等线Multimodal Coding Intro Module V2

In this course you will get all the necessary information, tips and key aspects to task in our MM Coding project. Stick to these alignments and you will be set for success!

Course Overview

Throughout this course, you will come across **knowledge check questions**. These are **NOT graded**—they are simply here to help you reinforce what you have learned in each section. Feel free to use them to gauge your understanding as you go.
At the end of the course, there will be a **graded assessment with 18 questions**. **To pass, you will need to answer at least 15 correctly**, but don't worry! These questions are **NOT** designed to trick you, all answers can be found throughout the course material, and you can have the specifications document at your side during the screening.

Take your time, and good luck!

Introduction

Project Overview

Welcome to **Sundial Coding Multimodal RLHF**! In this project, you will upload a coding-related image, compose prompts based on that image, rate a model response, and improve it with a rewrite. Your work on this project will enhance the ability of cutting-edge LLM models to analyze coding-related images.

Please save the attempter spec doc here and reference it as you task.

IMPORTANT!
The goal of this task is to get the model to FAIL in its response to your image-based coding problem.

Tasking Process

A task in this project will involve the following steps:

Step 1 - Image selection

In this step you need to select an image using the file selector tool. You can either input your own image link or browse and choose an image directly within the tool. When making your selection, it is crucial to consider the following guidelines:

1. **Code-Related Image**: Ensure the image is relevant to coding or programming.
2. **Prompt Dependency**: The image should provide information essential for constructing your prompt. The prompt must rely on details from the image to be fully addressed.
3. **Programming Language Alignment**: Choose an image that aligns with one of the available programming languages and fits the context of your coding problem.

Step 2 - Prompt writing

Based on the image you have chosen, create a coding problem that adheres to the following criteria:

1. **Programming Language Alignment**: The problem should be solvable using one of the programming languages that align with the content or context of the image.
2. **Image Dependency**: The problem must explicitly require information from the selected image to solve. The solution should depend on analyzing or extracting data, patterns, or insights from the image.
3. **Problem Clarity**: Clearly state the objective, constraints, and expected output of the coding task.

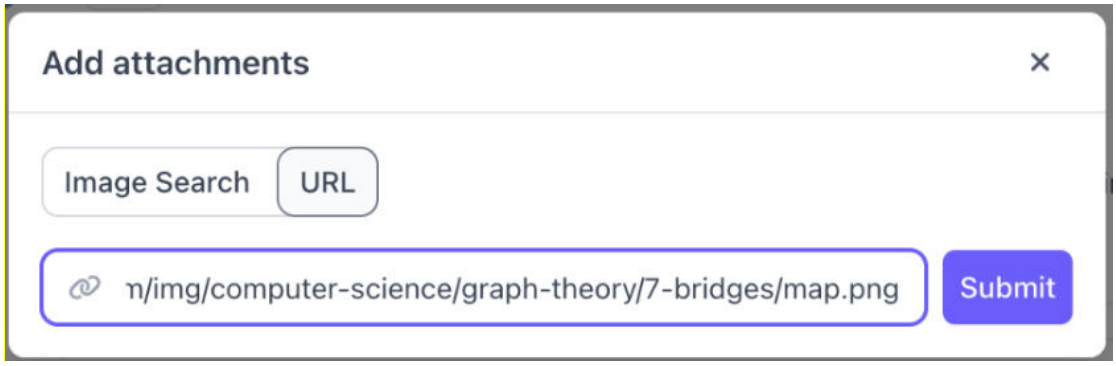
Step 3 - Rating the model response

Evaluate each response based on the dimensions and provide short justifications to support each rating:

1. **Evaluate Across Dimensions:** Rate responses on
    a. Image Understanding (1–3)
        i. ⚠️ The principal objective of your prompt is to make the model fail in this dimension, if your rating is 3, your task will be rejected (it did not make the model fail in the most important part).
    b. Image-Independent Reasoning (1–3)
    c. Overall Satisfaction (1–5).
2. **Provide Justifications**: Include a brief explanation for each score based on specific aspects of the response, be clear and descriptive of the defects that justify your ratings.

