Infosys
Responsible AI Office

Infosys Responsible AI Toolkit – Moderation Layer

API Usage Instructions

Contents

About Moderation Layer

The Moderation Layer offers a suite of APIs to ensure the safety and reliability of LLMs. These APIs can be used to moderate both input prompts and generated responses, checking for privacy violations, security risks, hallucination and biases. Additionally, the layer provides explainability features, such as Thread of Thought, Chain of Thought, and Chain of Verification, to enhance transparency and understanding of LLM reasoning.

Once API swagger page is populated as per instructions given in the github repository Readme file, click on 'try it out' to use required endpoints. Details of endpoints associated with moderation layer are outlined below.

Dependencies

Moderation, Coupled Moderation, Toxicity Popup, Profanity Popup, Privacy Popup, and OrgPolicy all endpoints in Moderation Layer depend on the Moderation model module. These features require the model repository to function correctly. In addition, getTemplates API depends on the Admin module. Please ensure both the Moderation model and Admin module are properly configured and running before using these components, refer to their respective readMe files for the setup steps.

Features & API End Points

Our system supports multiple Large Language Models (LLMs) to provide versatile AI capabilities. The supported models include GPT-4o-mini, GPT-3.5 Turbo, AWS Anthropic Bedrock Claude model, LLaMA 3 -70B, Gemini 2.5 Flash, and Gemini 2.5 Pro. Please ensure that any one of these models is available and correctly integrated to utilize the full functionality of our platform.

Moderation APIs

Moderation

Endpoint: /rai/v1/moderations
This API provides the decoupled guardrail (checks for the prompt like – privacy check, prompt injection check, jailbreak check, toxicity check, restricted topic, custom theme check; along with some other optional checks like gibberish check, invisible text check, ban code check, sentiment check).

Input: In input Json we need to replace prompt value with the text we want to be moderated. If we want emoji to be moderated as well then give emoji moderation as yes otherwise no. In moderation checks we can list the checks we want our text to undergo, in 'moderation checks threshold' we need to pass the threshold for the checks we included.

```
{
  "AccountName": "None",
  "userid": "None",
  "Source": "chatgpt",
  "PortfolioName": "None",
  "lotNumber": 1,
  "translate": "no",
  "EmojiModeration": "yes",
  "token_env": "others",
  "Prompt": "Which is the biggest country in the world?",
  "ModerationChecks": [
    "PromptInjection",
    "JailBreak",
    "Toxicity",
    "Piidetct",
    "Refusal",
    "Profanity",
    "RestrictTopic",
    "TextQuality",
    "CustomizedTheme",
    "Sentiment",
    "InvisibleText",
    "Gibberish",
    "BanCode"
  ],
  "ModerationCheckThresholds": {
    "PromptinjectionThreshold": 0.7,
    "JailbreakThreshold": 0.7,
    "PiientitiesConfiguredToBlock": [
      "AADHAR_NUMBER",
      "PAN_Number",
      "IN_PAN",
      "US_PASSPORT",
      "US_SSN"
    ],
    "RefusalThreshold": 0.7,
    "ToxicityThresholds": {
      "ToxicityThreshold": 0.6,
      "SevereToxicityThreshold": 0.6,
      "ObsceneThreshold": 0.6,
      "ThreatThreshold": 0.6,
      "InsultThreshold": 0.6,
      "IdentityAttackThreshold": 0.6,
      "SexualExplicitThreshold": 0.6
    },
    "ProfanityCountThreshold": 1,
    "RestrictedtopicDetails": {
      "RestrictedtopicThreshold": 0.7,
      "Restrictedtopics": [
        "Terrorism",
        "Explosives"
      ]
    },
    "CustomTheme": {
      "Themename": "string",
      "Themethresold": 0.6,
      "ThemeTexts": [
        "Text1",
        "Text2",
        "Text3"
      ]
    },
    "SentimentThreshold": -0.01,
    "InvisibleTextCountDetails": {
      "BannedCategories": [
        "Cf",
        "Co",
        "Cn",
        "So",
        "Sc"
      ]
    },
    "GibberishDetails": {
      "GibberishThreshold": 0.7,
      "GibberishLabels": [
        "word salad",
        "noise",
        "mild gibberish",
        "clean"
      ]
    }
  }
}
```


Execute

Response: Returns detailed moderation analysis with PASSED/FAILED results for each security check, including timing metrics, confidence scores, and detected content classifications

Response body


```
{
  "Moderation layer time": {
    "Time for each individual check": {
      "Ban Code Check": "0.069s",
      "Custom Theme Check": "0.296s",
      "Gibberish Check": "0.108s",
      "Invisible Text Check": "0.102s",
      "Jailbreak Check": "0.294s",
      "Privacy Check": "0.123s",
      "Profanity Check": "0.174s",
      "Prompt Injection Check": "0.232s",
      "Refusal Check": "0.295s",
      "Restricted Topic Check": "0.25s",
      "Sentiment Check": "0.072s",
      "Text Quality Check": "0.0s",
      "Toxicity Check": "0.173s"
    },
    "Time taken by each model": {
      "Ban Code Check": "0.019s",
      "Custom Theme Check": "0.141s",
      "Gibberish Check": "0.063s",
      "Invisible Text Check": "0.0s",
      "Jailbreak Check": "0.141s",
      "Privacy Check": "0.046s",
      "Prompt Injection Check": "0.114s",
      "Restricted Topic Check": "0.179s",
      "Sentiment Check": "0.006s",
      "Toxicity Check": "0.036s"
    },
    "Total time for moderation Check": "0.307s"
  },
  "Source": "chatgpt",
  "accountName": "None",
  "created": "2025-06-13 08:54:49.578694",
  "lotNumber": "1",
  "moderationResults": {
    "bancodeCheck": {
      "label": "NL",
      "result": "PASSED"
    },
    "customThemeCheck": {
      "customSimilarityScore": "0.29",
      "result": "PASSED",
      "themeThreshold": "0.6"
    },
    "gibberishCheck": {
      "gibberishScore": [
        {
          "gibberish_label": "clean",
          "gibberish_score": 0.97
        }
      ],
      "result": "PASSED",
      "threshold": "0.7"
```

```
  },
  "invisibleTextCheck": {
    "invisibleTextIdentified": [],
    "result": "PASSED"
  },
  "jailbreakCheck": {
    "jailbreakSimilarityScore": "0.49",
    "jailbreakThreshold": "0.7",
    "result": "PASSED"
  },
  "privacyCheck": {
    "entitiesConfiguredToBlock": [
      "IN_AADHAAR",
      "IN_PAN",
      "IN_PAN",
      "US_PASSPORT",
      "US_SSN"
    ],
    "entitiesRecognised": [],
    "result": "PASSED"
  },
  "profanityCheck": {
    "profaneWordsIdentified": [],
    "profaneWordsthreshold": "1",
    "result": "PASSED"
  },
  "promptInjectionCheck": {
    "injectionConfidenceScore": "0.0",
    "injectionThreshold": "0.7",
    "result": "PASSED"
  },
  "refusalCheck": {
    "RefusalThreshold": "0.7",
    "refusalSimilarityScore": "0.41",
    "result": "PASSED"
  },
  "restrictedtopic": {
    "result": "PASSED",
    "topicScores": [
      {
        "Explosives": "0.0",
        "Terrorism": "0.0"
      }
    ],
    "topicThreshold": "0.7"
  },
  "sentimentCheck": {
    "result": "PASSED",
    "score": "0.0",
    "threshold": "-0.01"
  },
  "summary": {
    "reason": [],
    "status": "PASSED"
```

```
  },
  "text": "Which is the biggest country in the world?",
  "textQuality": {
    "readabilityScore": "92",
    "textGrade": "2nd and 3rd grade"
  },
  "toxicityCheck": {
    "result": "PASSED",
    "toxicityScore": [
      {
        "toxicScore": [
          {
            "metricName": "toxicity",
            "metricScore": 0.001
          },
          {
            "metricName": "severe_toxicity",
            "metricScore": 0
          },
          {
            "metricName": "obscene",
            "metricScore": 0
          },
          {
            "metricName": "threat",
            "metricScore": 0
          },
          {
            "metricName": "identity_attack",
            "metricScore": 0
          },
          {
            "metricName": "sexual_explicit",
            "metricScore": 0
          }
        ]
      }
    ],
    "toxicitythreshold": "0.6"
  }
  },
  "portfolioName": "None",
  "uniqueid": "95431cbef2da49509e071b0110686497",
  "userid": "None"
}
```

Coupled Moderation

Endpoint – /rai/v1/moderations/coupledmoderations
This API provides the coupled guardrail (provides checks for input prompt, LLM interaction for generating response and checks for response)

Input: In input json we need to replace prompt value with the text we want to be moderated. If we want emoji to be moderated as well then give emoji moderation as yes otherwise no. In 'input moderation checks' we can list the checks we want our input text to undergo, in 'output moderation checks' we can list the checks we want our response to undergo and in 'moderation checks threshold' we need to pass the threshold for the checks we included.

