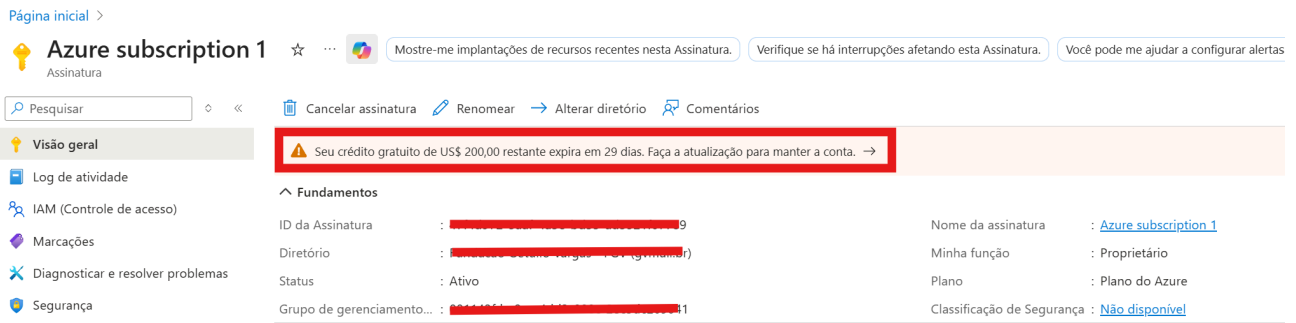


Passo a Passo da Criação do Projeto

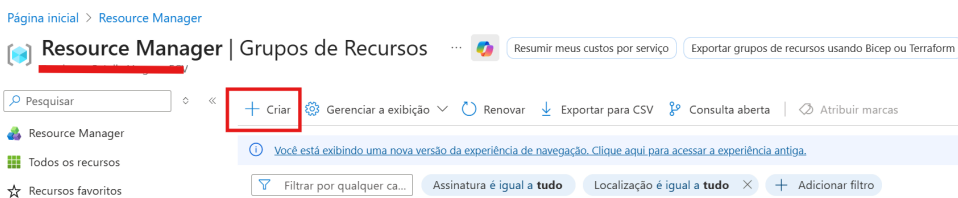
Autora: Iara Cristina Mescua Castro

Email: iaramescuacastro@gmail.com

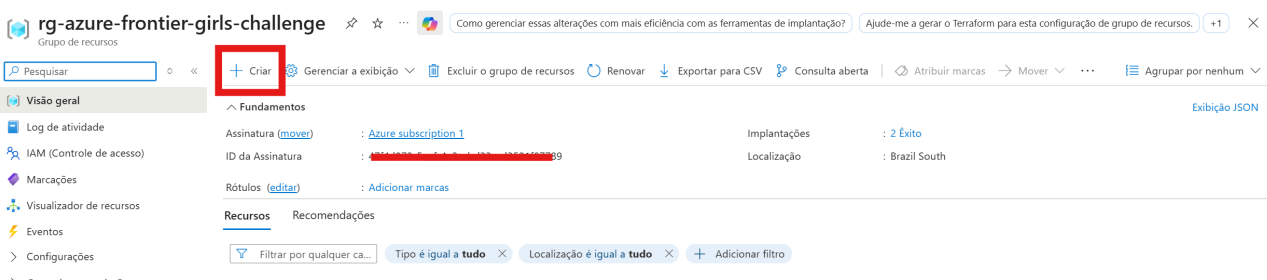
Neste projeto irei usar créditos do Azure pela conta de estudante, com disponibilidade de 200 dólares de créditos no primeiro mês.