Step 1: Image Selection and Upload

In this step, your job is to upload a high-quality coding-related image that can test the model's visual understanding skills. The image you choose should be challenging enough for the AI model to learn from.



SPECIFICATIONS

Most importantly, you should be able to ask a coding-related question about the image that requires the model to think critically about the image.

Image Relevance:

Select an image you find interesting and familiar. Your connection to the image will help you think of challenging prompts and be able to assess whether the model answered correctly. ‼️REMEMBER: The prompt MUST be image-dependent, which means that the model should only be able to answer the prompt if it has the information in the image you provide.

⚠ The image should incorporate a coding-related aspect that is essential to solving the prompt you provide, such as:

Code snippets, documents, screenshots, graphs, charts and others.

Quality Standards:

Make sure the image is clear and under 1MB. High resolution is important to ensure the model can identify details accurately.

Diversity:

Use a variety of image types and themes to keep the tasks dynamic. Avoid repeating similar content and explore different subjects.

Appropriateness:

Ensure your image does not contain any sensitive information (like PII) or inappropriate content.

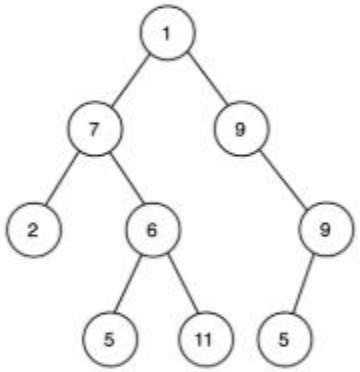
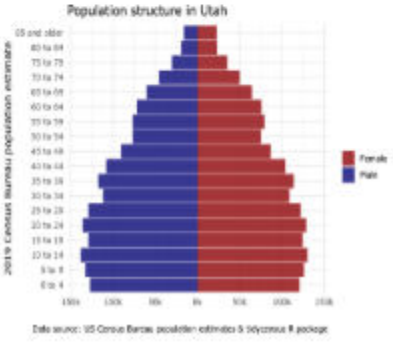
Choose your image wisely, as it will determine the quality of the prompts and the model's ability to respond effectively.

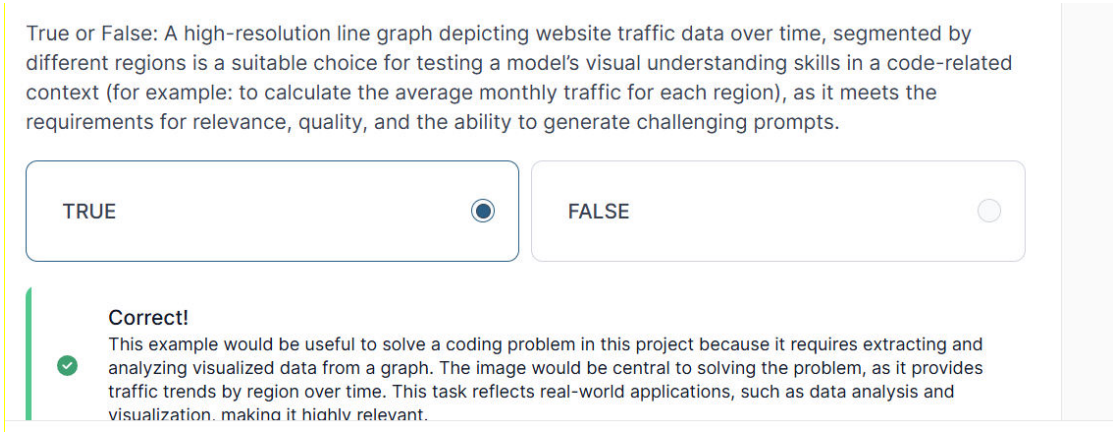
We'll cover more examples in the next page!

Image examples:

Image type	Example

Screenshots (code snippets)
 	
Images that contain numeric data or code snippets that your coding problem can be answered with.
 	
For example, the image may contain a partially implemented algorithm or an error to fix.
 	