Name	Description
authorization **string** *(header)*	authorization

Request body required

```
{
  "AccountName": "None",
  "PortfolioName": "None",
  "userid": "None",
  "lotNumber": 1,
  "model_name": "gpt4",
  "translate": "no",
  "temperature": "0",
  "LLMinteraction": "yes",
  "PromptTemplate": "GoalPriority",
  "EmojiModeration": "yes",
  "Prompt": "Which is the biggest country in the world?",
  "InputModerationChecks": [
    "PromptInjection",
    "JailBreak",
    "Toxicity",
    "Piidetct",
    "Refusal",
    "Profanity",
    "RestrictTopic",
    "TextQuality",
    "CustomizedTheme",
    "Sentiment",
    "InvisibleText",
    "Gibberish",
    "BanCode"
  ],
```

```
  "OutputModerationChecks": [
    "Toxicity",
    "Piidetct",
    "Refusal",
    "Profanity",
    "RestrictTopic",
    "TextQuality",
    "TextRelevance",
    "Sentiment",
    "InvisibleText",
    "Gibberish",
    "BanCode"
  ],
  "llm_BasedChecks": [
    "randomNoiseCheck",
    "advancedJailbreakCheck"
  ],
  "ModerationCheckThresholds": {
    "PromptinjectionThreshold": 0.7,
    "JailbreakThreshold": 0.7,
    "PiientitiesConfiguredToBlock": [
      "AADHAR_NUMBER",
      "PAN_Number",
      "IN_PAN",
      "US_PASSPORT",
      "US_SSN"
    ],
    "RefusalThreshold": 0.7,
    "ToxicityThresholds": {
      "ToxicityThreshold": 0.6,
      "SevereToxicityThreshold": 0.6,
      "ObsceneThreshold": 0.6,
      "ThreatThreshold": 0.6,
      "InsultThreshold": 0.6,
      "IdentityAttackThreshold": 0.6,
      "SexualExplicitThreshold": 0.6
    },
    "ProfanityCountThreshold": 1,
    "RestrictedtopicDetails": {
      "RestrictedtopicThreshold": 0.7,
      "Restrictedtopics": [
        "Terrorism",
        "Explosives"
      ]
    },
    "CustomTheme": {
      "Themename": "string",
      "Themethresold": 0.6,
      "ThemeTexts": [
        "Text1",
        "Text2",
        "Text3"
      ]
    },
    "SmoothLlmThreshold": {
      "input_pertubation": 0.1,
      "number_of_iteration": 4,
      "SmoothLlmThreshold": 0.6
    },
```

```
    "SentimentThreshold": -0.01,
    "InvisibleTextCountDetails": {
      "BannedCategories": [
        "Cf",
        "Co",
        "Cn",
        "So",
        "Sc"
      ]
    },
    "GibberishDetails": {
      "GibberishThreshold": 0.7,
      "GibberishLabels": [
        "word salad",
        "noise",
        "mild gibberish",
        "clean"
      ]
    }
  }
}
```


Execute

Response: Returns complete moderation pipeline results with timing metrics, input/output validation status, generated LLM response, and detailed security check results for both request and response content.

Response body


```
{
  "Moderation layer time": {
    "Llama3InteractionTime": "10.282s",
    "OpenAIInteractionTime": "0.41s",
    "Time taken by each model in requestModeration": {
      "Ban Code Check": "0.026s",
      "Custom Theme Check": "0.116s",
      "Gibberish Check": "0.053s",
      "Invisible Text Check": "0.001s",
      "Jailbreak Check": "0.116s",
      "Privacy Check": "0.051s",
      "Prompt Injection Check": "0.121s",
      "Restricted Topic Check": "0.152s",
      "Sentiment Check": "0.005s",
      "Toxicity Check": "0.051s"
    },
    "Time taken by each model in responseModeration": {
      "Ban Code Check": "0.015s",
      "Gibberish Check": "0.059s",
      "Invisible Text Check": "0.0s",
      "Privacy Check": "0.042s",
      "Restricted Topic Check": "0.186s",
      "Sentiment Check": "0.005s",
      "Toxicity Check": "0.08s"
    },
```

```
  "Total time for moderation Check": "3.661s",
  "requestModeration": {
    "Advanced Jailbreak Check": "1.068s",
    "Ban Code Check": "2.719s",
    "Custom Theme Check": "2.944s",
    "Gibberish Check": "2.754s",
    "Invisible Text Check": "2.713s",
    "Jailbreak Check": "2.941s",
    "Privacy Check": "2.782s",
    "Profanity Check": "2.821s",
    "Prompt Injection Check": "2.875s",
    "Random Noise Check": "1.578s",
    "Refusal Check": "2.942s",
    "Restricted Topic Check": "2.889s",
    "Sentiment Check": "2.727s",
    "Text Quality Check": "0.0s",
    "Text Relevance Check": "0.0s",
    "Toxicity Check": "2.82s"
  },
  "responseModeration": {
    "Advanced Jailbreak Check": "0s",
    "Ban Code Check": "0.118s",
    "Custom Theme Check": "0s",
    "Gibberish Check": "0.11s",
    "Invisible Text Check": "0.057s",
    "Jailbreak Check": "0s",
      "Privacy Check": "0.142s",
      "Profanity Check": "0.187s",
      "Prompt Injection Check": "0s",
      "Random Noise Check": "0s",
      "Refusal Check": "0.276s",
      "Restricted Topic Check": "0.27s",
      "Sentiment Check": "0.062s",
      "Text Quality Check": "0.0s",
      "Text Relevance Check": "0.227s",
      "Toxicity Check": "0.187s"
    },
    "translate": "0.0s"
  },
    "gibberishCheck": {
      "gibberishScore": [
        {
          "gibberish_label": "clean",
          "gibberish_score": 0.97
        }
      ],
      "result": "PASSED",
      "threshold": "0.7"
    },
    "invisibleTextCheck": {
      "invisibleTextIdentified": [],
      "result": "PASSED"
    },
    "jailbreakCheck": {
      "jailbreakSimilarityScore": "0.49",
      "jailbreakThreshold": "0.7",
      "result": "PASSED"
    },
    "privacyCheck": {
      "entitiesConfiguredToBlock": [
        "IN_AADHAAR",
        "IN_PAN",
        "IN_PAN",
        "US_PASSPORT",
        "US_SSN"
      ],
```

```
"accountName": "None",
"choices": [
  {
    "finishReason": "stop",
    "index": 0,
    "text": "The biggest country in the world is Russia. "
  }
],
"created": "2025-06-13 09:15:59.166290",
"lotNumber": "1",
"model": "gpt4",
"moderationResults": {
  "requestModeration": {
    "advancedJailbreakCheck": {
      "result": "PASSED",
      "text": "NON ADVERSARIAL"
    },
    "bancodeCheck": {
      "label": "NL",
      "result": "PASSED"
    },
    "customThemeCheck": {
      "customSimilarityScore": "0.29",
      "result": "PASSED",
      "themeThreshold": "0.6"
    },
```

```
    ],
    "topicTypesRecognised": []
  },
  "sentimentCheck": {
    "result": "PASSED",
    "score": "0.0",
    "threshold": "-0.01"
  },
  "summary": {
    "reason": [],
    "status": "PASSED"
  },
  "text": "Which is the biggest country in the world?",
  "textQuality": {
    "readabilityScore": "92",
    "textGrade": "2nd and 3rd grade"
  },
  "toxicityCheck": {
    "result": "PASSED",
    "toxicityScore": [
      {
        "toxicScore": [
          {
            "metricName": "toxicity",
            "metricScore": 0.001
          },
```

```
    "promptInjectionCheck": {
      "injectionConfidenceScore": "0.0",
      "injectionThreshold": "0.7",
      "result": "PASSED"
    },
    "randomNoiseCheck": {
      "result": "PASSED",
      "smoothLlmScore": "0.0",
      "smoothLlmThreshold": "0.6"
    },
    "refusalCheck": {
      "RefusalThreshold": "0.7",
      "refusalSimilarityScore": "0.41",
      "result": "PASSED"
    },
    "restrictedtopic": {
      "result": "PASSED",
      "topicScores": [
        {
          "Explosives": "0.0",
          "Terrorism": "0.0"
        }
      ],
      "topicThreshold": "0.7",
      "topicTypesConfiguredToBlock": [
        "Terrorism",
        "Explosives"
```

```
      "entitiesRecognised": [],
      "result": "PASSED"
    },
    "profanityCheck": {
      "profaneWordsIdentified": [],
      "profaneWordsthreshold": "1",
      "result": "PASSED"
    },
```

```
      "toxicityTypesConfiguredToBlock": [
        "toxicity",
        "severe_toxicity",
        "obscene",
        "threat",
        "insult",
        "identity_attack",
        "sexual_explicit"
      ],
      "toxicityTypesRecognised": [],
      "toxicitythreshold": "0.6"
    }
  },
  "responseModeration": {
    "bancodeCheck": {
      "label": "NL",
      "result": "PASSED"
    },
    "generatedText": "The biggest country in the world is Russia. ",
    "gibberishCheck": {
      "gibberishScore": [
        {
          "gibberish_label": "clean",
          "gibberish_score": 0.97
        }
      ],
      "result": "PASSED",
```

```
        {
          "metricName": "severe_toxicity",
          "metricScore": 0
        },
        {
          "metricName": "obscene",
          "metricScore": 0
        },
        {
          "metricName": "threat",
          "metricScore": 0
        },
        {
          "metricName": "insult",
          "metricScore": 0
        },
        {
          "metricName": "identity_attack",
          "metricScore": 0
        },
        {
          "metricName": "sexual_explicit",
          "metricScore": 0
        }
      ]
    }
  ],
```

```
    "threshold": "0.7"
  },
  "hallucinationScore": "0",
  "invisibleTextCheck": {
    "invisibleTextIdentified": [],
    "result": "PASSED"
  },
  "privacyCheck": {
    "entitiesConfiguredToBlock": [
      "IN_AADHAAR",
      "IN_PAN",
      "IN_PAN",
      "US_PASSPORT",
      "US_SSN"
    ],
    "entitiesRecognised": [],
    "result": "PASSED"
  },
  "profanityCheck": {
    "profaneWordsIdentified": [],
    "profaneWordsthreshold": "1",
    "result": "PASSED"
  },
  "refusalCheck": {
    "RefusalThreshold": "0.7",
    "refusalSimilarityScore": "0.41",
    "result": "PASSED"
```

```
  },
  "restrictedtopic": {
    "result": "PASSED",
    "topicScores": [
      {
        "Explosives": "0.0",
        "Terrorism": "0.0"
      }
    ],
    "topicThreshold": "0.7",
    "topicTypesConfiguredToBlock": [
      "Terrorism",
      "Explosives"
    ],
    "topicTypesRecognised": []
  },
  "sentimentCheck": {
    "result": "PASSED",
    "score": "0.0",
    "threshold": "-0.01"
  },
  "summary": {
    "reason": [],
    "status": "PASSED"
  },
  "textQuality": {
    "readabilityScore": "92",
    "textGrade": "2nd and 3rd grade"
  },
  "textRelevanceCheck": {
    "PromptResponseSimilarityScore": "99"
  },
  "toxicityCheck": {
    "result": "PASSED",
    "toxicityScore": [
      {
        "toxicScore": [
          {
            "metricName": "toxicity",
            "metricScore": 0.001
          },
          {
            "metricName": "severe_toxicity",
            "metricScore": 0
          },
          {
            "metricName": "obscene",
            "metricScore": 0
          },
          {
            "metricName": "threat",
            "metricScore": 0
          },
```

