Infosys
Responsible AI Office

Infosys Responsible AI Toolkit

Explainability for LLMs

API usage Instructions

Contents

Introduction

LLM Explain provides explanations for Large Language Models using methods such as token importance, Graph of Thoughts, Logic of Thought, Thread of Thought , Chain of Thought and Search Augmentation. It evaluates the responses with metrics including uncertainty, relevancy, and coherence to ensure the reliability and clarity of Generative AI models' outputs.

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 LLM-Explainability tenet are outlined below.

Sentiment Detection and Explainability

Endpoint – /rai/v1/llm-explainability/sentiment-analysis
Using this API, we can see sentiment analysis of the prompt along with token importance.

Input:

inputPrompt: Replace the input prompt with the prompt you want to check

```
{
  "inputPrompt": "Unfortunately the movie served with bad visuals but the actors performed well"
}
```

Response:


```
{
  "explanation": [
    {
      "predictedTarget": "Negative",
      "anchor": [
        "unfortunately",
        "bad"
      ],
      "explanation": "The model identified the sentiment of the prompt as negative based on two key words it picked up on: 'unfortunately' and 'bad'. These words are typically associated with a negative sentiment. In this context, despite the actors' good performance, the model interpreted the overall sentiment as negative due to the phrasing about the movie's bad visuals being presented unfortunately.",
      "token_importance_mapping": [
        {
          "token": "Unfortunately",
          "importance_score": 48,
          "position": 0
        },
        {
          "token": "bad",
          "importance_score": 52,
          "position": 5
        }
      ],
      "time_taken": 4.826
    }
  ]
}
```

Uncertainty and Coherence Metrics

Endpoint - /rai/v1/llm-explainability/uncertainty

Using this API, we can evaluate the LLM response in terms of confidence level. to evaluate this, we have metrics such as uncertainty and coherence. Along with evaluation metrics, It provides corresponding explanation and corrective actions to improve the LLM response.

Request Payload:

Example Value | Schema


```
{
  "inputPrompt": "Who are the co-founders of Infosys?",
  "response": "Infosys was co-founded by Narayana Murthy along with six other engineers: Nandan Nilekani, S. Gopalakrishnan (Kris), S. D. Shibulal, K. Dinesh, N. S. Raghavan, and Ashok Arora. Established in 1981, Infosys started with a modest capital of $250 and has since grown into one of the largest IT services companies in the world. Narayana Murthy, often regarded as the face of Infosys, played a pivotal role in shaping the company's culture and vision, while the combined efforts of all co-founders contributed to its remarkable growth and success in the global IT industry.",
  "endpointDetails": {
    "modelEndpointUrl": "http://localhost:8002/model/endpoint",
    "endpointInputParam": {
      "input_parameter": "inputs",
      "parameters": {
        "temperature": 0.2
      }
    },
    "endpointOutputParam": "['output']['choices'][0]['text']"
  }
}
```


inputPrompt: provided your input prompt.
response: provide your input prompt response for the provided prompt.
endpointDetails [Optional Parameter]: If model is deployed and available as an inference endpoint, this parameter can be configured to use that model else you keep it as null.

- **modelEndpointUrl**": "YOUR ENDPOINT", Reachable inference endpoint of the model. Ex: http://localhost:8002/model/endpoint
- **endpointInputParam**:
  - **input_parameter**: “inputs", # Variable name used to pass Prompt/Query to endpoint payload.
  - **parameters**: # Additional parameters to be passed in endpoint payload.
- **endpointOutputParam**: "['output']['choices'][0]['text']"

Response:

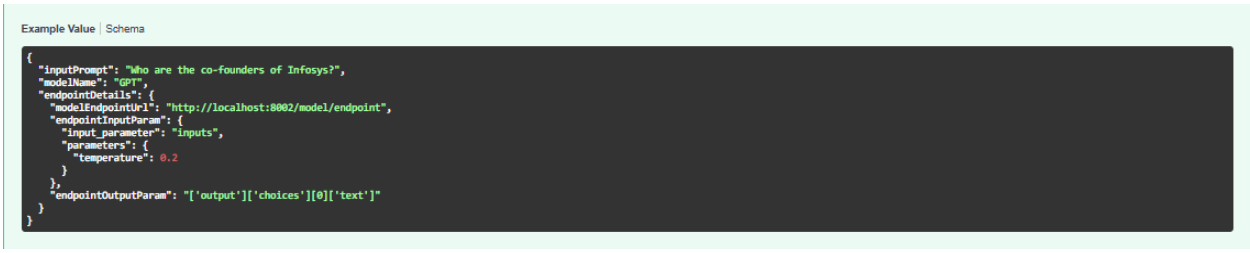

```
{
  "uncertainty": {
    "score": 0,
    "explanation": "The response provided is confident and specific. It lists all the co-founders correctly and includes their contributions and the company's journey. Furthermore, the details provided, especially around the establishment date and initial capital, are commonly known facts and are not debated.",
    "recommendation": "The response is certain and clear in answering the prompt about the Infosys co-founders and the narrative surrounding the initial formation of the company. Therefore, no changes are required for the prompt to improve this metric score.",
    "uncertainty_level": "Highly Certain"
  },
  "coherence": {
    "score": 95,
    "explanation": "The response is highly coherent. It logically begins with the names of the co-founders, then transitions smoothly into a brief history of Infosys' establishment and growth, and the roles that the co-founders played. All the information is logically connected and relevant to the given prompt.",
    "recommendation": "The response showcases effective coherence. It successfully maintains a logical flow and connects all the bits of information smoothly. Therefore, there are no recommendations for changing the input prompt to improve the coherence score as the flow and linkage of the information provided is excellent.",
    "coherence_level": "Highly Coherent"
  },
  "time_taken": 8.25
}
```


Download

Token Importance

Endpoint - /rai/v1/llm-explainability/token_importance
Using this API, we can get importance of each token in the input prompt.

Input:



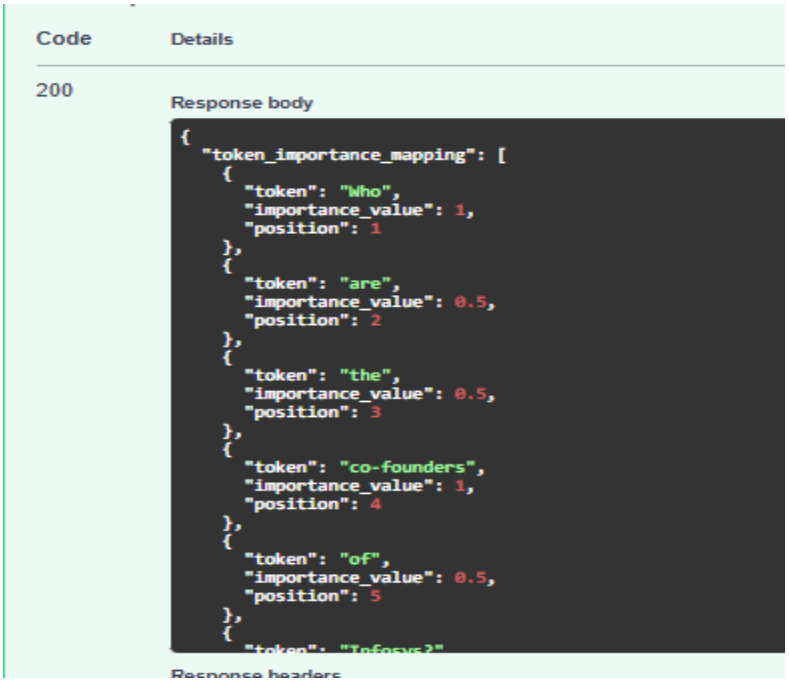
inputPrompt: provided your input prompt.

modelName [Optional Parameter]: provide model name. if you are using endpoint details keep model name as null.

endpointDetails [Optional Parameter]: If model is deployed and available as an inference endpoint, this parameter can be configured to use that model else you keep it as null.