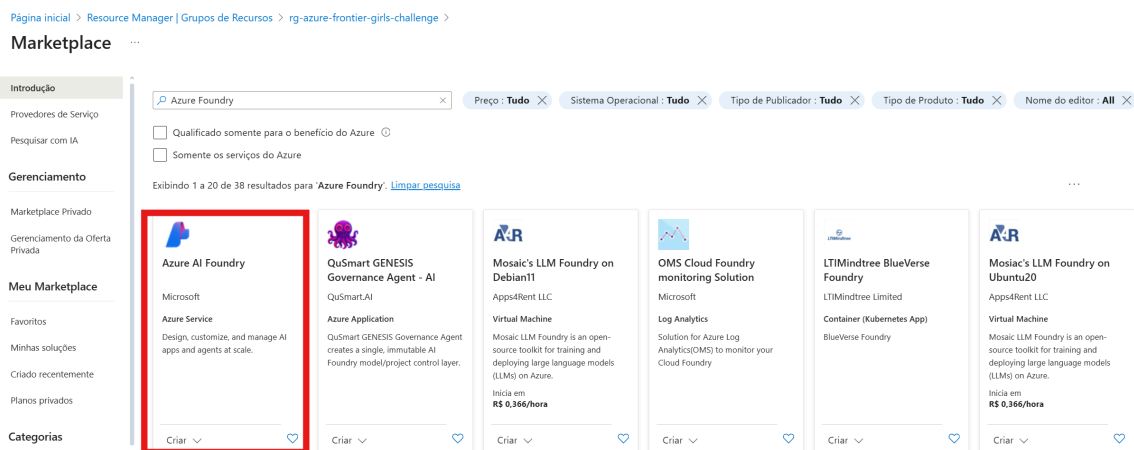
1. Crie um **Grupo de Recursos** no Pannel do Azure. Meu grupo de recursos teve o nome “rg-azure-frontier-girls-challenge”.



2. Crie um **Recurso**: Entre no seu grupo de recursos e crie um recurso.



3. No marketplace selecione **Azure Foundry AI** e configure-o de acordo com suas necessidades. No meu agente selecionei a localização “EAST US 2” para maior facilidade de acesso aos modelos.



4. Ao voltar para a tela de **Grupo de Recursos** e clicando no recurso criado, podemos acessar o **Portal do Azure Foundry**. Meu recurso teve o nome “b4388-mi0pgeii-eastus2”.

Página inicial > rg-azure-frontier-girls-challenge >

b4388-mi0pgeii-eastus2 Como usar este IA do Azure com Python Qual pacote npm devo instalar para usar este recurso de IA? Práticas recomendadas para gerenciar recursos de IA

Fábrica de IA do Azure

Pesquisar < > Go to Azure AI Foundry portal Excluir

Visão geral

- Log de atividade
- IAM (Controle de acesso)
- Marcações
- Diagnosticar e resolver problemas
- Visualizador de recursos
- Gerenciamento de Recursos
- Security
- Monitoramento
- Automação
- Ajuda

Fundamentos

Grupo de recursos ([mover](#)) : [rg-azure-frontier-girls-challenge](#) API Kind : AIServices

Assinatura ([mover](#)) : [Azure subscription 1](#) Localização : eastus2

ID da Assinatura : Status : Succeeded

Pontos de Extremidade : [Clique aqui para exibir os pontos de extremidade](#)

Marcações ([editar](#)) : [Adicionar marcas](#)

Build apps and agents in Azure AI Foundry

Unlock advanced AI capabilities, seamlessly integrate with your systems, and accelerate AI application development with Azure AI Foundry.

[Go to Azure AI Foundry portal](#)

5. Ao entrar no **Painel do Azure AI Foundry**, fiz o deploy de um modelo pela seção “Models + endpoints”, onde selecionei o **gpt-4o-mini** após clicar em **Deploy Model**. Selecionei o gpt-4o-mini não apenas por consumir menos créditos por token em comparação aos modelos 4.1 e 5.0 do gpt, mas também por oferecer um excelente equilíbrio entre custo, velocidade e qualidade de análise. Ele é otimizado para tarefas de compreensão e sumarização, entregando respostas consistentes e contextuais. Além disso, possui baixa latência, permitindo que o agente processe listas extensas de comentários de forma rápida. Quando for escolher seu modelo, verifique se ele está disponível na região do seu recurso.

Azure AI Foundry / b4388-mi0pgeii-eastus2_project / Models + endpoints

Unsupported browser: This site is supported and works best with the latest version of the new Microsoft Edge (Edge legacy is not fully supported), Google Chrome, Mozilla Firefox or Apple Safari. The site may not work correctly on other versions or browsers.