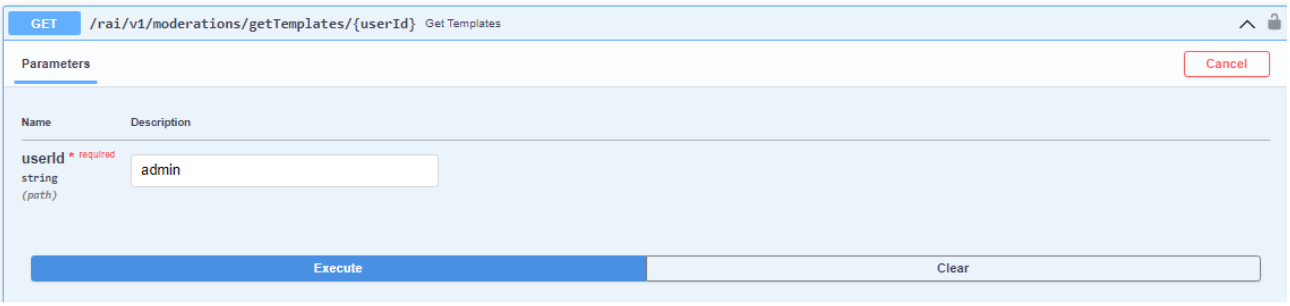
```
          {
            "metricName": "insult",
            "metricScore": 0
          },
          {
            "metricName": "identity_attack",
            "metricScore": 0
          },
          {
            "metricName": "sexual_explicit",
            "metricScore": 0
          }
        ]
      }
    ],
    "toxicityTypesConfiguredToBlock": [
      "toxicity",
      "severe_toxicity",
      "obscene",
      "threat",
      "insult",
      "identity_attack",
      "sexual_explicit"
    ],
    "toxicityTypesRecognised": [],
    "toxicitythreshold": "0.6"
  }
```

```
    }
  },
  "object": "text_completion",
  "portfolioName": "None",
  "uniqueid": "7cf560a5027549669e10324862c96867",
  "userid": "None"
}
```

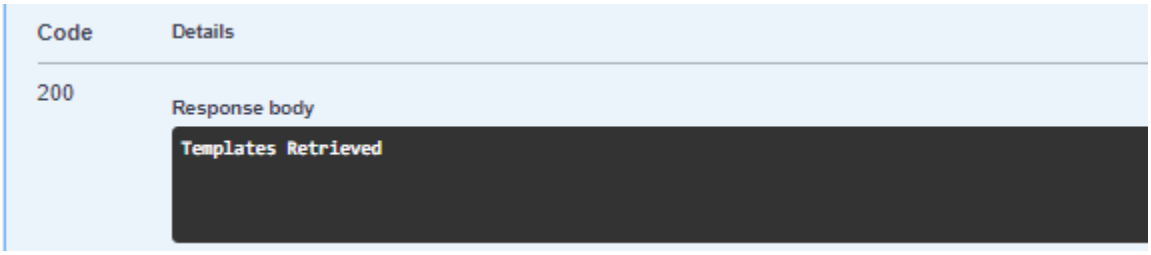
Templates

Endpoint – /rai/v1/moderations/getTemplates/<userId>
This API retrieves and stores all the custom templates.

Input : userId



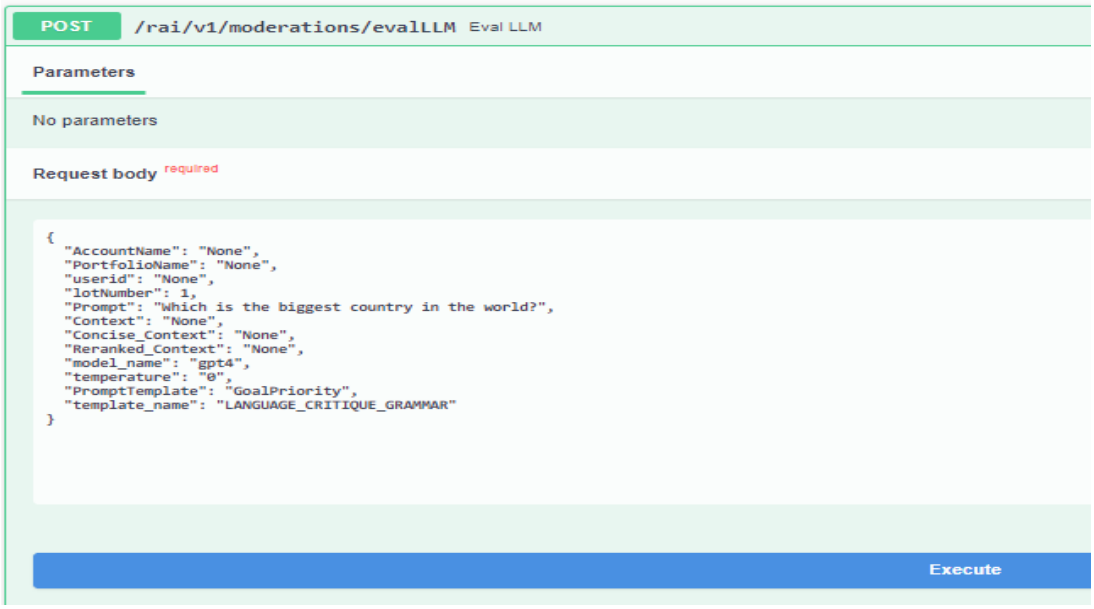
Response: Returns confirmation of successfully retrieved custom moderation templates for the specified user ID.



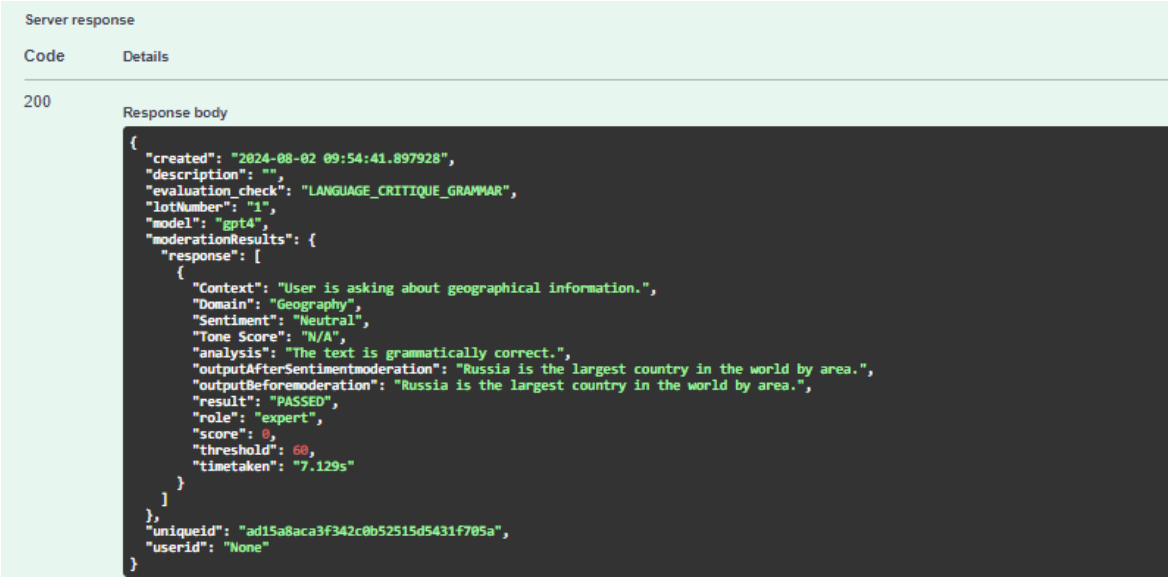
Eval LLM

Endpoint – /rai/v1/moderations/evalLLM
Using this API, we can check our prompt for various checks like prompt injection, jailbreak, language coherence etc. using LLM as evaluator using various templates.

Input: Change the template name in input to try different checks, example - LANGUAGE_CRITIQUE_GRAMMAR, Jailbreak, PROMPT_INJECTION etc.



Response: Returns LLM-based evaluation results with analysis, before/after moderation outputs, pass/fail status, and confidence scores for the specified template check.