- **modelEndpointUrl**": "YOUR ENDPOINT", Reachable inference endpoint of the model. Ex: http://localhost:8002/model/endpoint
- **endpointInputParam**:
    - **input_parameter**: “inputs", # Variable name used to pass Prompt/Query to endpoint payload.
    - **parameters**: # Additional parameters to be passed in endpoint payload.
- **endpointOutputParam**: "['output']['choices'][0]['text']"

Response:



Code | Details

200

Response body


```
{
  "token_importance_mapping": [
    {
      "token": "Who",
      "importance_value": 1,
      "position": 1
    },
    {
      "token": "are",
      "importance_value": 0.5,
      "position": 2
    },
    {
      "token": "the",
      "importance_value": 0.5,
      "position": 3
    },
    {
      "token": "co-founders",
      "importance_value": 1,
      "position": 4
    },
    {
      "token": "of",
      "importance_value": 0.5,
      "position": 5
    },
    {
      "token": "Infosys?"
```


Response headers

Response body


```
    {
      "token": "are",
      "importance_value": 0.5,
      "position": 2
    },
    {
      "token": "the",
      "importance_value": 0.5,
      "position": 3
    },
    {
      "token": "co-founders",
      "importance_value": 1,
      "position": 4
    },
    {
      "token": "of",
      "importance_value": 0.5,
      "position": 5
    },
    {
      "token": "Infosys?",
      "importance_value": 1,
      "position": 6
    }
  ],
  "time_taken": 3.883,
  "token_cost": 0.015
}
```

Factuality Check by Internet Search

Endpoint - /rai/v1/llm-explainability/serper_response
Using this API, we can verify the LLM's response with the Google search engine to check for factual score and an explanation

Input:

inputPrompt: Replace the input prompt with the prompt you want to check

lls_response: provide the response of input text


```
{
  "inputPrompt": "Who are the co-founders of Infosys?",
  "llm_response": "Infosys, a global leader in technology services and consulting, was founded in 1981 by seven visionaries: N.R. Narayana Murthy, Nandan Nilekani, S. Gopalakrishnan, S.D. Shibulal, K. Dinesh, N.S. Raghavan, and Ashok Arora. These co-founders combined their expertise and entrepreneurial spirit to create a company that has since grown into one of the largest and most respected IT services firms in the world. Infosys, headquartered in Bangalore, India, has been instrumental in the global IT revolution, providing innovative solutions and services to clients across various industries. The founders' commitment to excellence and their forward-thinking approach laid a strong foundation for the company's enduring success."
}
```


Response:


```
{
  "internetResponse": [
    "Infosys was co-founded by Salil Parekh in January 2018."
  ],
  "metrics": [
    {
      "metricName": "Factuality Check",
      "score": 1,
      "explanation": [
        {
          "Fact": "The co-founders of Infosys are N.R. Narayana Murthy, Nandan Nilekani, S. Gopalakrishnan, S.D. Shibulal, K. Dinesh, N.S. Raghavan, and Ashok Arora.",
          "Reasoning": "The context provides a quote from an article which confirms that these seven individuals are indeed the co-founders of Infosys. However, it is unclear if this statement is valid as of today since it does not mention any dates. It is possible that some of these co-founders may have stepped down or new co-founders may have joined. Therefore, we cannot confirm the validity of this statement without more information.",
          "Judgement": "Fact Unclear"
        }
      ]
    }
  ],
  "time_taken": 5.478
}
```


Download

Graph of Thought Reasoning

Endpoint - /rai/v1/llm-explainability/got
Using this API, we can view a graph of thought reasoning for the given input. Here, we provide the final thought, score and consistency level.

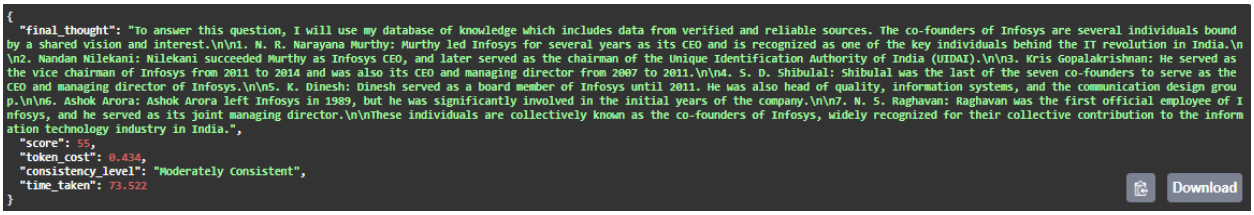
Input:

inputPrompt: Replace the input prompt with the prompt you want to check

modelName: Provide deployment name from Azure


```
{
  "inputPrompt": "Who are the co-founders of Infosys?",
  "modelName": "gpt4"
}
```