Examples:
 	
• An incomplete function that needs to be finished.
• A snippet with syntax or logic errors to be identified and corrected.
• A code block with missing comments requiring documentation.
	
	package studying
	
	object Fibonacci {
	def main(args: Array[String]): Unit = {
	
	println(fibonacci(10))
	
	}
	
	def fibonacci(n:Int):Int={
	if(n==0)
	return 0
	
	if(n==1)
	return 1
	
	else
	return fibonacci(n-1)+fibonacci(n-2)
	
	}
	}

Flowcharts or Diagrams Images of flowcharts, system architecture diagrams, or UML (Unified Modeling Language) diagrams. These images provide visual representations of algorithms, workflows, or system designs that users can translate into functional code. **Examples**: • A flowchart representing a sorting algorithm (e.g., bubble sort). • A UML diagram showing the relationships between classes for object-oriented programming. • A database schema that needs to be implemented in Python.	 1 7 9 2 6 9 5 11 5
Graphs and Tables Images containing structured data (tables) or visualized data (graphs). These can be used to extract data from these images to write code that processes, analyzes, or visualizes the data. **Examples**: • A table of sales data that requires generating a Python script to calculate totals or create visualizations. • A bar graph representing population growth, used to write a script for regression analysis. • A CSV-like table needing conversion into a specific format (e.g., JSON).	 Population structure in Utah Female Male



Step 2: Prompt Writing

In this step, write a prompt based on the image you've selected.

✳The goal is to create a thoughtful, challenging prompt that encourages the model to analyze and reason based on the image.

⚠REMEMBER:

Your prompt has to be image-dependent which means it SHOULD not be possible to answer without the image.

Your prompt HAS to make the model reason and analyze based on your image, the prompt should not only ask the model to "find" data if this does not include reasoning.

PROMPT GUIDANCE

☆ High quality prompts (DOs)

Encourage Complexity: Design prompts that challenge the model's ability to understand the image (IU - image understanding). If the model answers correctly without error, increase the complexity of the prompt.

Image-dependency: Your prompt SHOULD not be possible to answer without the image

Require Critical Thinking: Ensure the prompt requires the model to reason and apply logic based on the information within the image.

Be Creative and Diverse: Use a variety of formats for prompts to avoid repetition and ensure diversity in question types.

Sound Natural: Write prompts in a clear and conversational tone to ensure they are easy to understand.

✗ Low quality prompts (DONTs)

Avoid Standalone Answers: Prompts should not be answerable without referencing the image.
Avoid Over-Simplification: Ensure the image's content alone isn't enough to answer without additional reasoning.
Stay Within Your Expertise: Do not ask about unfamiliar topics, as this may compromise prompt quality and relevance.
Avoid Unverifiable Queries: Do not ask questions about information or situations that cannot be easily verified or checked.

Errors we commonly see in this project:

Not thoroughly testing code generated by the models.
Solution: Test each part of the code to ensure it runs correctly and meets your requirements.
Tips for Avoiding this Error:

Write Clear and Testable Prompts
Create prompts that result in outputs you can easily test and review.
Use Tools Like Replit
Ask the model to generate code that runs directly on platforms like Replit.
Replit supports many libraries and can automatically set up testing environments.

Minimize External Dependencies
Avoid requiring libraries or databases that are difficult to set up unless you provide clear instructions or alternative solutions.

Image understanding (IU) and Image independent Reasoning (IIR). Some useful definitions:

- **Image Understanding (IU):**
  - Refers to the model's ability to correctly interpret or extract the necessary information from the image.
  - Focus is on whether the visual data has been correctly processed.

⚠️Your prompt has to be able to make the model FAIL in this category, you will (in next steps) rate the model regarding IU and justify your rating.

- **Image-Independent Reasoning (IIR):**
  - Refers to the reasoning or logic the model applies based on what it interprets from the image.