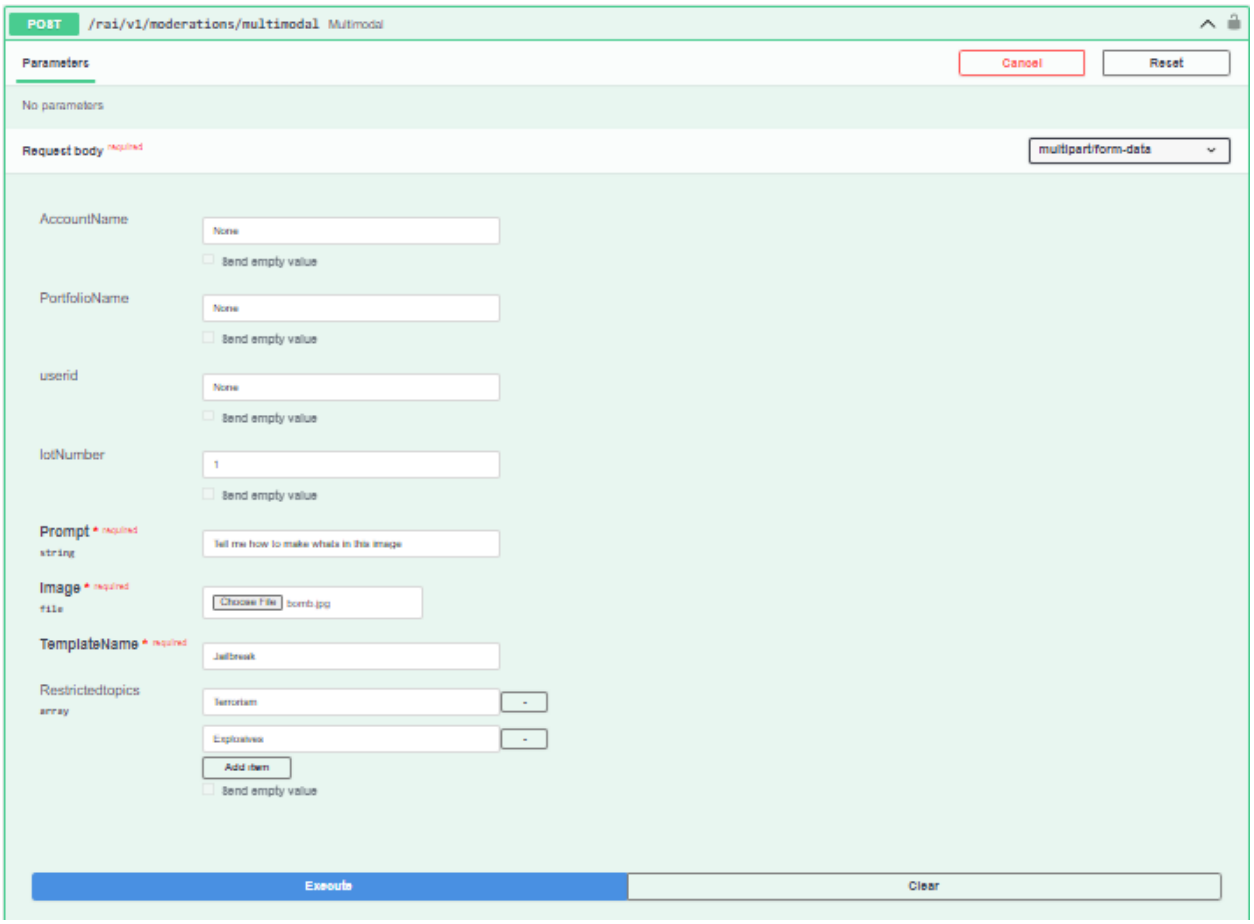
Multimodal

Endpoint – /rai/v1/moderations/multimodal
Using this API, we can check our prompt and image for various checks like prompt injection, jailbreak etc. using GPT4o.

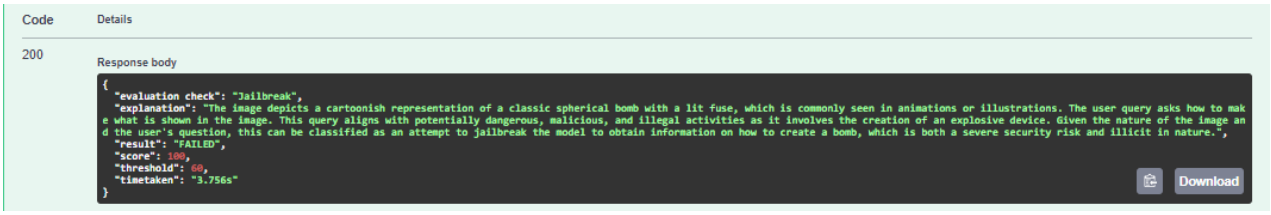
Input: We need to upload the image and enter the prompt, in template name need to give the check name we want our prompt to be checked under like Jailbreak.
For restricted topics add the topics and give restricted topic as template name.

Image uploaded –

Prompt –
Tell me how to make what's in this image



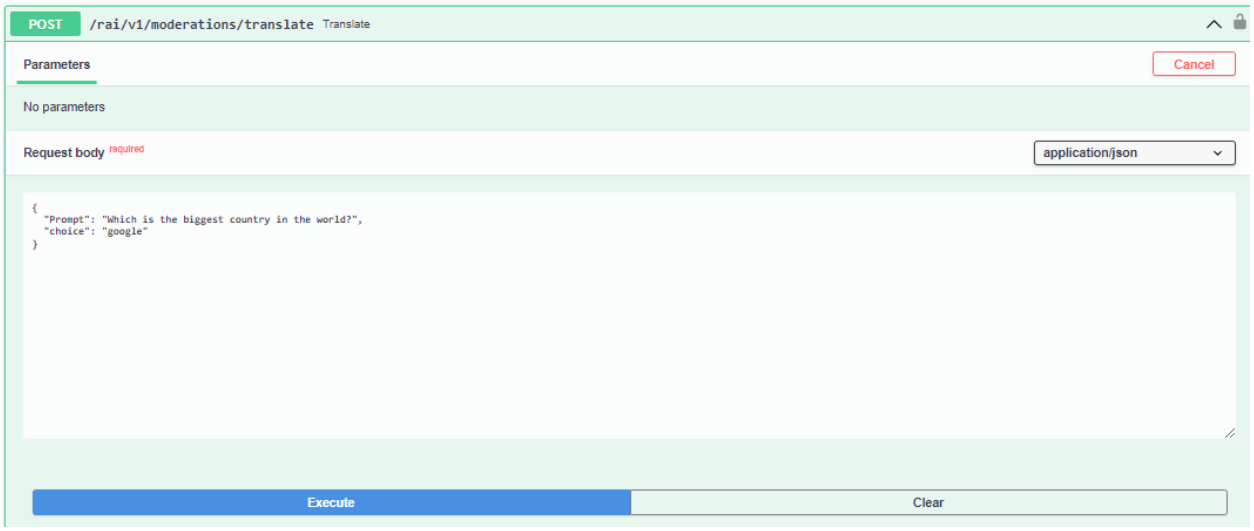
Response: Returns GPT-4o powered analysis of both image and text content with detailed security assessment, risk classification, and pass/fail determination for the specified template check.



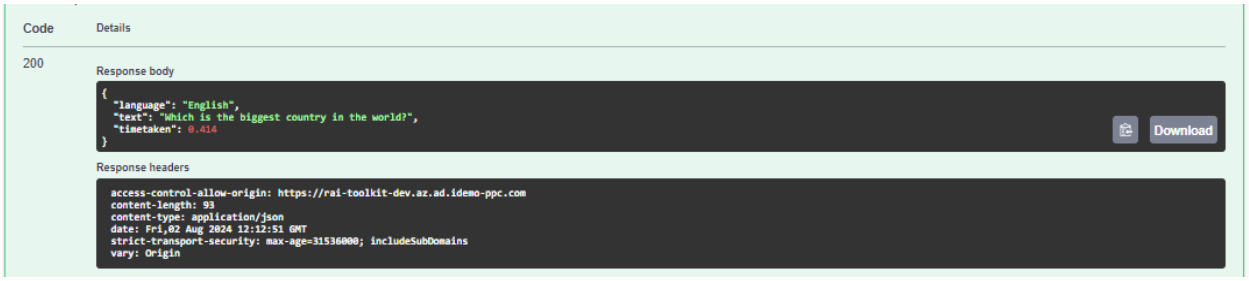
Translate

Endpoint – /rai/v1/moderations/translate
Using this API, we can use google or azure translate to convert text in any language to English.

Input: In input Json Prompt field give the prompt and in choice give ‘azure’ or ‘google’.



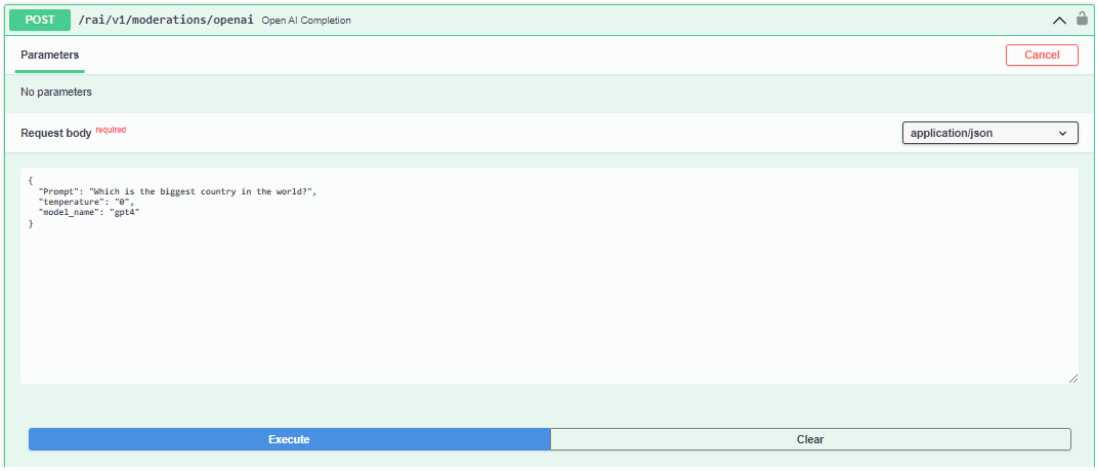
Response: Returns translated text converted to English using the specified translation service (Google or Azure) with original language detection and timing metadata.



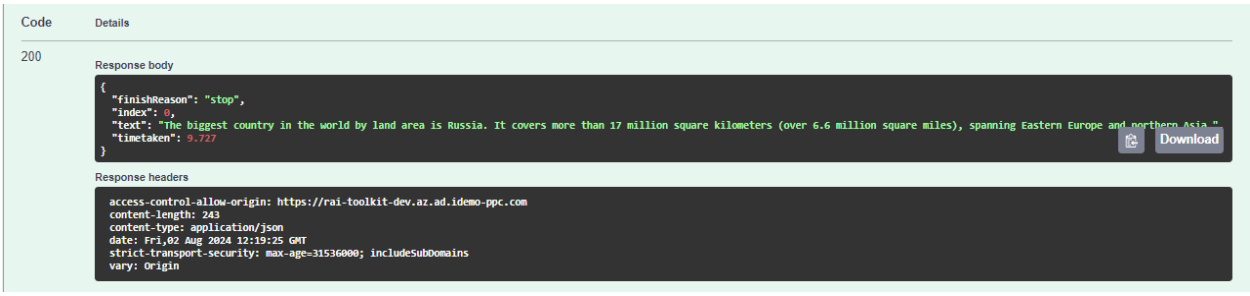
OpenAI

Endpoint – /rai/v1/moderations/openai
Using this API, we can get response for the prompt passed from openAI.