Model deployments [Help](#)

Model deployments Service endpoints

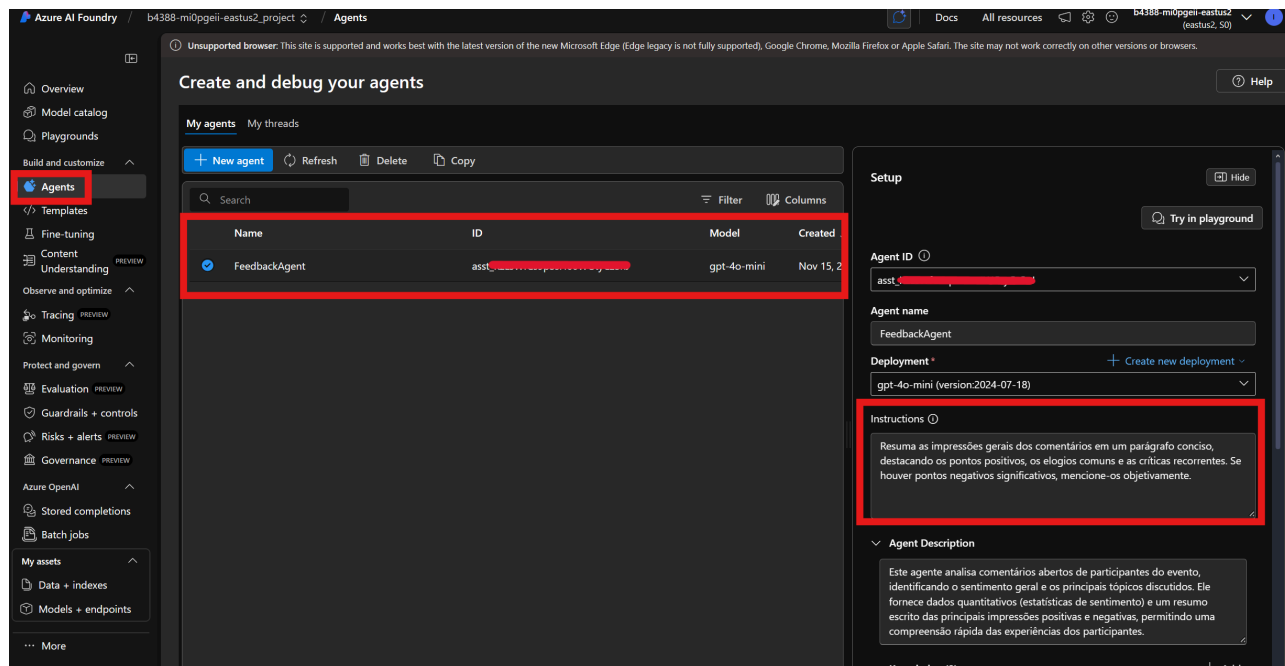
[+ Deploy model](#) [Refresh](#) [Edit](#) [Open in playground](#) [Reset view](#)

Name	Model name	Model version	State	Model retirement date	Content filter	Deployment type	Fine-tur
^ This resource							
gpt-4o-mini	gpt-4o-mini	2024-07-18	Succeeded	Feb 26, 2026 9:00 PM	DefaultV2	Global Standard	