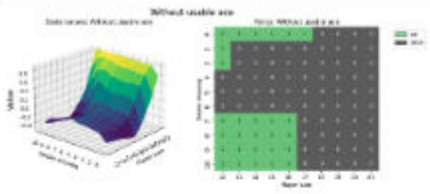
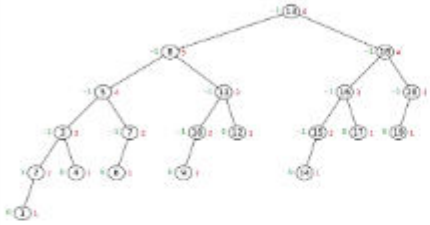
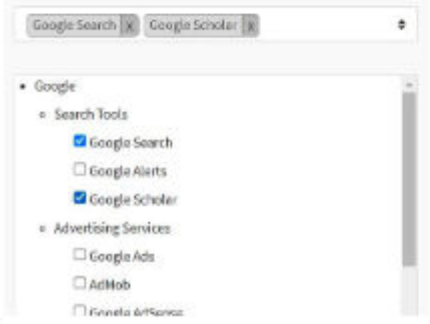
PROGRAMMING LANGUAGES:

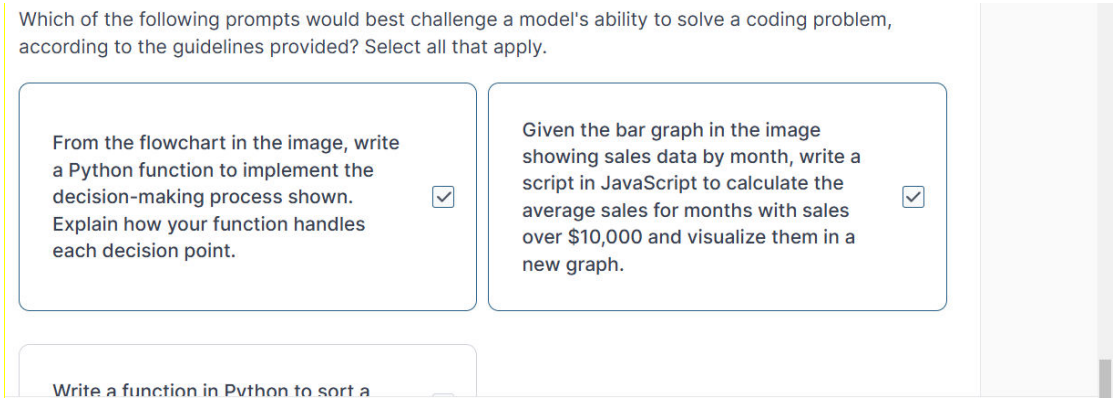
Available coding languages
Python
Java
C
C++
C#
CSS
Go
HTML
Kotlin
Objective-C
PHP
AppleScript
JavaScript
Ruby
Rust
Scala
Shell > Bash
Shell > Zsh
Swift

🚨 Important Instructions for Coding Problems 🚨

1. **Ensure the Model Fails in Image Understanding (IU)**:
   - ✅ Your coding problem is valid **only if the model response scores a '1' or '2'** in Image Understanding, indicating a **Major or Minor Issue**.
   - ❌ If the response rates as a **'3' (No Issues)** in Image Understanding, the problem is too simple and you NEED to submit another prompt that makes the model fail in this step.
2. **Revise Your Prompt for Complexity**:
   - 🔄 If the model scores a '3' in Image Understanding, you **must revise your prompt** to make it more challenging and image-dependent.

💡 **Tip**: Focus on creating problems that require the model to deeply interpret the image for an accurate response.

Image	Prompt
	What does the image depict here? What approach should the model take to reproduce the results in this image? Write python code without using any external ML libraries except the basic ones like np and the ones for arrays to reproduce the graphs.
	Explain the image data structure to me. Implement it in C++.
	Analyze the function defined in the code provided in the image. Identify the specific approach used to reverse the string and explain why the order of operations inside the loop produces the desired result. Propose a modification to the code that makes it more Pythonic while achieving the same functionality. Justify why the suggested change improves readability , using examples from the code shown
	I want you to create this component in JavaScript with proper functionalities working. Users should be able to select values and selected values will be visible to users. Design should be exactly like the attached image.



Step 3: Rating the Model Response

In Step 3, you will evaluate the model's responses based on two key dimensions:

Image Understanding (Rating Score of 1-3)

Image-Independent Reasoning (Rating Score 1-3)

You will then rate your overall satisfaction based on the two dimensions above.