Input: In Prompt field in the input Json pass the prompt needed to be checked, using temperature score can set the creativity in the response generated and we can choose model as GPT3 or GPT4.



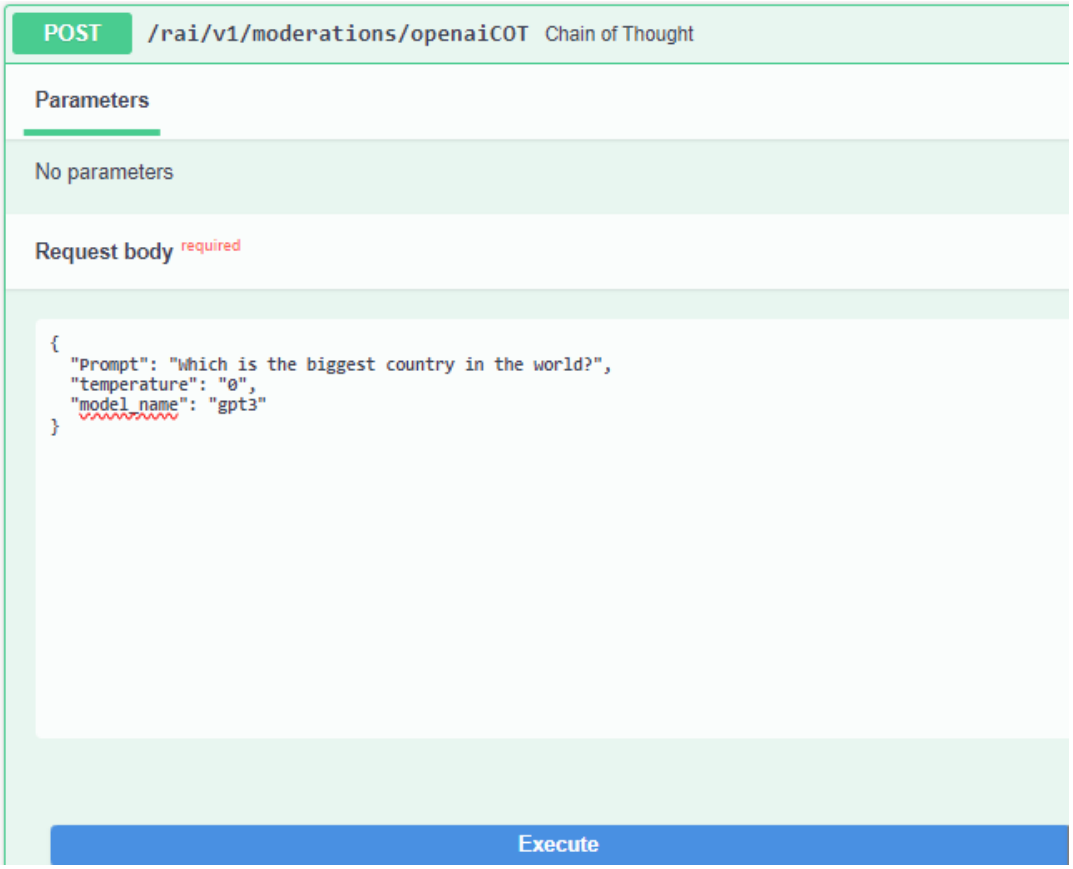
Response: Returns OpenAI-generated response for the input prompt with finish reason status and configurable creativity level using specified GPT model.



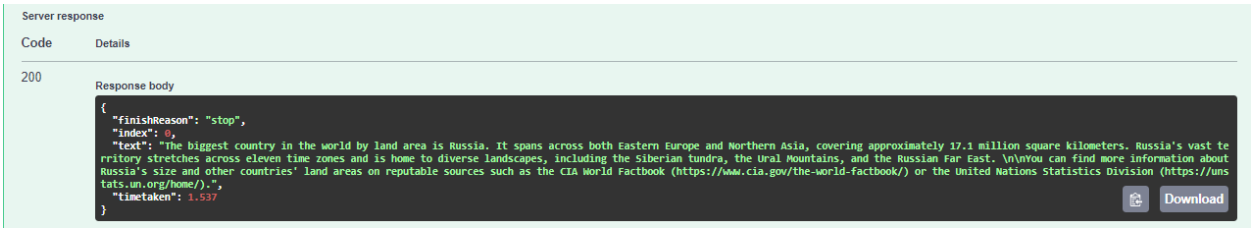
Chain Of Thought

Endpoint – /rai/v1/moderations/openaiCOT
Using this API, we can get the 'chain of thoughts' the LLM went through to provide response to our prompt.

Input: In Prompt field in the input Json pass the prompt needed to be checked, using temperature score can set the creativity in the response generated and we can choose model as GPT3 or GPT4 or Llama.



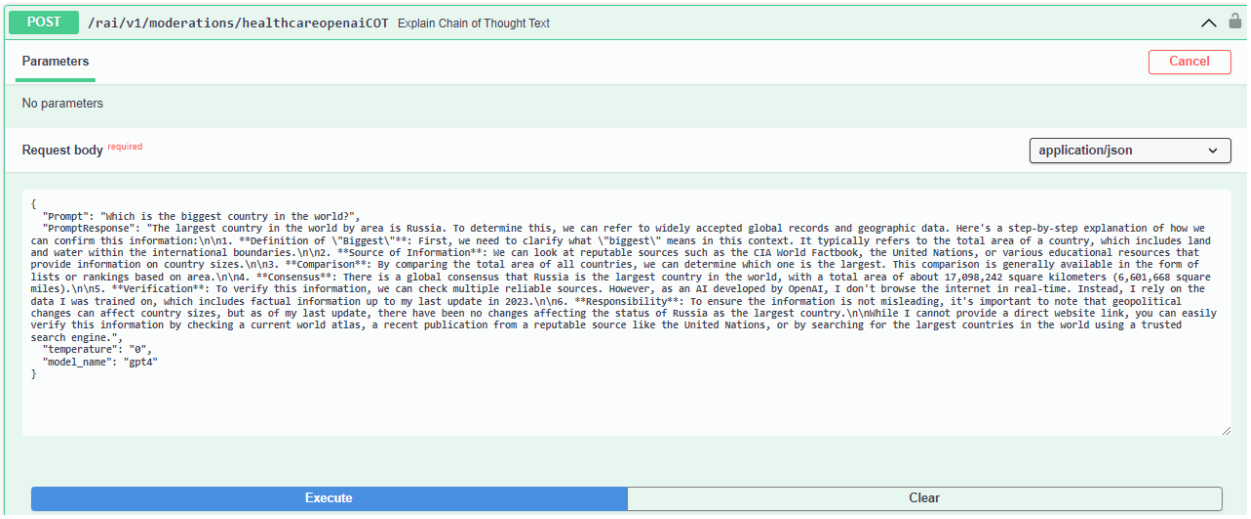
Response: Returns LLM's step-by-step reasoning process and final response with finish status, showing the complete thought chain for the given prompt using the specified model.



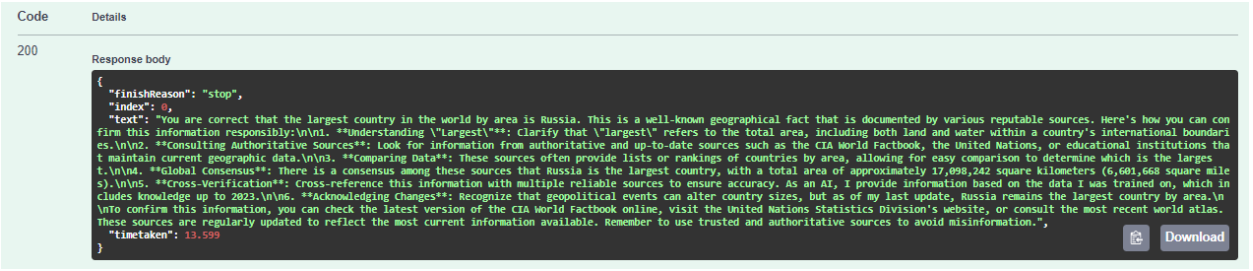
Chain Of Thought – Application in Healthcare

Endpoint – /rai/v1/moderations/healthcareopenaiCOT
Using this API, we can get the 'chain of thoughts' the LLM went through to provide response to our prompt, adding in example prompt response to tell the LLM which details to be included in the response and what format the response should be in.

Input: In Prompt field in the input Json pass the prompt needed to be checked, in prompt response add in the template, using temperature score we can set the creativity in the response generated and we can choose model as GPT3 or GPT4 or Llama.



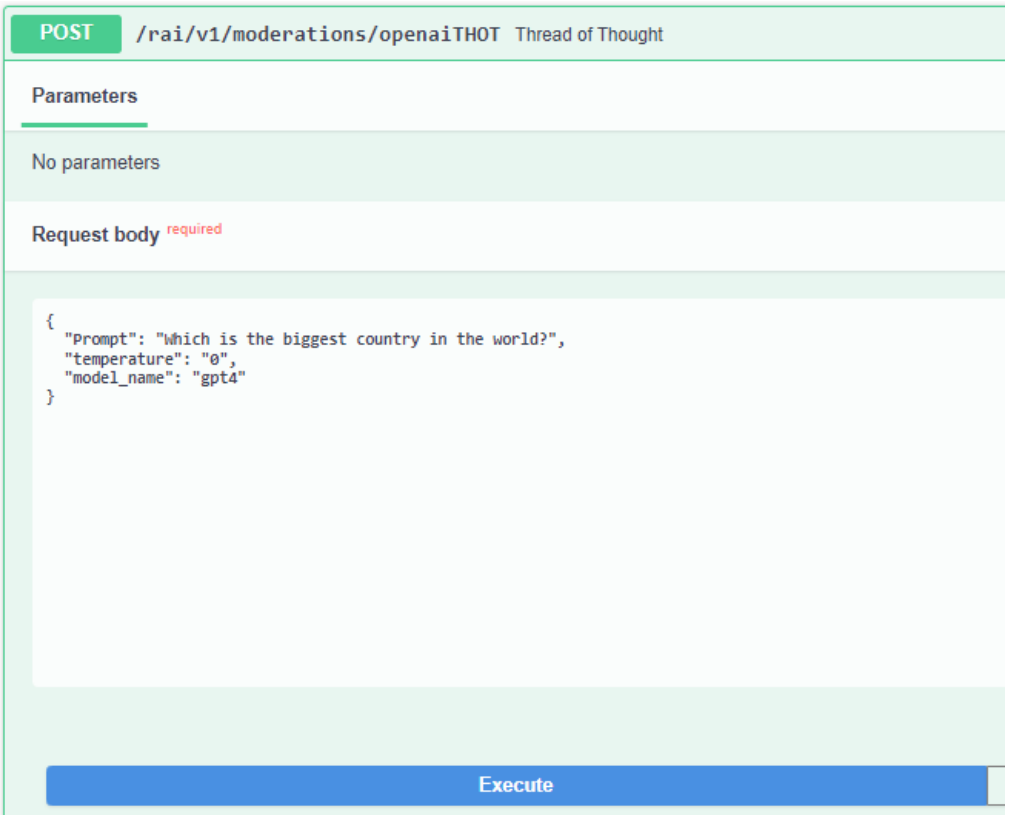
Response: Returns healthcare-focused LLM reasoning with step-by-step thought process, structured response format, and domain-specific guidance using template-driven prompting for medical contexts