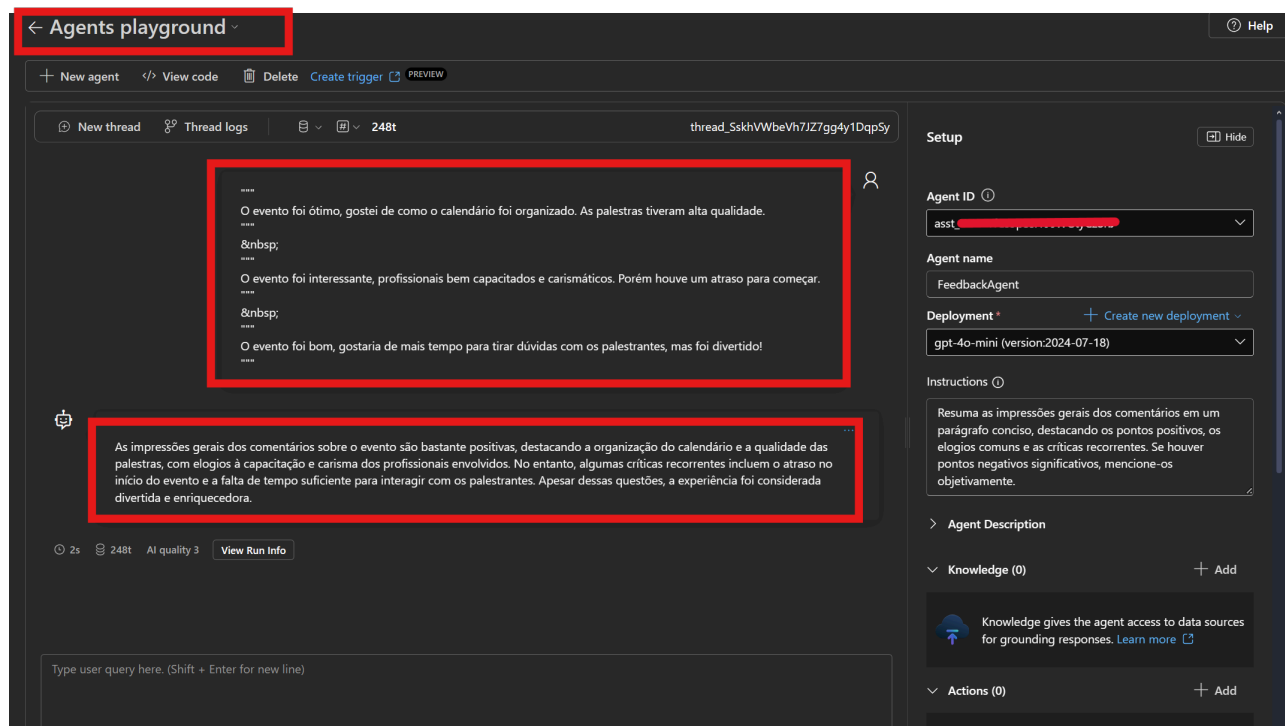
My assets

- Models + endpoints
- More

6. Configure o agente. Na seção **Agents**, nomeei meu agente de **FeedbackAgent** e configurei sua instrução como “Resuma as impressões gerais dos comentários em um parágrafo conciso, destacando os pontos positivos, os elogios comuns e as críticas recorrentes. Se houver pontos negativos significativos, mencione-os objetivamente”. Também adicionei uma descrição opcional para fins de organização.