Overall Satisfaction (Rating Score 1-5)

The goal is to assess the quality of each response, making sure it aligns with the task's expectations.

Model Response Rating Dimensions

Criteria	1 - Not at All	2 - Partial	3 - Completely
Image Understanding The degree to which the response accurately comprehends, interprets, and engages with the content within the image.	**Visual perception:** The response does not extract correct visual information from the image. **Visual reasoning:** The response makes mistakes in interpreting spatial relationships and visual data (such as charts, graphs, diagrams, geometric figures, etc.).	**Visual perception:** The response extracts most correct visual information from the image. Any mistakes in extracting visual information could be easily corrected to enable the prompt to be 'on track' to fulfilling the prompt. **Visual reasoning:** The response makes minor mistakes in interpreting spatial relationships and visual data.	**Visual perception:** The response makes no errors in extracting visual information from the image. **Visual reasoning:** The response makes no mistakes in interpreting spatial relationships and visual data.
Image-Independent Reasoning The degree to which the model accurately builds upon visual understanding and reasoning to deliver a correct final answer. **Note:** This dimension evaluates how the response processes the information that it extracted from the image, NOT whether the context extracted is correct.	The response contains logical reasoning errors or makes severe omissions in its reasoning steps when applying the visual information extracted from the image.	The response contains minor logical reasoning errors or makes minor omissions in its reasoning steps. Alternatively, the response reasoning is fully correct and there are minor calculation errors.	The response contains fully logical and correct reasoning.

REMEMBER: Your prompt has to make the model FAIL in the image understanding category (IU) if this does NOT happen, then you have to try out a new, more challenging prompt.

Dimension	1 - Horrible	2 - Bad	3 - Okay	4 - Good	5 - Perfect
Overall Satisfaction	The response is terrible. Relative to the prompt /request, the response is almost useless. It makes several mistakes or at least one egregious mistake. This includes serious misinterpretations of the prompt. A user would be severely disappointed with this response	The response makes some mistakes or a single medium mistake. Overall, a user would be dissatisfied with the response, but it is intelligible and going in the right direction.	The response is okay. It might border on being satisfying or not. There may be one or two small mistakes. However, the response is free of egregious, serious, or major errors such as instruction following, image understanding, or truthfulness mistakes.	The response is solid. It is generally free of mistakes, but allows for very minor mistakes at worst. A user would be satisfied with this response.	The response is amazing! It is entirely error free. The user would be delighted to read this response after submitting the image and prompt.

There will be a box for writing a brief justification for each dimension. This justification has to point out clearly the reasons of why you rated the response this way.

Guidelines for Writing Justifications:

Be Specific: Focus on one key aspect of the response and explain why it meets or fail

REMEMBER:

- Your coding problem only qualifies when the response rates as a 'fail' in Image Understanding (scored as a 1 or 2, indicating a Major or Minor Issue).
- If Image Understanding is a '3' (No Issues), you will need to revise your prompt to make it more complex.

Step 4: Rewriting the Model Response (WHEN THE TASK ASKS FOR)

If the task indicates it, you will additionally need to rewrite the response to help the model learn what the optimal response should be.

If in the previous step you rated Satisfaction 1 (Horrible), you will additionally rewrite the response to help the model learn what the optimal response should be.

Warning: Your tasks will be rejected if you incorrectly score a response a 2 instead of 1 to avoid having to do the rewrite.

