:**
Ensures “Hard” questions require logical inference rather than obscure factual recall.
 - **Strict JSON-Only Output:**
Eliminates conversational text (e.g., “Here is your JSON”) to prevent API parsing failures.
 - **Negative Constraints:**
Explicitly forbids options such as “*All of the above*” and “*None of the above*”.
-

3.1 Prompt Behaviour & Design Constraints (Production Rules)

Role: You are an expert Psychometrician and Instructional Designer.

Output Requirement: Strict JSON only.

Task: Generate exactly 4 MCQs from the transcript chunk.

Difficulty Distribution (Bloom's Taxonomy):

- Question 1 – Easy (Recall): Direct fact or definition from text
- Question 2 – Medium (Application): Apply a concept to a scenario
- Question 3 – Medium (Application): Classify or interpret information
- Question 4 – Hard (Analysis): Infer structural or functional implications

Design Constraints:

- Exactly 4 options per MCQ
- Distractors must be plausible but incorrect
- Do not use “All of the above” or “None of the above”
- Avoid repeating transcript phrasing in the question stem
- Include correct_option_index for backend verification
- Output must strictly match the JSON schema

Status: LOCKED & READY FOR PRODUCTION

4. Final JSON Schema (Locked Version)

Technical Note:

This schema is finalized. All backend responses must match this structure exactly. Any deviation should trigger a retry or validation failure.

```
{  
  "name": "mcq_assessment",  
  "strict": true,  
  "schema": {  
    "type": "object",  
    "properties": {  
      "assessment_metadata": {  
        "type": "object",  
        "properties": {  
          "topic_summary": { "type": "string" },  
          "blooms_distribution": { "type": "string" }  
        },  
        "required": ["topic_summary", "blooms_distribution"],  
        "additionalProperties": false  
      },  
      "questions": {  
        "type": "array",  
        "items": {  
          "type": "object",  
          "properties": {  
            "id": { "type": "integer" },  
            "difficulty": { "type": "string", "enum": ["Easy", "Medium", "Hard"] },  
            "cognitive_level": { "type": "string", "enum": ["Recall", "Application", "Analysis"] },  
            "question_stem": { "type": "string" },  
            "options": {  
              "type": "array",  
              "items": {  
                "type": "object",  
                "properties": {  
                  "text": { "type": "string" },  
                  "is_correct": { "type": "boolean" }  
                }  
              }  
            }  
          }  
        }  
      }  
    }  
  }  
}
```

```
"options": {  
    "type": "array",  
    "items": { "type": "string" },  
    "minItems": 4,  
    "maxItems": 4  
},  
"correct_option_index": {  
    "type": "integer",  
    "description": "0-based index of the correct answer"  
},  
"correct_answer_text": { "type": "string" },  
"explanation": { "type": "string" }  
},  
"required": [  
    "id",  
    "difficulty",  
    "cognitive_level",  
    "question_stem",  
    "options",  
    "correct_option_index",  
    "correct_answer_text",  
    "explanation"  
],  
"additionalProperties": false  
}  
}  
},  
"required": ["assessment_metadata", "questions"],
```

```
    "additionalProperties": false
  }
}
```

5.1 Sample Output 1 – Standard Transcript Chunk MCQs

```
{
  "assessment_metadata": {
    "topic_summary": "Array structure and indexing",
    "blooms_distribution": "1 Recall, 2 Application, 1 Analysis"
  },
  "questions": [
    {
      "id": 1,
      "difficulty": "Easy",
      "cognitive_level": "Recall",
      "question_stem": "According to the text, how are elements stored inside an array?",
      "options": [
        "In random memory locations",
        "In a contiguous memory location",
        "In linked nodes",
        "In separate files"
      ],
      "correct_option_index": 1,
      "correct_answer_text": "In a contiguous memory location",
      "explanation": "The transcript explicitly states that arrays store elements in a contiguous memory location."
    }
  ]
}
```

```
}
```

5.2 Sample Output 2 – Optimized (Hard / Analysis)

```
{
```

```
  "assessment_metadata": {
```

```
    "topic_summary": "Array data structure constraints",
```

```
    "blooms_distribution": "1 Recall, 2 Application, 1 Analysis"
```

```
  },
```

```
  "questions": [
```

```
    {
```

```
      "id": 4,
```

```
      "difficulty": "Hard",
```

```
      "cognitive_level": "Analysis",
```

```
      "question_stem": "Why does the requirement for 'similar types of elements' strictly limit how arrays handle mixed data?",
```

```
      "options": [
```

```
        "It prevents the array from using zero-based indexing",
```

```
        "It forces the creation of separate arrays for different data types",
```

```
        "It automatically converts floats to integers to save space",
```

```
        "It causes the array to delete the first element"
```

```
      ],
```

```
      "correct_option_index": 1,
```

```
      "correct_answer_text": "It forces the creation of separate arrays for different data types",
```

```
      "explanation": "The text explains that storing floating values requires a separate array, implying arrays cannot hold mixed data types."
```

```
    }
```

```
  ]
```

```
}
```

6. Conclusion

All MCQ generation components have been finalized, validated, and documented.

Key Outcomes

- **Reliability:** Locked schema ensures API responses remain frontend-safe
- **Quality:** Analysis-level questions now test genuine conceptual understanding
- **Scalability:** Pipeline supports transcript chunks of varying length

Recommendation

For backend integration, set the LLM **temperature to 0.2** to maximize determinism and output consistency.

The MCQ generation module is now fully handed over to the development team.

Submitted by

Arya

AI / LLM Content Generation

Skill Guru Foundation