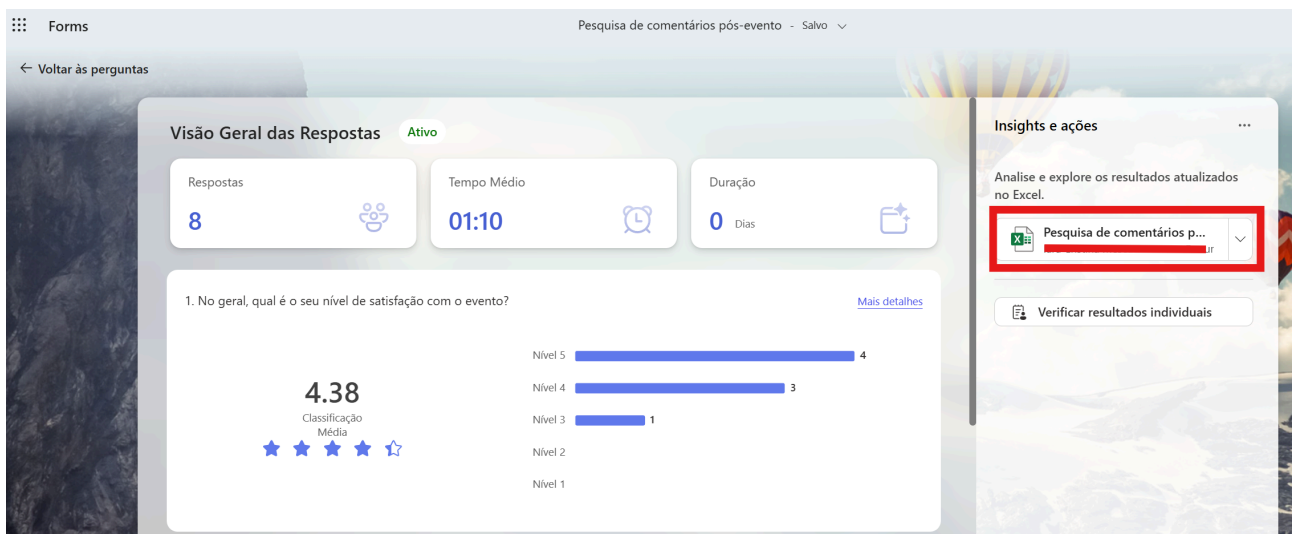


7. De preferência, teste se o agente está coerente em **Try in playground**. Em um chat, enviei 3 simulações de feedback de um evento separados por aspas e espaço, onde o agente deveria retornar um resumo das impressões dos participantes conforme a instrução dada. Note que a resposta do agente está de acordo com o **Input** e a **Instrução**. Esse agente pode economizar tempo de organizadores que precisam analisar opiniões e feedbacks de inúmeros usuários de uma única vez.



8. Selecione seus dados. Minha ideia principal era integrar o agente à respostas de um formulário. Então pensando no **Microsoft Forms**, isso pode ser feito rapidamente clicando em **Exibir respostas** e **Abrir Resultados no Excel**. Criei uma simulação de formulário abaixo para representar a situação quando você é proprietário no formulário.

The screenshot shows the Microsoft Forms interface for a survey titled "Pesquisa de comentários pós-evento". The survey is in Portuguese and asks for feedback on an event. The question is "1. No geral, qual é o seu nível de satisfação com o evento?". The response options are "Extremamente insatisfeito" (Extremely dissatisfied) and "Extremamente satisfeito" (Extremely satisfied), with five stars in between. The "Exibir respostas" (Show responses) button is highlighted with a red box.



Mas o agente também pode ser útil para outros tipos de formulário, feedbacks e comentários desde que estejam armazenados em um arquivo de texto. O código também poderia ser adaptado para receber outros tipos de arquivo além de CSV.

9. Faça a integração com Python ou a linguagem que for realizar sua automação. Criei o arquivo `app.py` com o aplicativo que gerasse a interface. Ela recebe um CSV na variável `upload_file`, o separador `separator` e o encoder `encoding`. A interface gera erros caso o usuário não preencha um dos campos corretamente e avisa o usuário caso a leitura do CSV falhe.

The screenshot shows the "Feedback Analyzer with Azure Foundry Agent" application interface. It features a dark theme and a sidebar with "Azure Agent Configuration" options. The main area has a "Upload your feedback CSV" section with a "Drag and drop file here" area, a "Browse files" button, and a "CSV Separator" dropdown set to ";". Below this is a "File Encoding" dropdown set to "windows-1252". A blue button at the bottom says "Please upload a CSV file to start. Default parameters are meant for Microsoft Forms exports."

É uma interface simples em streamlit que pode ser aberta localmente desde que o usuário tenha a biblioteca (e as outras dependências em “requirements.txt”) com o comando: `streamlit run app.py`. A imagem acima mostra a tela inicial e a imagem abaixo como ela fica ao selecionar um arquivo.

Azure Agent Configuration

AGENT_ID

PROJECT_ENDPOINT

If you prefer, set `AGENT_ID` and `PROJECT_ENDPOINT` as environment variables instead.

Quick run: `python -m streamlit run app.py`

Feedback Analyzer with Azure Foundry Agent

Upload your feedback CSV

Drag and drop file here

Limit 200MB per file • CSV

Browse files

pesquisa_de_comentarios_pos_evento.csv

1.7KB

X

CSV Separator

File Encoding

;

windows-1252

Preview of uploaded data (first 5 rows):

	Id	Hora de início	Hora de conclusão	Email	Nome	No geral, qual é o seu nível de satisfação com o evento?	Esta é a primeira vez que você participa de eventos como este?	Qual é a probabilidade de vc
0	1	15/11/2025 17:20	15/11/2025 17:23	anonymous	None	5	Sim	
1	2	15/11/2025 17:23	15/11/2025 17:24	anonymous	None	5	Não	
2	3	15/11/2025 17:24	15/11/2025 17:24	anonymous	None	3	Não	
3	4	15/11/2025 17:24	15/11/2025 17:25	anonymous	None	4	Não	
4	5	15/11/2025 17:25	15/11/2025 17:26	anonymous	None	4	Não	

Select column with feedback text

O que você mais gostou no evento?