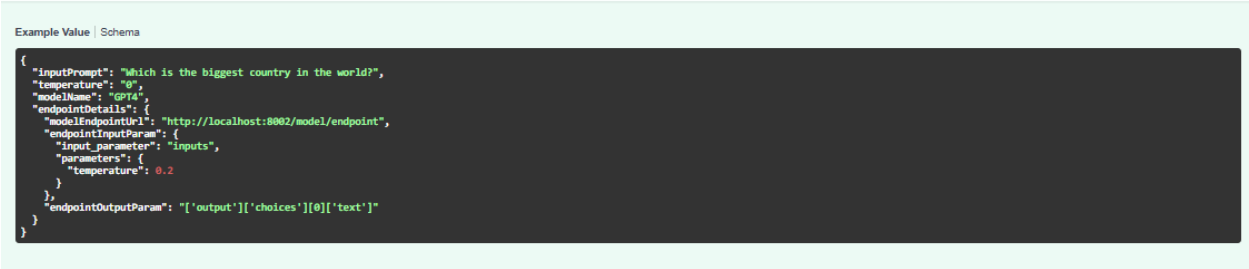
Response:



Chain Of Thought

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

Input:



inputPrompt: provided your input prompt.

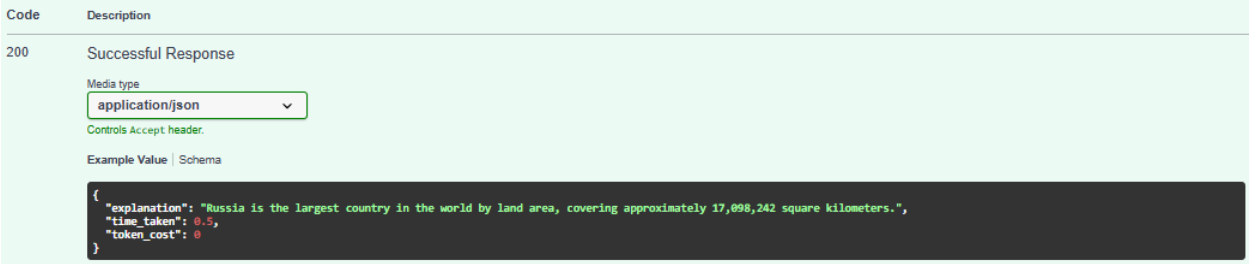
temperature: provide the temperature value.

modelName [Optional Parameter]: provide model name. if you are using endpoint details keep model name as null.

endpointDetails [Optional Parameter]: If model is deployed and available as an inference endpoint, this parameter can be configured to use that model else you keep it as null.

- **modelEndpointUrl**": "YOUR ENDPOINT", Reachable inference endpoint of the model. Ex: http://localhost:8002/model/endpoint
- **endpointInputParam**:
  - **input_parameter**: "inputs", # Variable name used to pass Prompt/Query to endpoint payload.
  - **parameters**: # Additional parameters to be passed in endpoint payload.
- **endpointOutputParam**: "['output']['choices'][0]['text']"

Response:



Thread Of Thought

Endpoint – /rai/v1/llm-explainability/thot
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:

inputPrompt: provided your input prompt.

temperature: provide the temperature value.

modelName [Optional Parameter]: provide model name. if you are using endpoint details keep model name as null.

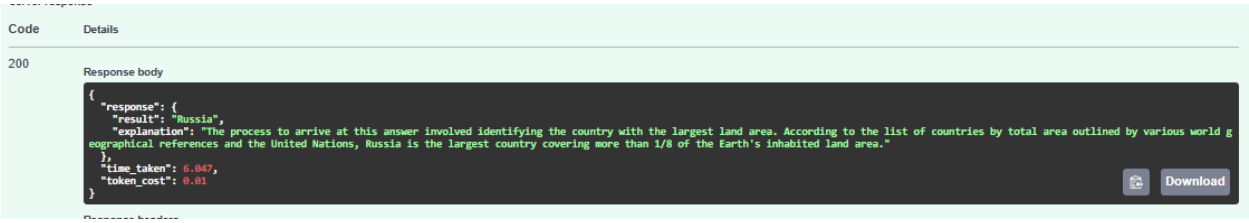
endpointDetails [Optional Parameter]: If model is deployed and available as an inference endpoint, this parameter can be configured to use that model else you keep it as null.