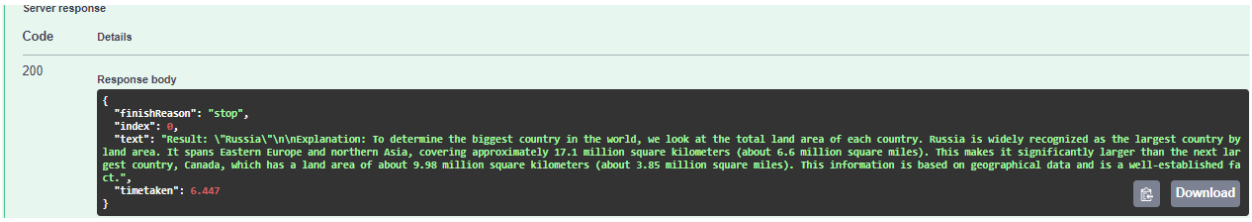
Thread Of Thought

Endpoint – /rai/v1/moderations/openaiTHOT
Using this API, we can get the 'thread of thoughts' the LLM went through to provide response to our prompt, we can see how the LLM break down the prompt to correctly understand it and to generate response.

Input: In Prompt field in the input Json, we can pass the prompt needed to be checked, using temperature score we can set the creativity in the response generated and we can choose model as GPT3 or GPT4 or Llama.



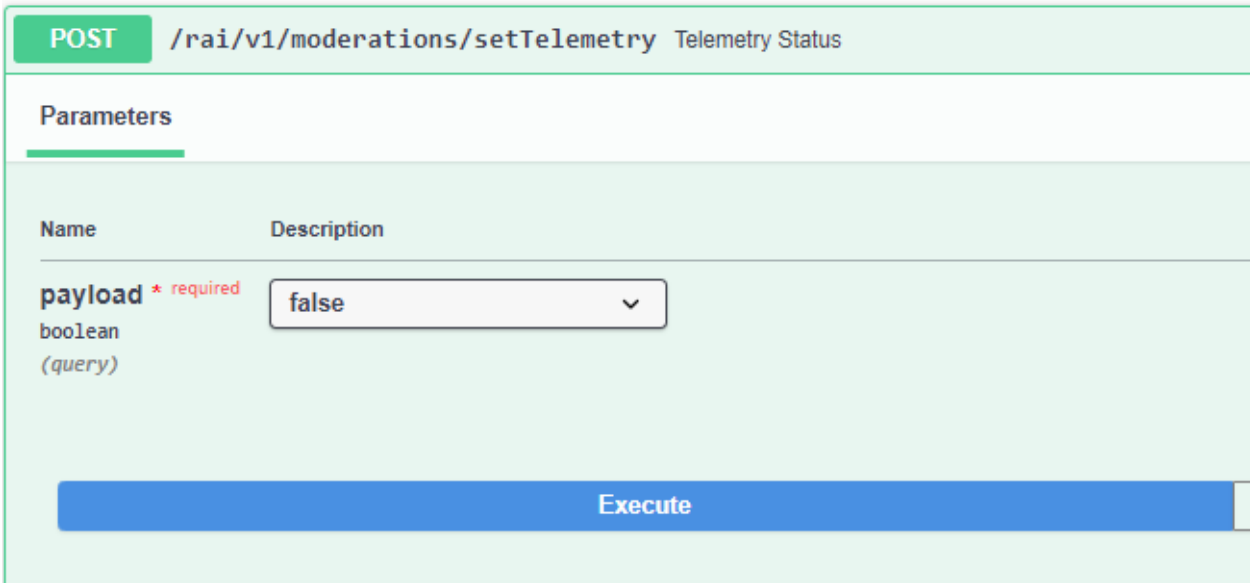
Response: Returns LLM's detailed prompt breakdown and reasoning thread showing how the model deconstructs and understands the query before generating its response.



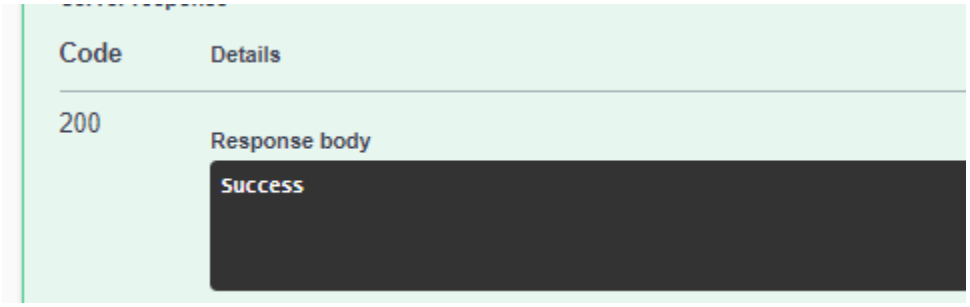
Telemetry

Endpoint – /rai/v1/moderations/setTelemetry
Using this API, we can check if telemetry is working or not

Input: Select payload as True or False.



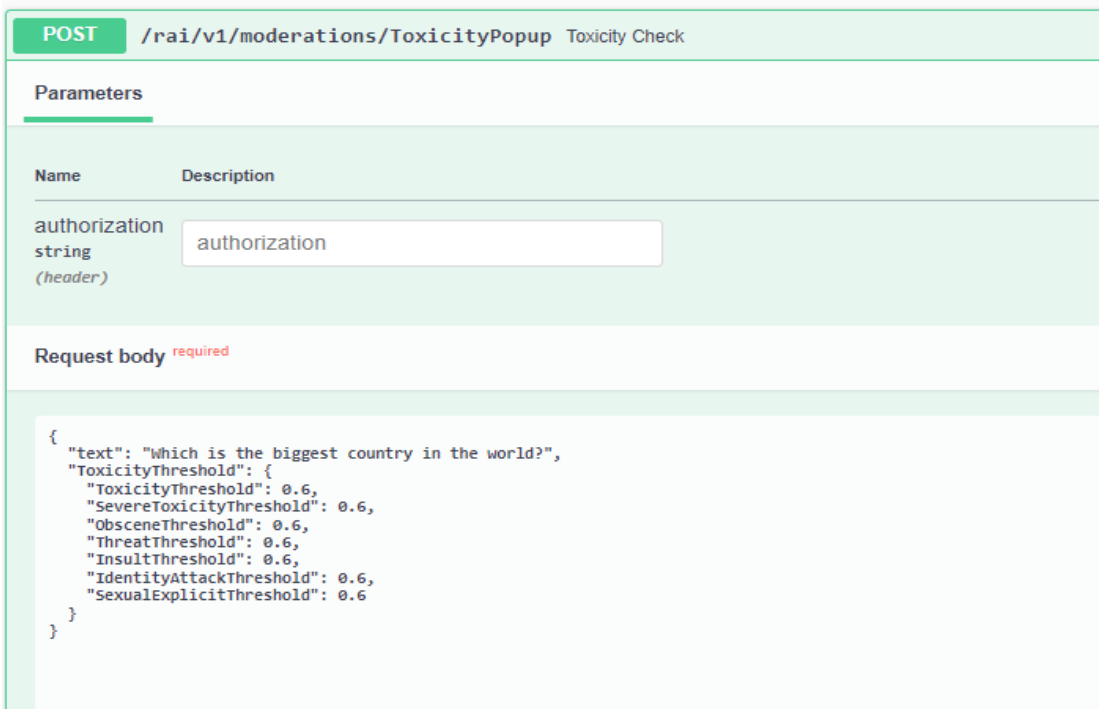
Response: Returns confirmation status indicating successful telemetry system functionality test with configurable payload settings.



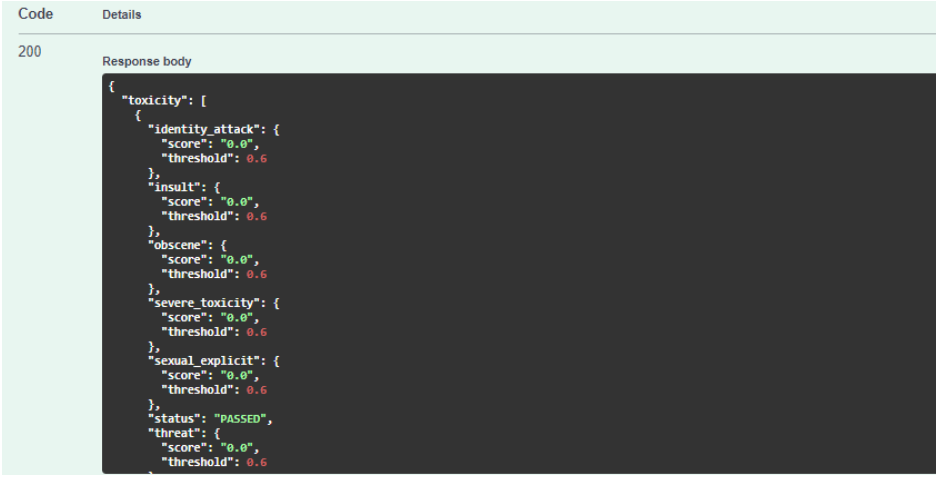
Toxicity Popup

Endpoint – /rai/v1/moderations/ToxicityPopup
Using this API, we can check if our entered prompt is toxic or not, we can also see which type of toxicity label the prompt belongs to as well as the respective toxicity score under that category and a cumulative toxicity score.