Send to model

Quando selecionamos um arquivo, a interface mostra uma prévia dele em uma tabela para o usuário selecionar com segurança a coluna desejada. No caso do Microsoft Forms exportado para CSV, basta selecionar a pergunta na qual queremos obter todas as suas respostas. Para mais detalhes do código da interface veja o arquivo `app.py` no repositório.

10. Desenvolva uma função que use a API do Azure AI Foundry para chamar o seu agente. Crie a função `run_agent()` no arquivo `azure_agent_worker.py` para criar uma thread (baseado no **Endpoint do Projeto**) através das classes `azure.ai.agents` e `azure.identity` onde seleciono meu agente através do seu **ID** e envio o conteúdo a ser analisado pelo parâmetro “content”. Veja que antes o Input passa pela função `build_message()` que separa os comentários com aspas e espaço, uma abordagem eficiente de Prompt Engineering para ajudar o modelo a diferenciar um feedback do outro.

```
azure_agent_worker.py M X
azure_agent_worker.py > ...
1  import os
2  from typing import Optional
3  import pandas as pd
4  from azure.ai.agents import AgentsClient
5  from azure.identity import DefaultAzureCredential
6  from dotenv import load_dotenv
7
8  load_dotenv()
9
10 def build_message(data: list[str]) -> str:
11     quotes = '""'
12     separator = f"\n{quotes}\n\n{quotes}\n"
13     return quotes + "\n" + separator.join(data) + "\n" + quotes
14
15
16 def get_response(client: AgentsClient, thread_id: str) -> str:
17     # Messages are Listed from newest to oldest
18     for message in client.messages.list(thread_id):
19         if message.role == "assistant":
20             return message.content[0].text.value
21
22     raise RuntimeError("Could not find the assistant response message.")
23
24
25 def run_agent(
26     data: list[str],
27     agent_id: Optional[str] = None,
28     project_endpoint: Optional[str] = None,
29     temperature: float = 0.8,
30 ) -> str:
31     """Run the Azure Foundry agent with the provided list of texts and return its response.
32
33     This function reads `AGENT_ID` and `PROJECT_ENDPOINT` from the environment if not provided.
34     Raises ValueError when required configuration is missing, RuntimeError on auth or API failures.
35     """
36     # Load env again in case Streamlit changed it or user provided in UI
37     load_dotenv(override=True)
38     AGENT_ID = agent_id or os.getenv("AGENT_ID")
39     PROJECT_ENDPOINT = project_endpoint or os.getenv("PROJECT_ENDPOINT")
40
41     if not AGENT_ID or not PROJECT_ENDPOINT:
42         raise ValueError("Missing AGENT_ID or PROJECT_ENDPOINT environment variables.")
43
44     try:
45         credential = DefaultAzureCredential()
46         client = AgentsClient(PROJECT_ENDPOINT, credential)
47     except Exception as exc:
48         raise RuntimeError(f"Authentication failed: {exc}")
49
50     message = build_message(data)
51
52     try:
53         response = client.create_thread_and_process_run(
54             agent_id=AGENT_ID,
55             thread={"messages": [{"role": "user", "content": message}]},
56             temperature=temperature,
57             polling_interval=1,
58         )
59     except Exception as exc:
60         raise RuntimeError(f"Failed to send request to agent: {exc}")
61
62     try:
63         return get_response(client, response.thread_id)
64     except Exception as exc:
65         raise RuntimeError(f"Failed to get agent response: {exc}")
66
```