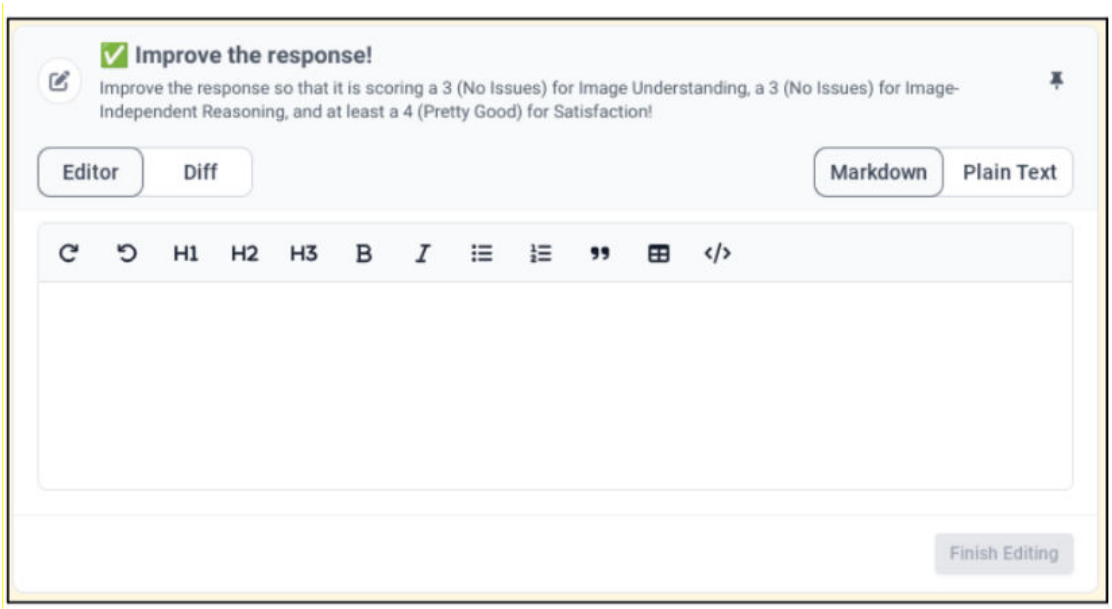
In this step, you will rewrite the response to help the model learn what the optimal response should be.

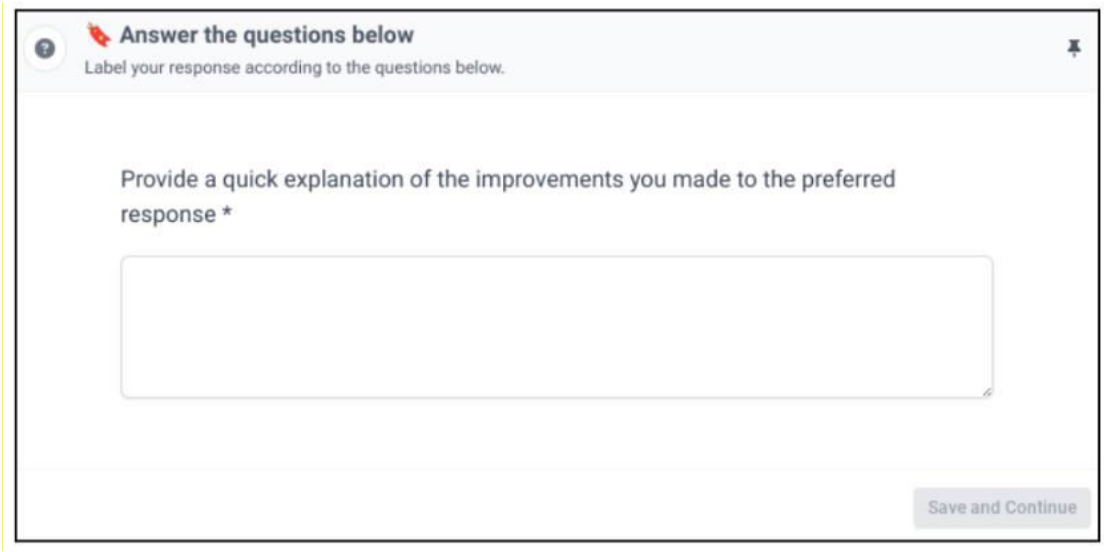
The rewrite should improve the response to a 3 rating (No Issues) across all dimensions and the overall satisfaction score to a 5.

The rewritten response should be perfect!

Most importantly, code in the response must compile and execute!s to meet expectations.
Use Examples: Reference specific details from the response to support your rating.
Keep it Concise: Avoid lengthy explanations; one full sentence should be enough to justify your rating.

Clarity in Rewriting	• Rewrite the step clearly and concisely, using simple language that avoids jargon. • The rewrite should be self-contained and understandable, following a logical sequence of reasoning.
Focus on Problem-Solving	• Only include information necessary to solve the problem. • Do not add extraneous details like definitions of basic concepts.





IMPORTANT!

🔒Copying and Pasting from an LLM is strictly prohibited

and can result in you being removed from the project!🔒

REMEMBER

Step 1 - Image selection

In this step you need to select an image using the file selector tool. You can either input your own image link or browse and choose an image directly within the tool. When making your selection, it is crucial to consider the following guidelines:

1. **Code-Related Image**: Ensure the image is relevant to coding or programming.
2. **Prompt Dependency**: The image should provide information essential for constructing your prompt. The prompt must rely on details from the image to be fully addressed.
3. **Programming Language Alignment**: Choose an image that aligns with one of the available programming languages and fits the context of your coding problem.

Step 2 - Prompt writing

Based on the image you have chosen, create a coding problem that adheres to the following criteria:

1. **Programming Language Alignment**: The problem should be solvable using one of the programming languages that align with the content or context of the image.
2. **Image Dependency**: The problem must explicitly require information from the selected image to solve. The solution should depend on analyzing or extracting data, patterns, or insights from the image.
3. **Problem Clarity**: Clearly state the objective, constraints, and expected output of the coding task.

Step 3 - Rating the model response

Evaluate each response based on the dimensions and provide short justifications to support each rating:

1. **Evaluate Across Dimensions:** Rate responses on
    a. Image Understanding (1–3)
        i. ⚠️ The principal objective of your prompt is to make the model fail in this dimension, if your rating is 3, your task will be rejected (it did not make the model fail in the most important part).
    b. Image-Independent Reasoning (1–3)
    c. Overall Satisfaction (1–5).
2. **Provide Justifications**: Include a brief explanation for each score based on specific aspects of the response, be clear and descriptive of the defects that justify your ratings.

High Quality Prompts Should:

- Be complex enough to elicit a model failure in Image Understanding
  - If the model doesn't make a mistake, submit a more complex prompt for your image
- Require the model to think critically and apply reasoning to the information within the image
- Reflect a creative diverse set of formats that aren't repeated
- Sound natural

High Quality Prompts Should NOT:

- Be answerable without the image
- Be easily answerable by the image's content only
- Ask about topics with which you're not familiar
- Ask for things that are difficult to verify

🏆Key Differences

Image Understanding (IU):

Refers to the model's ability to correctly interpret or extract the necessary information from the image. Focus is on whether the visual data has been correctly processed.

🚨Important: If you rate IU as a 3, it means the task did not cause the model to fail. In this case, you must revisit the prompt and make adjustments to ensure the model fails. This is imperative.🚨

Image-Independent Reasoning (IIR):

Refers to the reasoning or logic the model applies based on what it interprets from the image.

Focus is on whether the reasoning is valid, given the model's understanding of the image.
IIR evaluates whether the reasoning and interpretation based on the image are logical and accurate, even if the initial image understanding was incomplete.

Example

Prompt: "From the image, calculate the result of 2+2+2."

Image Content: Shows "2 + 2 + 2".
Model Response: "The result of 2+2 is 4."
Analysis:

⛔IU: 1/2 (Fail) – The model misinterpreted the image, failing to account for the last +2.
✅IIR: 3 (Pass) – Based on its (partial) understanding of "2+2," the reasoning is correct because 2 + 2 equals 4.

🚨 **Important Instructions for Coding Problems** 🚨

1. **Ensure the Model Fails in Image Understanding (IU)**:
   - ✅ Your coding problem is valid **only if the model response scores a '1' or '2'** in Image Understanding, indicating a **Major or Minor Issue**.
   - ❌ If the response rates as a **'3' (No Issues)** in Image Understanding, the problem is too simple and you NEED to submit another prompt that makes the model fail in this step.
2. **Revise Your Prompt for Complexity**:
   - 🔄 If the model scores a '3' in Image Understanding, you **must revise your prompt** to make it more challenging and image-dependent.

💡 **Tip**: Focus on creating problems that require the model to deeply interpret the image for an accurate response.