You are an Expert AI Engineer

I have to build an AI Examiner who can mark narrative answer using LLM and ideal answer. AI system should understand the actual meaning of ideal answer, after that, he check user provided answer. Finally, based on ideal answer marking the narrative answer.

1. System Design & Tool Selection

1.1) Select the Core LLM --> Choose a powerful LLM (e.g., GPT-4, Claude 3, or a strong open-source model like Llama 3) that excels at complex reasoning, semantic understanding, and instruction following.

1.2) Define the Grading Rubric --> Establish clear, quantifiable criteria for marking (e.g., Points for main idea, Points for supporting evidence, Points for clarity/structure). This is crucial for consistent marking.

1.3) Choose a Prompting Framework --> Use a framework like RAG (Retrieval-Augmented Generation) if the ideal answers are very long or domain-specific, but Chain-of-Thought (CoT) prompting will be key for the marking logic itself.

2. Prompt Engineering (The Core Logic)

2.1) Define the Role & Goal --> Start by instructing the LLM to adopt the persona of an expert academic examiner.

Prompt Component: "You are an expert academic examiner specializing in [Subject]. Your task is to evaluate a Student Answer based on an Ideal Answer and a Grading Rubric. You must provide a score and detailed justification."

2.2) Provide the Ideal Answer --> Give the LLM the definitive, correct response.

Prompt Component: "IDEAL ANSWER: [Insert detailed ideal answer here]"

2.3) Inject the Grading Logic (CoT) --> Instruct the LLM to perform a step-by-step comparison before assigning a final score.

Prompt Component: "PROCESS: Step 1: Semantic Understanding: Analyze the Ideal Answer and break it down into 3-5 key concepts. Step 2: Comparison: Compare the Student Answer to each key concept. Does the student cover it? Is the explanation accurate? Step 3: Point Allocation: Allocate points based on the Rubric for each matched concept. Step 4: Final Score & Justification: Sum the points and provide a detailed reason for the final score."

2.4) Input the Student Answer --> Clearly separate the student's submission.

Prompt Component: "STUDENT ANSWER: [Insert student's narrative response here]"

2.5) Specify the Output Format --> Use JSON or a clear structured text format for easy machine consumption and consistency.

Output Structure: {"Score": "X/10", "Justification": "...", "Key\_Concepts\_Covered": ["Concept A (2/3 points) - Reason...", "Concept B (1/2 points) - Reason..."]}

3. Deployment & Maintenance

3.1) API Integration & Scalability --> Integrate the LLM's API or inference engine into your application backend. Ensure the system can handle the required throughput (e.g., rate limits, batch processing).

3.2) Bias Monitoring --> Periodically check the system to ensure marking is fair and unbiased across different writing styles or demographic inputs. LLMs can inherit subtle biases.

3.3) Auditing & Explainability --> Store the detailed LLM justification for every grade. This is critical for students to understand their mark and for system auditing. The CoT prompt ensures this is generated.