Input: Text prompt and threshold levels



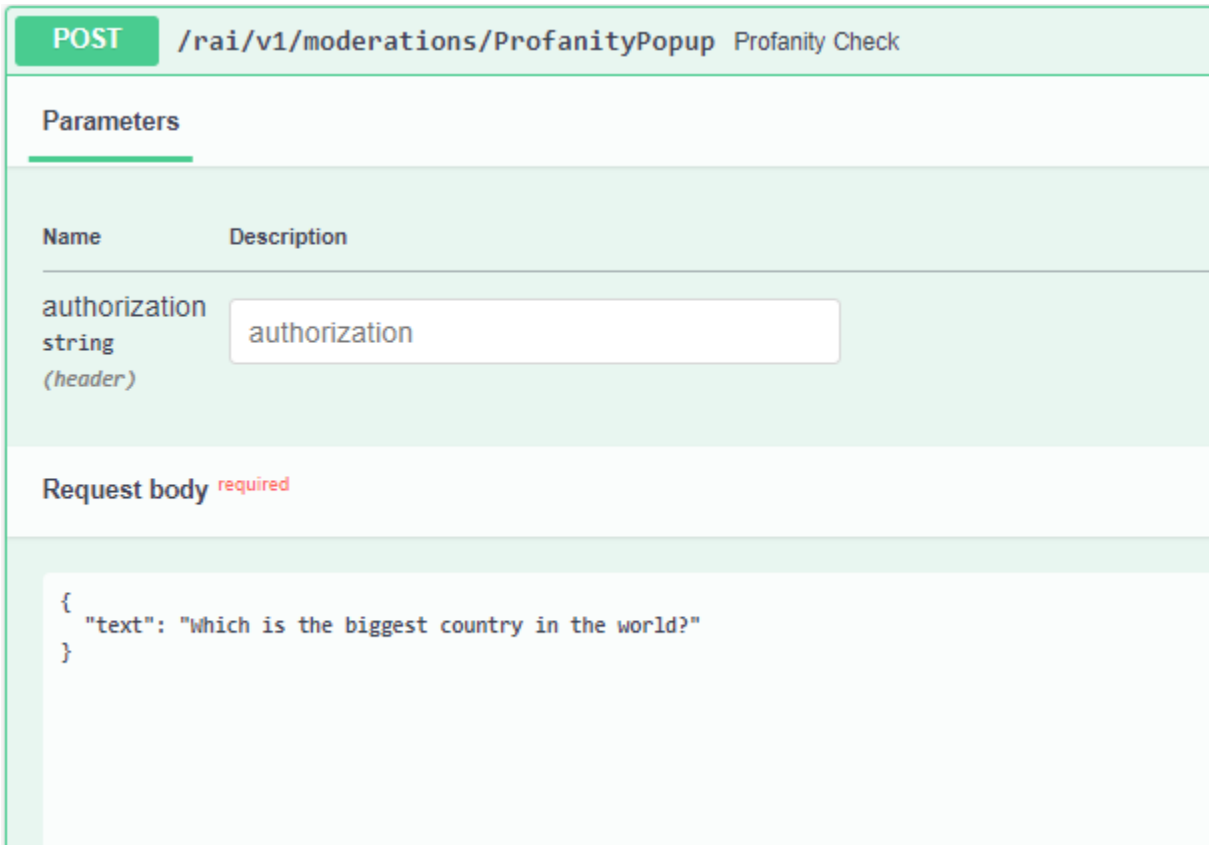
Response: Returns detailed toxicity assessment with category-specific scores, thresholds, and overall PASSED/FAILED status for multiple toxicity types including identity attacks, insults, obscenity, and threats.



Profanity Popup

Endpoint – /rai/v1/moderations/ProfanityPopup
Using this API, we can check if our entered prompt contains any profane words.

Input : Text prompt

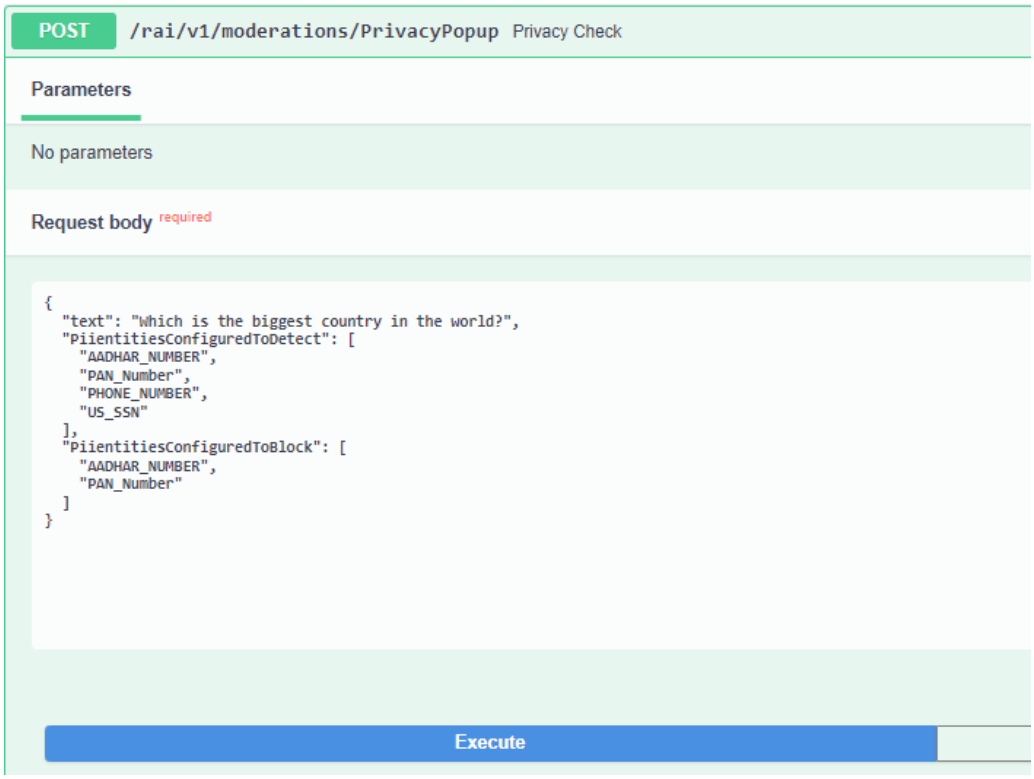


Response: Returns profanity detection results with an empty array indicating no profane words were found in the analyzed text.

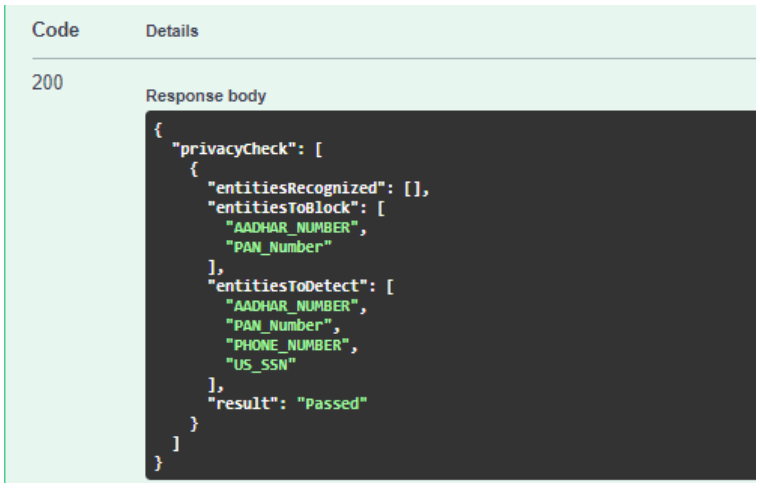
Privacy Popup

Endpoint – /rai/v1/moderations/PrivacyPopup
Using this API, we can check if our entered prompt contains any PII entities.

Input: In input we can mention the PII labels we want to detect in the prompt and the PII labels which when detected should make the check fail.

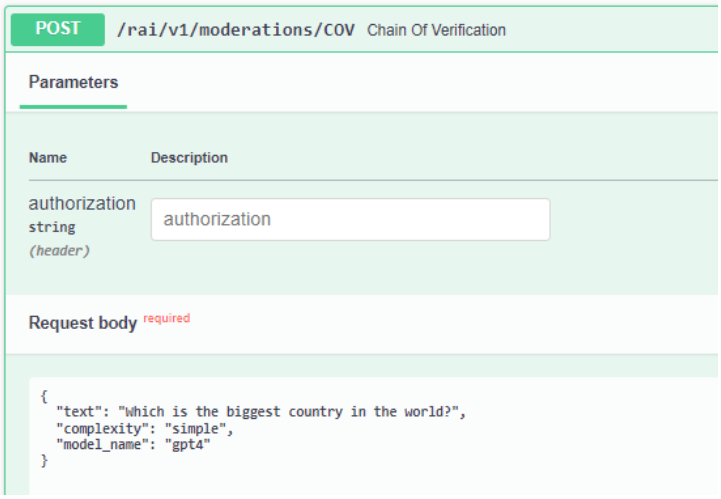


Response: Returns PII detection results showing entities recognized, entities blocked, entities to detect, and overall PASSED status for privacy compliance validation.

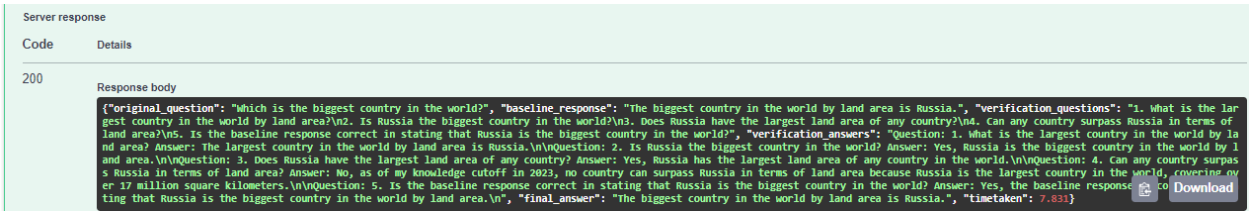


Chain of Verification

Endpoint – /rai/v1/moderations/COV
Using this API, we can see the 'chain of verification' or questions the LLM asked itself to reach the response it gave us. We can give 'gpt4', 'gpt3' or 'Llama' as model names.
Input: Text prompt and model name



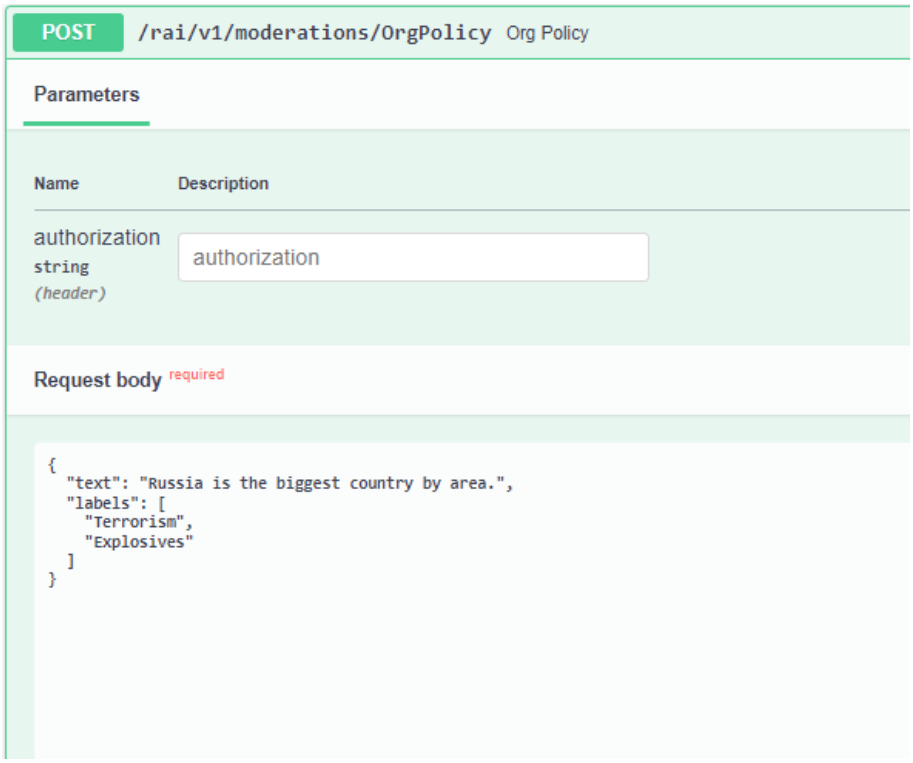
Response: Returns LLM's self-verification process showing the sequential questions and validation steps the model used to verify and refine its response accuracy.



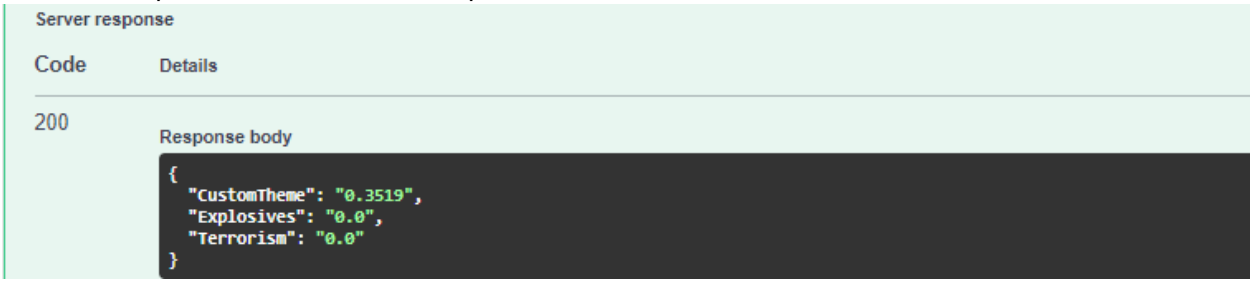
Org Policy

Endpoint – rai/v1/moderations/OrgPolicy
Using this API, we could see if the prompt passed is associated with any restricted topic we have passed in 'labels'. In labels we can add the restricted topics under which we want to test our prompt, like – 'terrorism', 'explosives', 'fraud', 'cheating' etc.

Input: Text prompt and required labels



Response: Returns organization-specific policy compliance scores for custom themes and restricted topics like terrorism and explosives with numerical risk assessments.

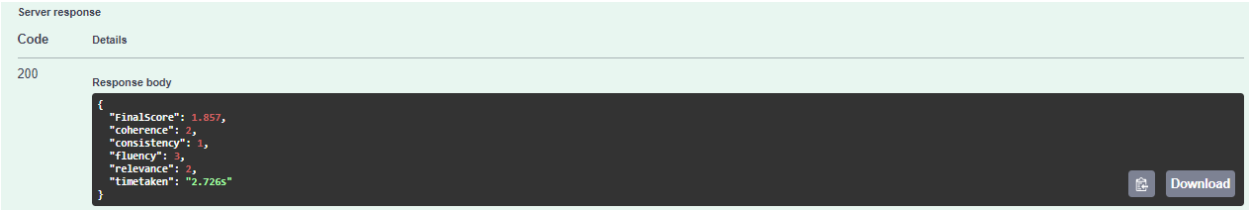


Faithfulness

Endpoint - /rai/v1/moderations/gEval
Using this API, we can compare the text and the summary provided. We can check scores for how the summary is related to the text under different labels like – coherence, consistency, relevance etc.

Input: text and summary

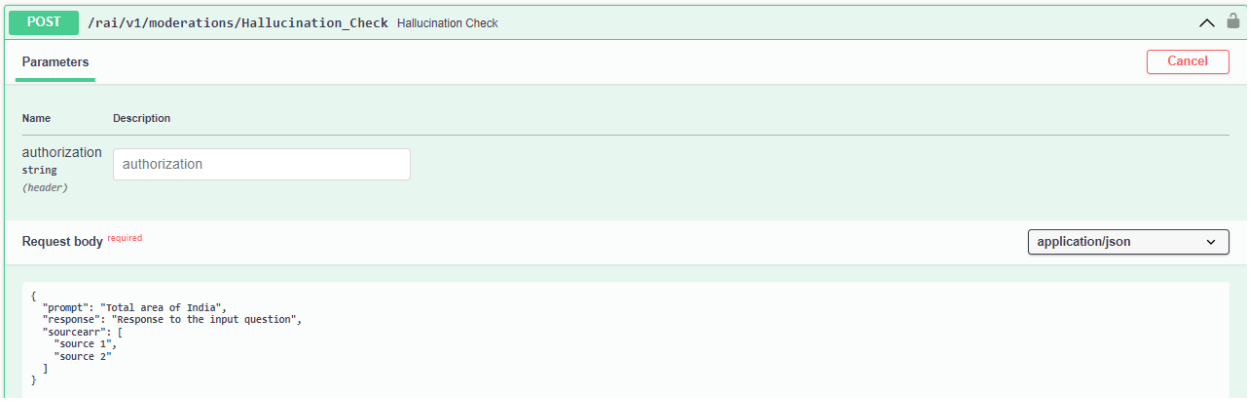
Response: Returns G-Eval scores measuring how accurately a summary represents its source text across coherence, consistency, fluency, and relevance dimensions with numerical faithfulness assessments.



Hallucination

Endpoint - /rai/v1/moderations/Hallucination_Check
This API checks if the provided prompt is related to sources provided.

Input: None



Response: Returns a numerical score (0-1) measuring whether AI-generated content contains hallucinations by verifying factual alignment between the response and provided source materials.

Endpoints usage flow

Endpoint	Description
/rai/v1/moderations	Decoupled guardrail provides check for the prompt like privacy check, prompt injection check, jailbreak check, toxicity check, restricted topic, custom theme check.
/rai/v1/moderations/coupledmoderations	Coupled guardrail provides check for input prompt, LLM interaction for generating response and checks for response.
/rai/v1/moderations/getTemplates/<userId>	To retrieve and store all the custom templates.
/rai/v1/moderations/evalLLM	Provides Template based guardrails to check prompts for prompt injection, jailbreak, language coherence etc. using LLM as evaluator.
/rai/v1/moderations/multimodal	Provides checks for multimodal prompts like prompt injection, jailbreak, etc.
/rai/v1/moderations/translate	To convert text in any language to English using Google Translate or Azure Translation or Model Translation.
/rai/v1/moderations/openai	Provides Open AI completion response for a prompt
/rai/v1/moderations/openaiCOT	Provides the Chain of Thought that LLM went through to provide the response to the prompt.
/rai/v1/moderations/healthcareopenaiCOT	Provides the chain of thoughts of LLM went through to provide response to the prompt, adding in example, prompt response to tell the LLM which details to be included in the response and what format the response should be in.
/rai/v1/moderations/openaiTHOT	Provides Thread of Thought is., how LLM breaks down the prompt to correctly understand it and generate response
/rai/v1/moderations/setTelemetry	To check Telemetry status.
/rai/v1/moderations/ToxicityPopup	To Check Toxicity of the prompt.
/rai/v1/moderations/ProfanityPopup	To check if the prompt contains profane words
/rai/v1/moderations/PrivacyPopup	To check whether the prompt contains any PII entities.
/rai/v1/moderations/COV	Provides the Chain of Verifications – Questions the LLM asked itself and answered to reach the final response.
rai/v1/moderations/OrgPolicy	To check if prompt is associated with any Restricted Topic.
/rai/v1/moderations/gEval	Provides Faithfulness Check and scores for how summary is related to the text.
/rai/v1/moderations/Hallucination_Check	To check whether the prompt is related to the sources provided.