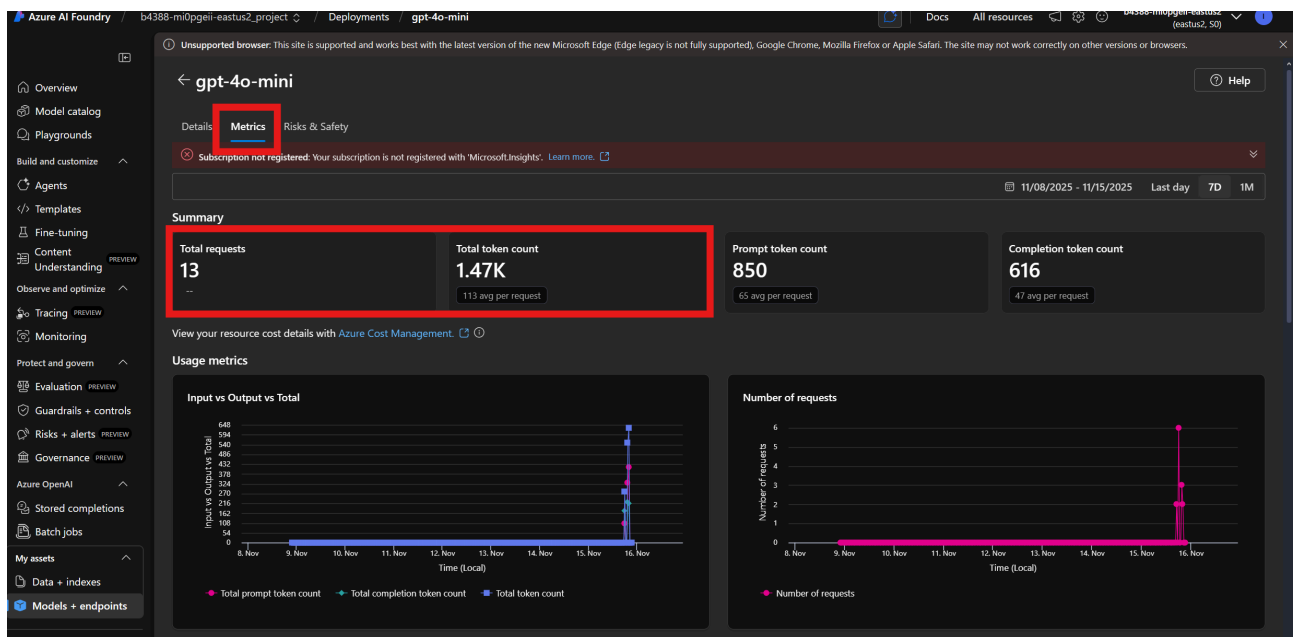
11. Integre a interface com o agente. Continuando o código da interface, caso o botão **Run model** seja pressionado, obteremos o **ID do Agente** e o **EndPoint do Projeto** que foram preenchidos nos campos da Barra lateral **Azure Agent Configuration**. Caso estejam vazios, serão utilizados as variáveis em `.env`. Depois chamaremos a função `run_agent()` para receber todos os comentários selecionados e retornar a resposta do modelo contida na variável `result`. Exemplo de uso está no README

```
if run_button:
    try:
        texts = df[selected_col].dropna().astype(str).tolist()
        if len(texts) == 0:
            st.error("Selected column is empty. Choose another column with text values.")
            st.stop()
    except Exception as e:
        st.error(f"Failed to extract column data: {e}")
        st.stop()

    # Use sidebar values if provided, otherwise rely on environment variables
    agent_id = agent_id_input.strip() or None
    project_endpoint = project_endpoint_input.strip() or None

    with st.spinner("Sending data to the model and waiting for response..."):
        try:
            result = run_agent(texts, agent_id=agent_id, project_endpoint=project_endpoint)
            st.success("Model returned a response:")
            st.code(result, "markdown", wrap_lines=True)
        except Exception as e:
            st.error(e)
```

12. Usei a subseção **Metrics** na seção **Models + endpoints** para acompanhar as requests de testes e tokens gastos pelo meu modelo durante testes. É importante não fazer requisições em massa para não consumir créditos desnecessariamente.



Limitações e Futuras Expansões

Atualmente, o projeto está limitado à análise de comentários textuais extraídos de arquivos CSV, e depende da qualidade do modelo de linguagem selecionado, que pode apresentar limitações na identificação de sarcasmo, ironia ou contextos muito específicos. Futuramente, o sistema pode ser expandido para aceitar outros formatos de entrada (como integrações diretas com Google Forms ou Microsoft Forms via API), gerar relatórios detalhados com os gráficos mais relevantes e realizar análises por categorias ou segmentos, processar grandes volumes de dados de forma escalável e permitir feedbacks individuais automatizados para cada participante quando há poucos comentários. Também é possível integrar análise de sentimentos multimodal (texto + áudio ou imagem) para situações específicas.