- **modelEndpointUrl**": "YOUR ENDPOINT", Reachable inference endpoint of the model. Ex: http://localhost:8002/model/endpoint
- **endpointInputParam**:
    - **input_parameter**: "inputs", # Variable name used to pass Prompt/Query to endpoint payload.

- **parameters**: # Additional parameters to be passed in endpoint payload.

- **endpointOutputParam**: "['output']['choices'][0]['text']"

Response:



Chain of Verification

Endpoint – /rai/v1/llm-explainability/cov
Using this API, we can see the 'chain of verification' or questions the LLM asked itself to reach the response it gave us.

Input:

inputPrompt: provided your input prompt.

Complexity: provide the complexity.

Translate:

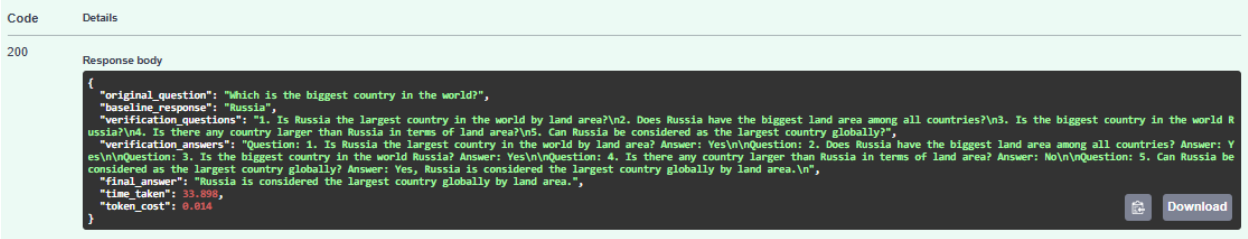
temperature: provide the temperature value.

modelName [Optional Parameter]: provide model name. if you are using endpoint details keep model name as null.

endpointDetails [Optional Parameter]: If model is deployed and available as an inference endpoint, this parameter can be configured to use that model else you keep it as null.

- **modelEndpointUrl**": "YOUR ENDPOINT", Reachable inference endpoint of the model. Ex: http://localhost:8002/model/endpoint
- **endpointInputParam**:
    - **input_parameter**: “inputs", # Variable name used to pass Prompt/Query to endpoint payload.
    - **parameters**: # Additional parameters to be passed in endpoint payload.
- **endpointOutputParam**: "['output']['choices'][0]['text']"

Response:

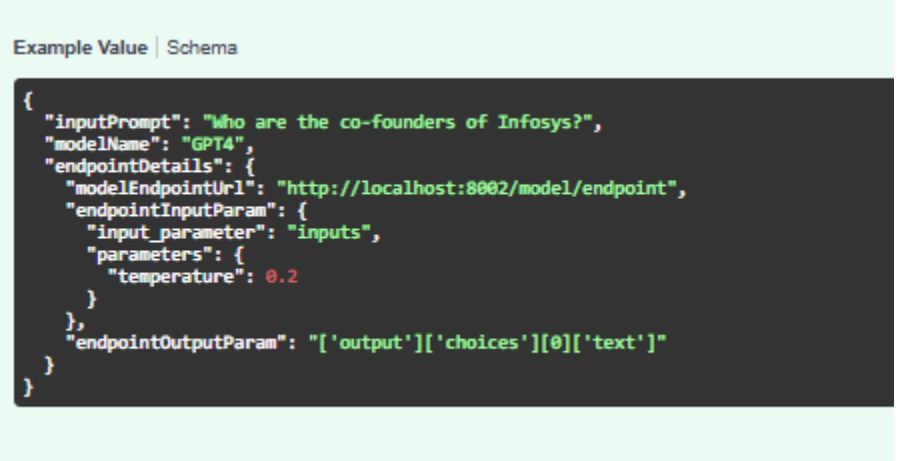


Reread with Thread of Thought:

Using this API, we can get the ‘thread of thoughts’ with rereading the prompt, 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

Endpoint – /rai/v1/llm-explainability/reread_reasoning

Input:



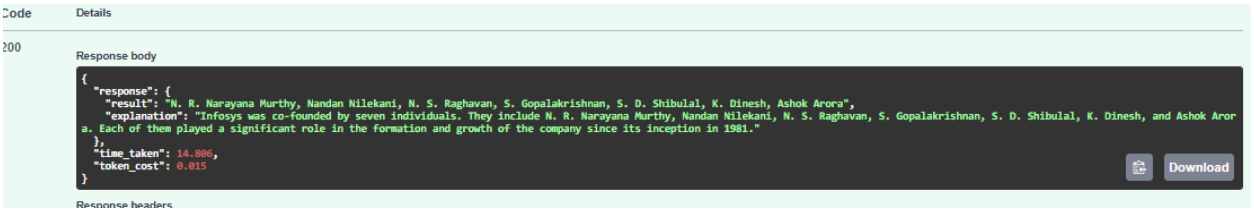
inputPrompt: provided your input prompt.

modelName [Optional Parameter]: provide model name. if you are using endpoint details keep model name as null.

endpointDetails [Optional Parameter]: If model is deployed and available as an inference endpoint, this parameter can be configured to use that model else you keep it as null.

- **modelEndpointUrl**": "YOUR ENDPOINT", Reachable inference endpoint of the model. Ex: http://localhost:8002/model/endpoint
- **endpointInputParam**:
    - **input_parameter**: “inputs", # Variable name used to pass Prompt/Query to endpoint payload.
    - **parameters**: # Additional parameters to be passed in endpoint payload.
- **endpointOutputParam**: "['output']['choices'][0]['text']"

Response:

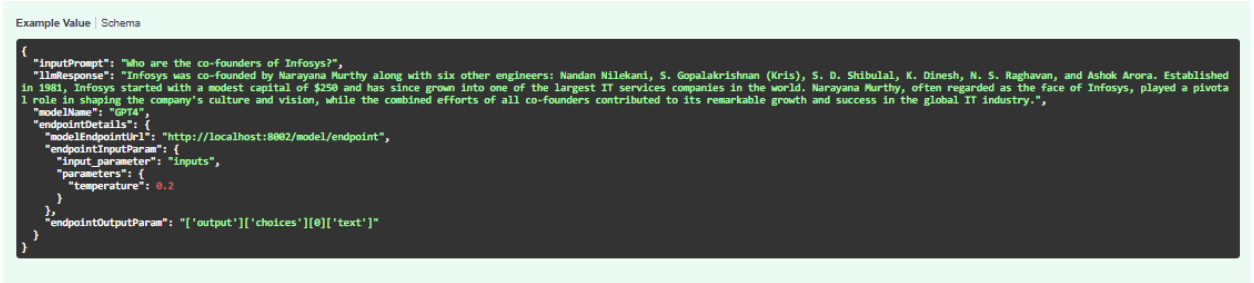


Logic of Thought:

Using this API, we can get response for the logical reasoning prompts with better reasoning and decisions

Endpoint – /rai/v1/llm-explainability/reread_reasoning

Input:



inputPrompt: provided your input prompt.

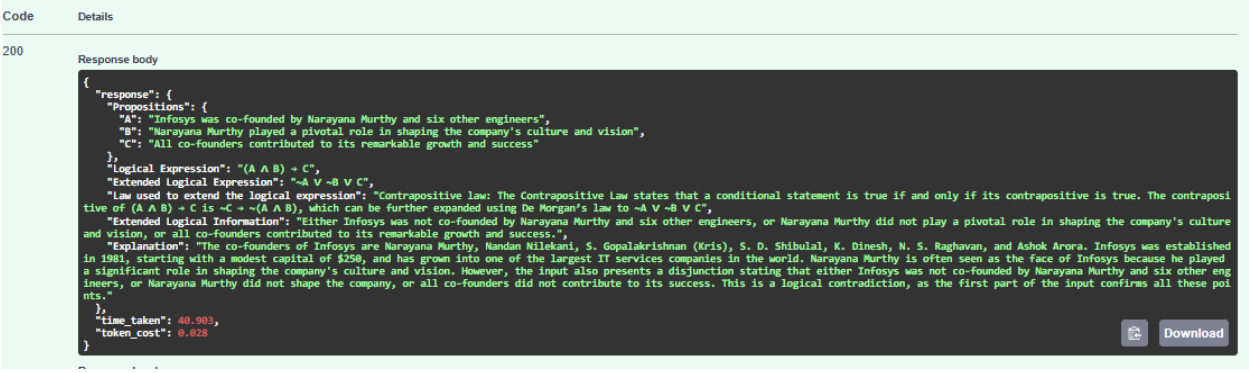
Response [Optional Parameter]: provide your input prompt response for the provided prompt.

modelName [Optional Parameter]: provide model name. if you are using endpoint details keep model name as null.

endpointDetails [Optional Parameter]: If model is deployed and available as an inference endpoint, this parameter can be configured to use that model else you keep it as null.

- **modelEndpointUrl**": "YOUR ENDPOINT", Reachable inference endpoint of the model. Ex: http://localhost:8002/model/endpoint
- **endpointInputParam**:
    - **input_parameter**: “inputs", # Variable name used to pass Prompt/Query to endpoint payload.
    - **parameters**: # Additional parameters to be passed in endpoint payload.
- **endpointOutputParam**: "['output']['choices'][0]['text']"

Response:



Bulk Processing:

Using this API, we can get response for some set of prompts instead of single prompt with all methods or specific methods.

Endpoint: /rai/v1/llm-explainability/bulk_processing

Input:

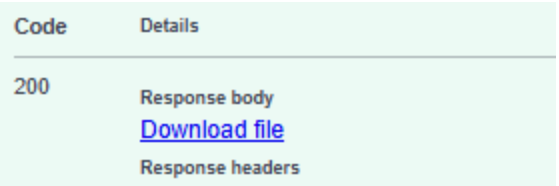
methods: Specify the method names you want to check or get an explanation for.

responseFileType: Indicate the desired response file type as json or excel.

userId: Provide the user ID.

File: Upload the file containing the input prompts and responses.

Response:



LLM Explainability endpoints in Moderation Layer

The following LLM reasoning endpoints are currently available in the Moderation Layer (responsible-ai-moderationlayer) repository. Please follow the setup instructions in the README file of the moderation layer repository to configure them. Ensure that the service is up and running to execute.

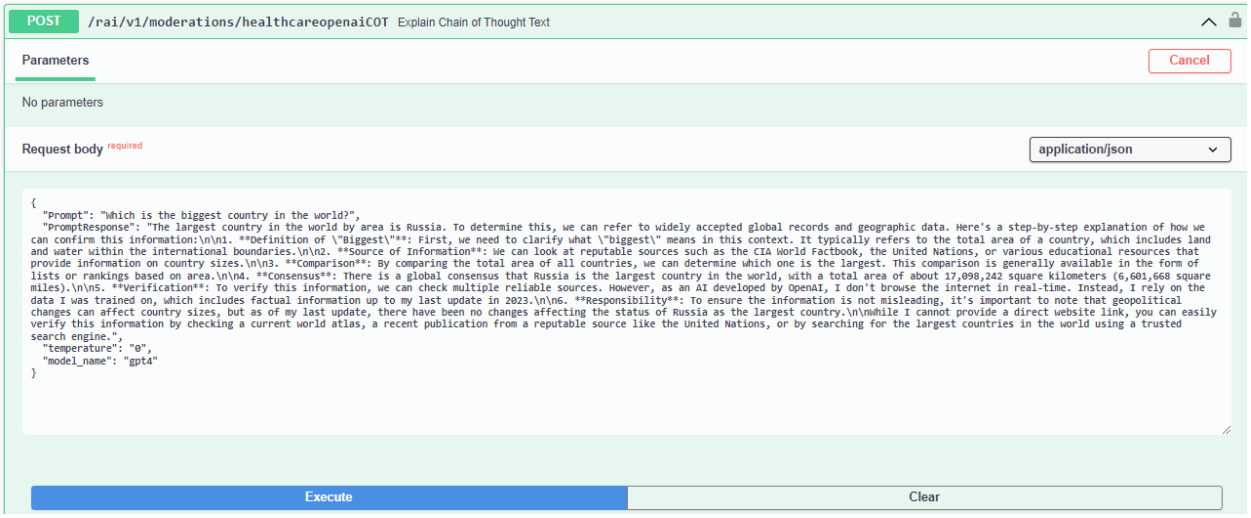
Chain Of Thought for RAG

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. If you are using RAG based application, you can generate chunks from vector storage add the respective context of RAG files along with the actual prompt. This API will consider context and generates explanation for the same.

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: