**Lab 4: What's Your Vector, Victor?**

Now it's time to see the end-to-end process in action. In this lab , you will load new data using the data loading mechanism you created in a previous lab , then observe the automatic vectorization in action. Once you ingest the new data, ask a question through the prompt interface and see if it returns an answer about the new data you loaded.

## **Objective :**

* Understand the steps involved in making new data ready for LLM interaction.
* Explore how to elicit different response formats from the LLM.

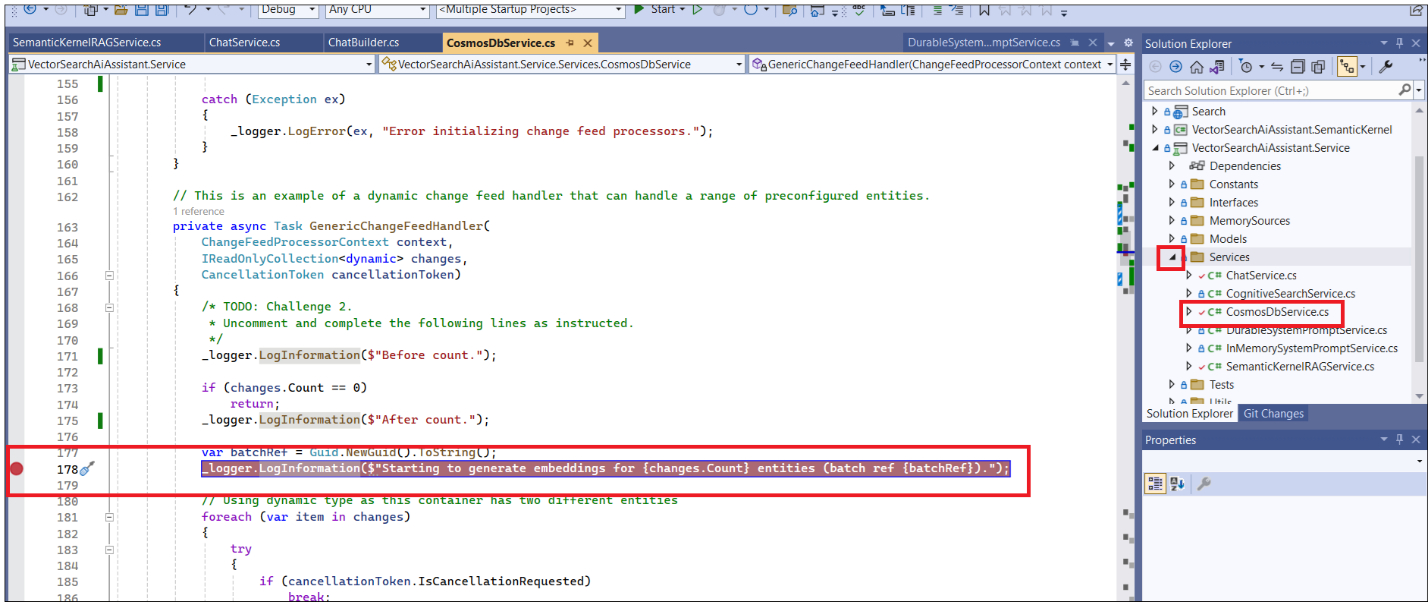
## **Task 1: Create and load new data**

1. We will create embeddings and store them in Cognitive search. To know the process, we will apply some breakpoints in the Cosmos DB change feed process code before we add documents to the Azure Cosmos DB account.
2. **Go to Visual Studio and run your solution.** Ask your interface to **provide product details of FR-M94B-46**. The interface doesn’t have product information for FR-M94B-46 right ?.

**A computer screen shot of a computer

Description automatically generated**

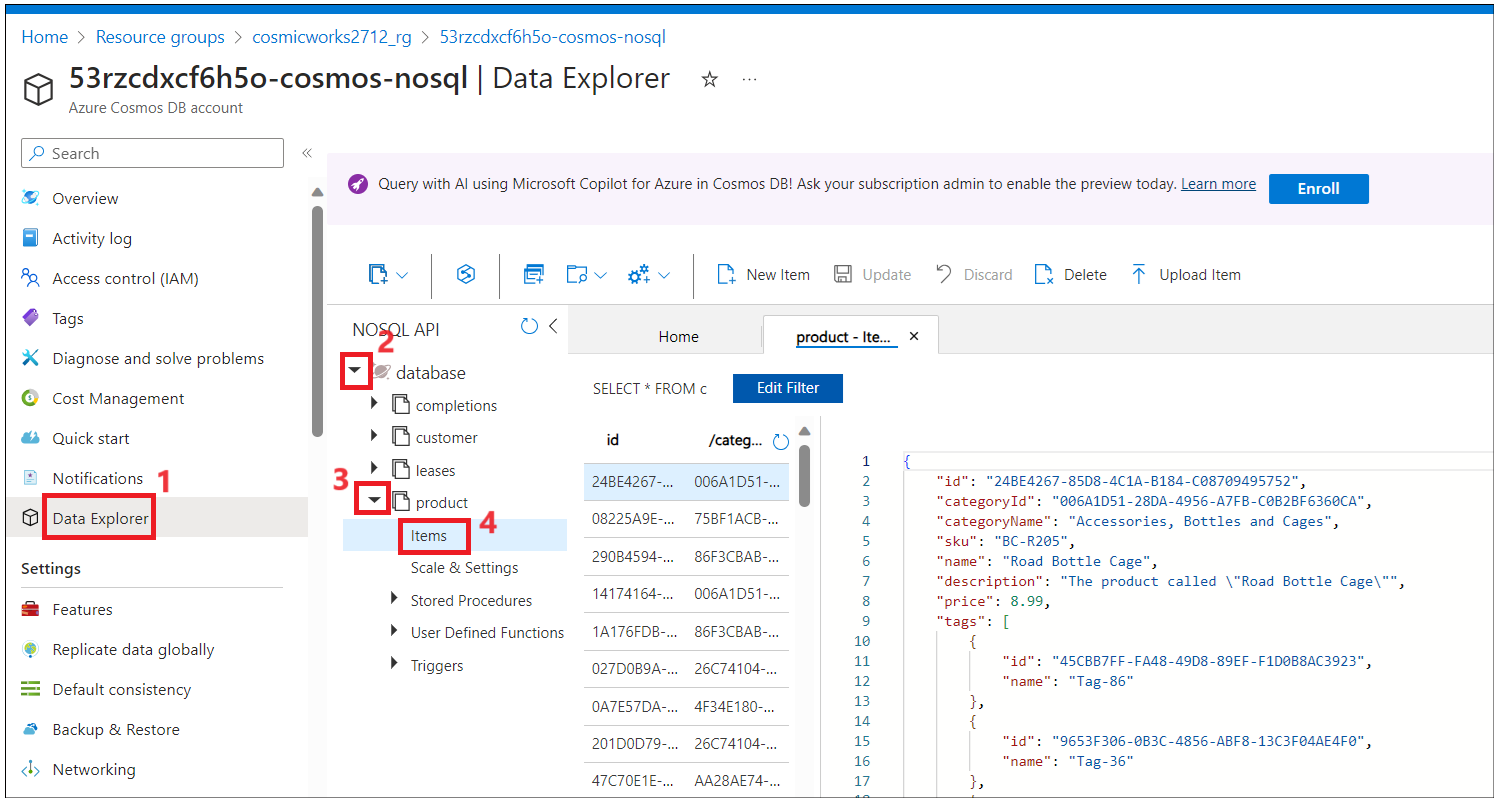
1. Stop the solution and we will add the product details to the container.
2. Switch back to visual Studio, open **CosmosDbService.cs** file from **VectorSearchAiAssistant.Service -> Services. Go to** GenericChangeFeedHandler method and keep breaktpoint LogInformation to activate breakpoint whenever a new document is added to the Cosmos DB product container.

****

1. Switch back to the **Azure portal** and click on your **Resource group -> Azure Cosmos DB account.**



1. Click on **Data Explorer** from the left navigation menu expand **database-> Products** and then click on **Items** as shown in the below image.



1. Click on **New Item** and then replace the code with the below code and **Save** it.

{

"id": "ABBB5A68A-6284-4DC7-B48D-232F347CA827",

"categoryId": "3B75F02D-6443-4C83-B182-8BB38192C33B",

"categoryName": "Components, Mountain Frames",

"sku": "FR-M94B-46",

"name": "HL Mountain Frame - Black, 46",

"description": "The product called \"HL Mountain Frame - Black, 46\"",

"price": 1349.6,

"tags": [

{

"id": "3BFB03A9-3106-44C7-823A-DB1A67E283C3",

"name": "Tag-47"

},

{

"id": "54C5E2EB-EE2D-496D-8AE2-200D7575968A",

"name": "Tag-156"

},

{

"id": "7DF71D87-FB6F-498B-9D7B-E7EBE40350E1",

"name": "Tag-88"

},

{

"id": "A0BA4E3B-AD4A-42AF-BFA4-5F48E2E57F07",

"name": "Tag-58"

},

{

"id": "A9834752-41CA-47F5-8A5A-D9A878DF0ACB",

"name": "Tag-198"

}

],

"\_rid": "F896AJPj0PmcAQAAAAAAAA==",

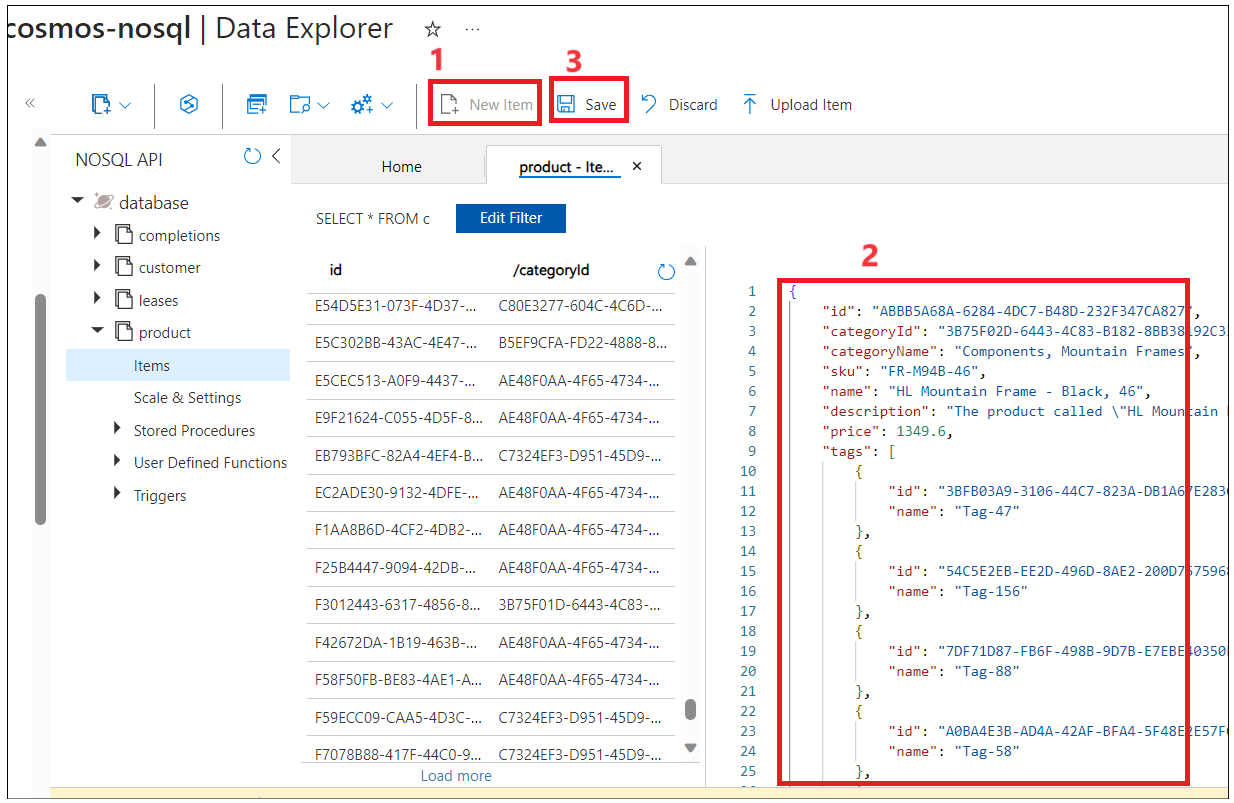
"\_self": "dbs/F896AA==/colls/F896AJPj0Pw=/docs/F896AJPj0PmcAQAAAAAAAA==/",

"\_etag": "\"7e005amc-0000-0100-0000-65b61dce0000\"",

"\_attachments": "attachments/",

"\_ts": 1706433998

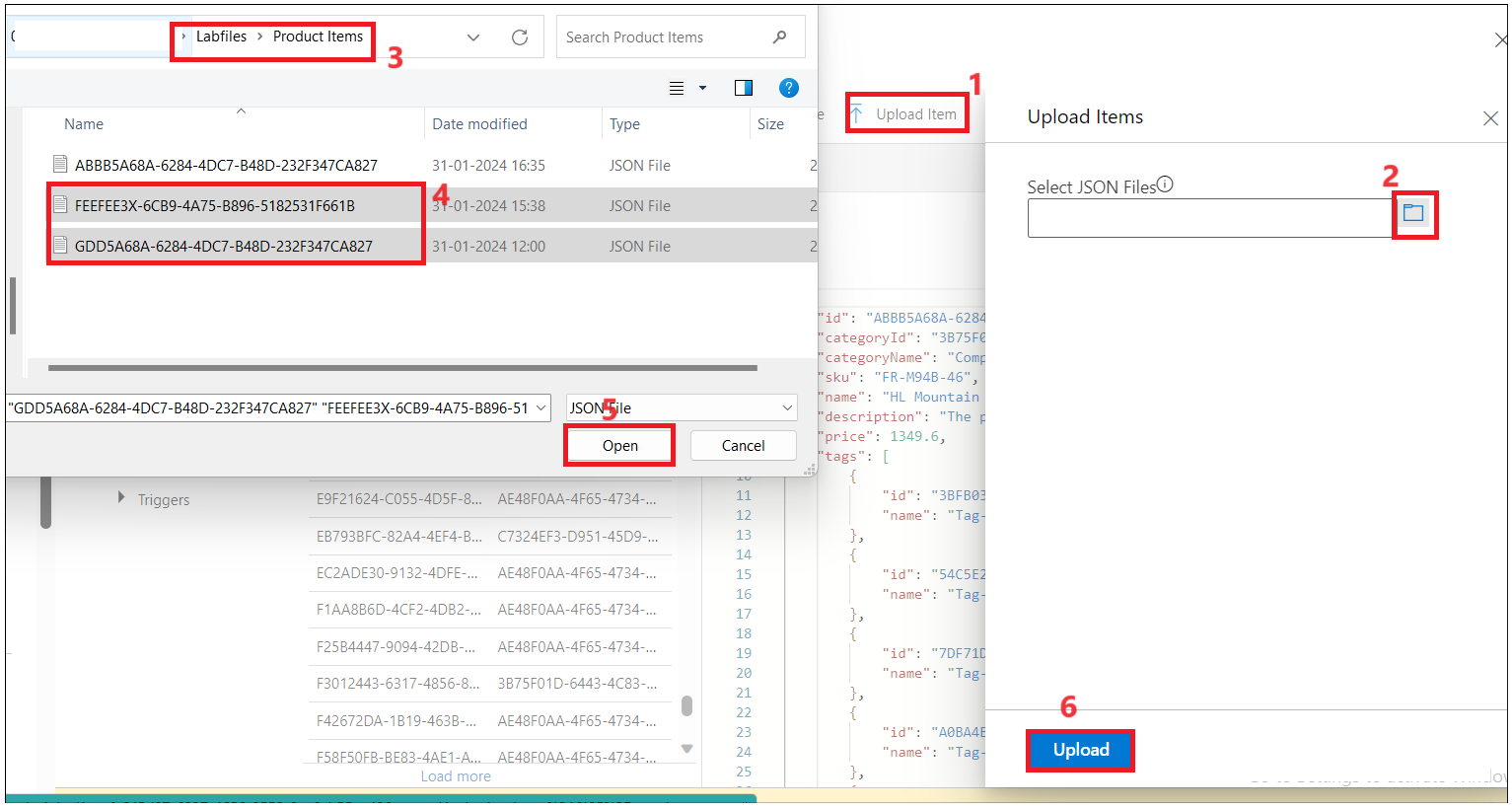
}



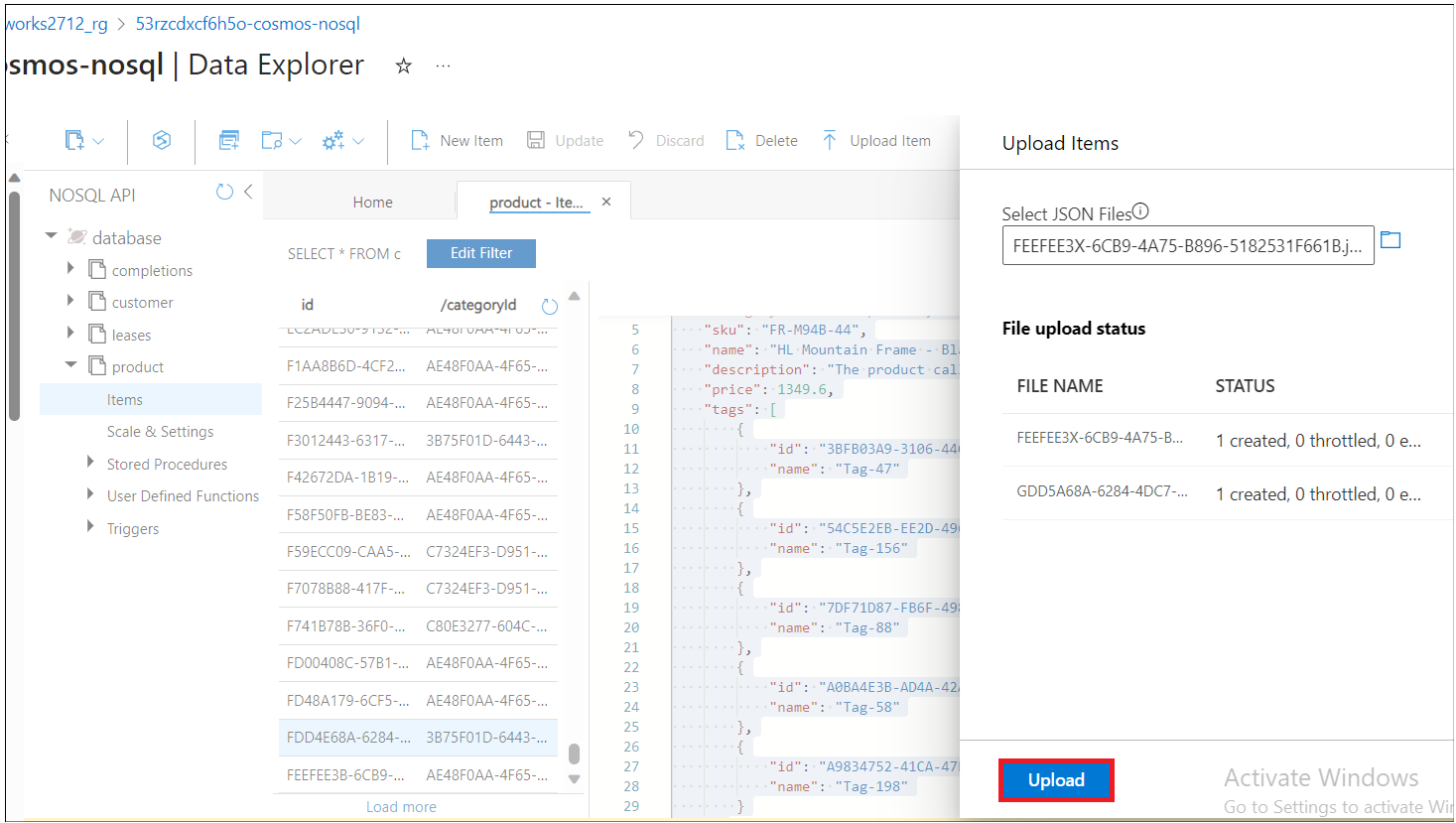
1. Click on **Upload items**. Click on **Select JSON Files Upload Items** folder and browse to **C:\Labfiles\Product Items\** and select two files, click on **Open,** and then click on **Upload** button.

Note: Ignore if you see any error. You can manually also add each item as we did in the previous step. A screenshot of a computer

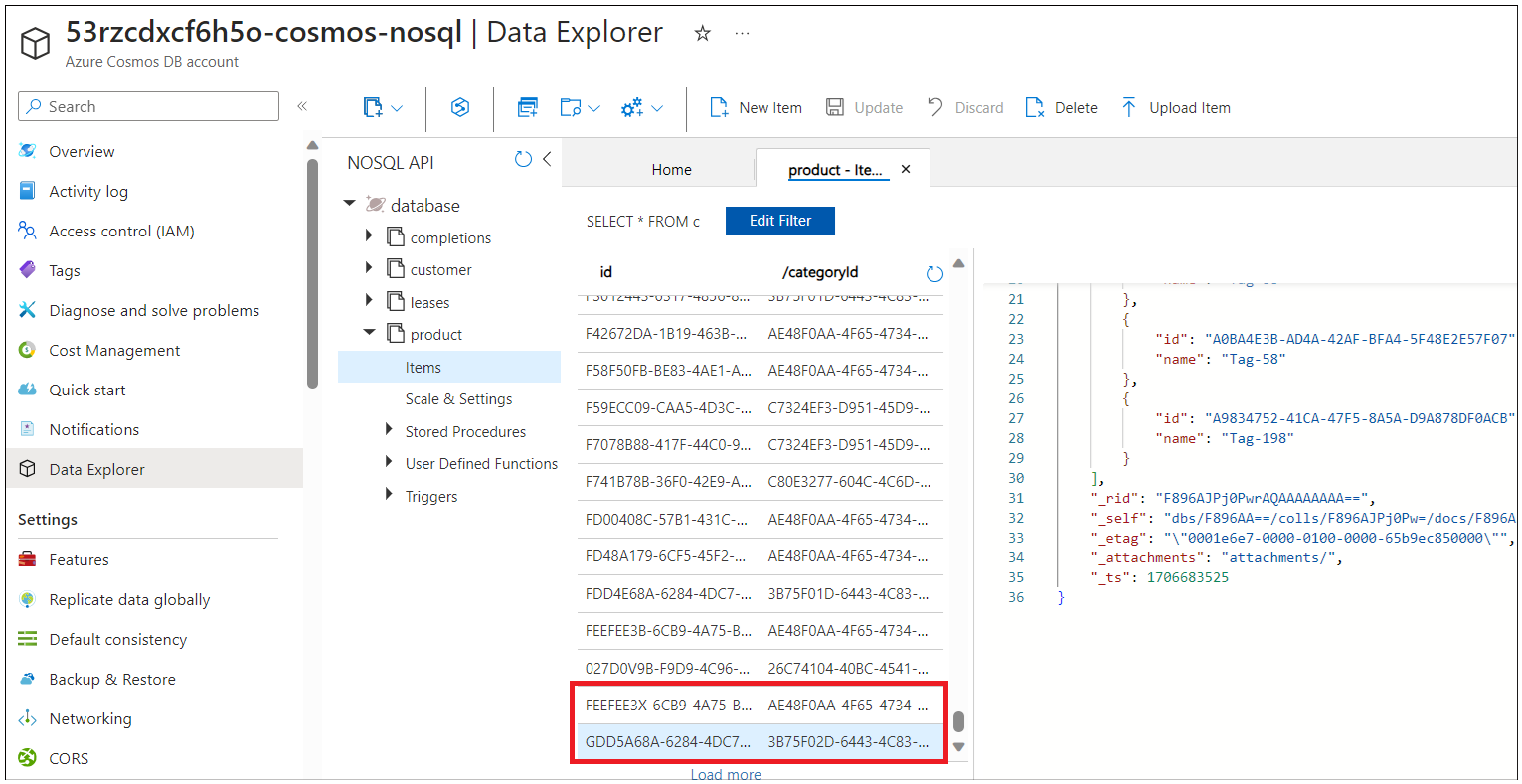
Description automatically generated



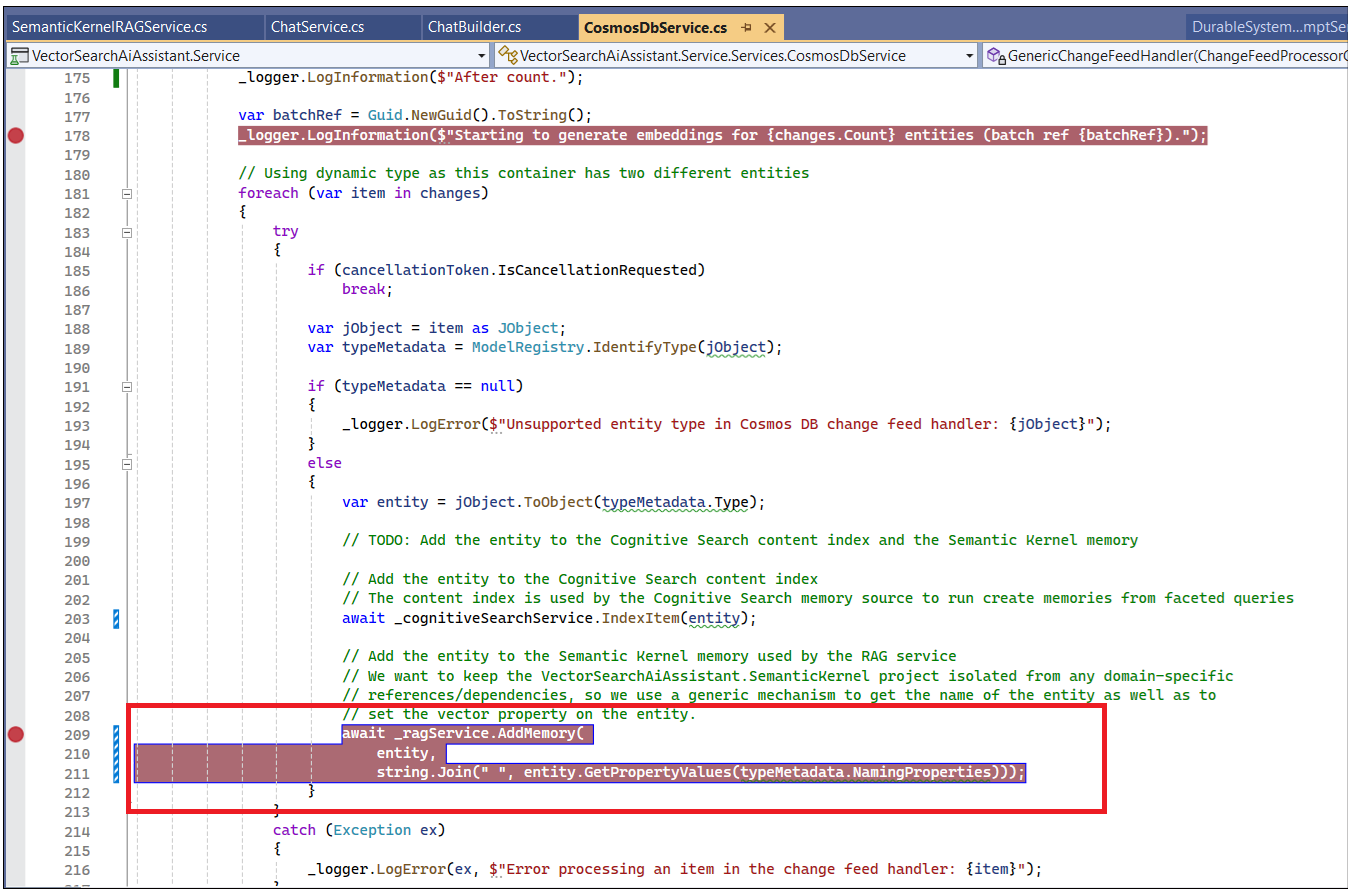
1. Click on **Upload**.

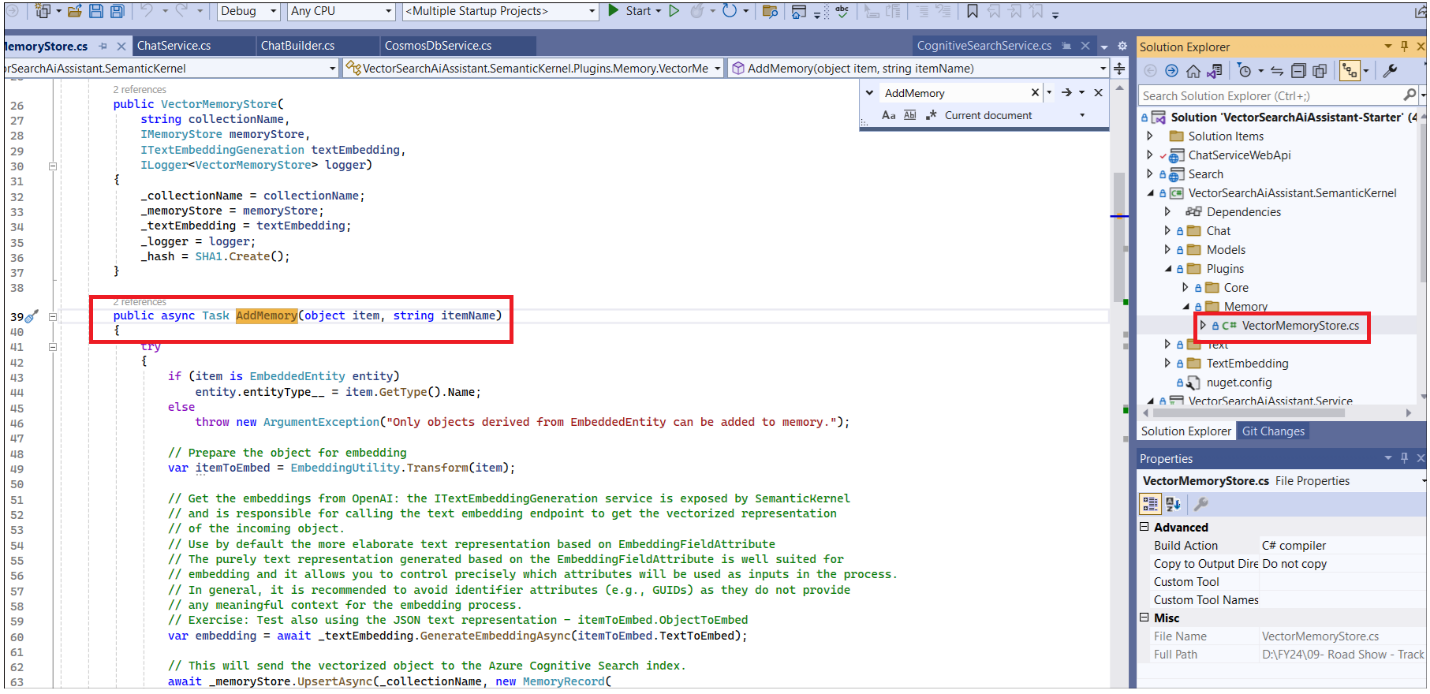


1. Make sure the items are added to the list of items. AS soon as you save ,code in change feed process will trigger.

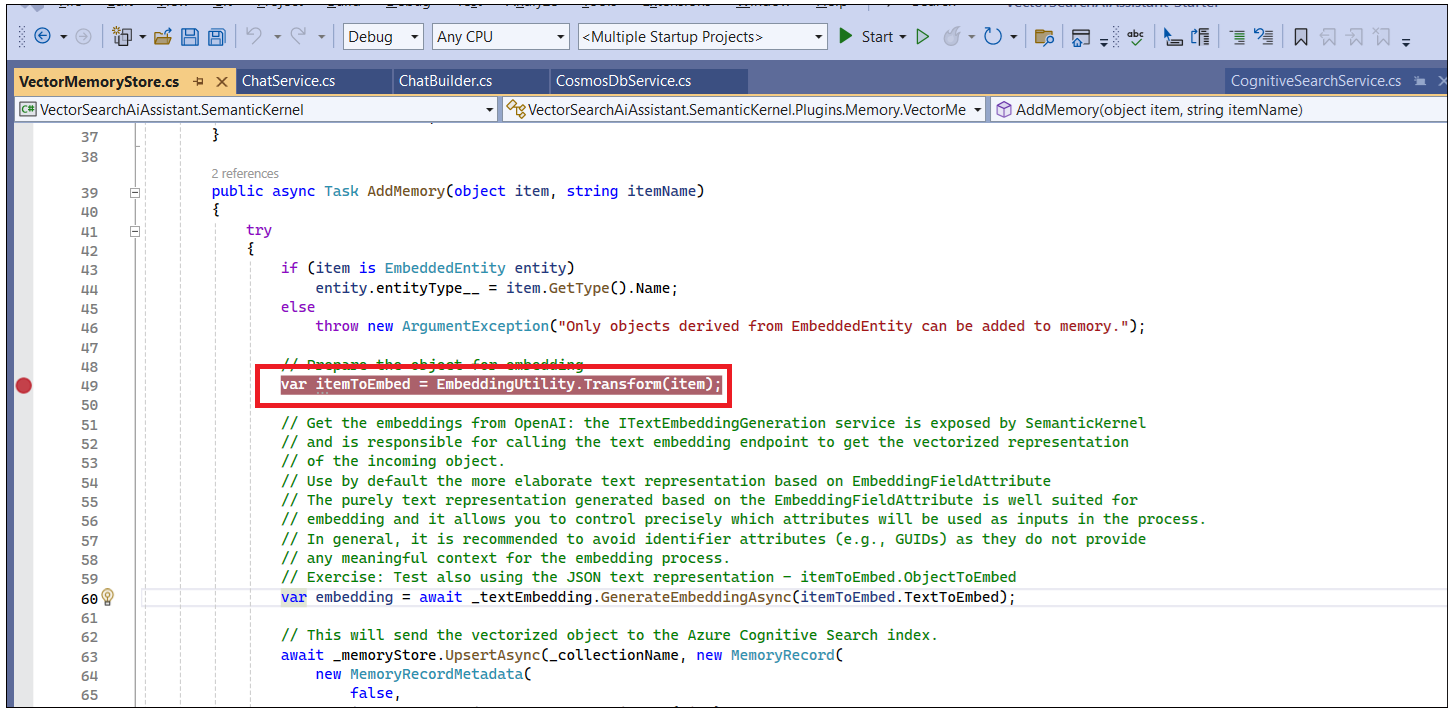


1. Switch back to **CosmosDbService.cs** file in Visual studio. Put breakpoint at await \_ragService.AddMemory function. **ragService.AddMemory** inter calls AddMemory function from VectorMemoryStore.cs in VectorSearchAiAssistant.SemanticKernel ,it process the documents and add them to the Azure cognitive search for further processing.

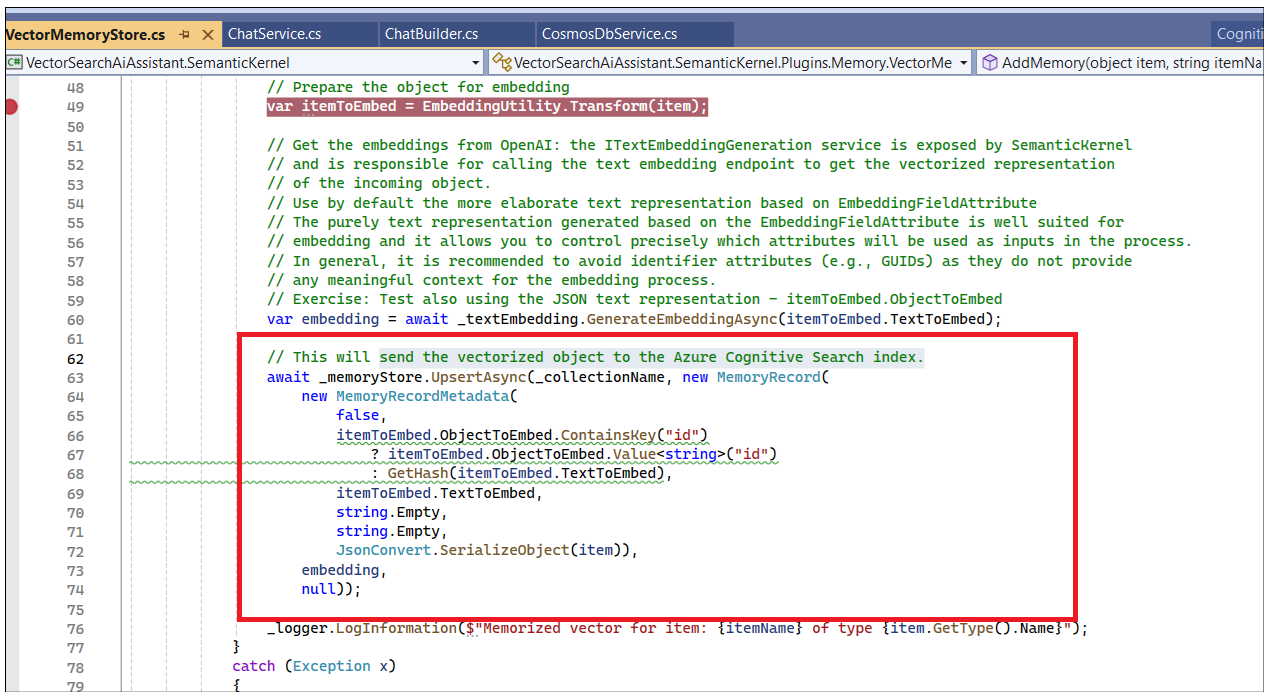
****

****

1. We receive a cognitive search document and convert it into strong in **itemToEmbed to send for embedding**.



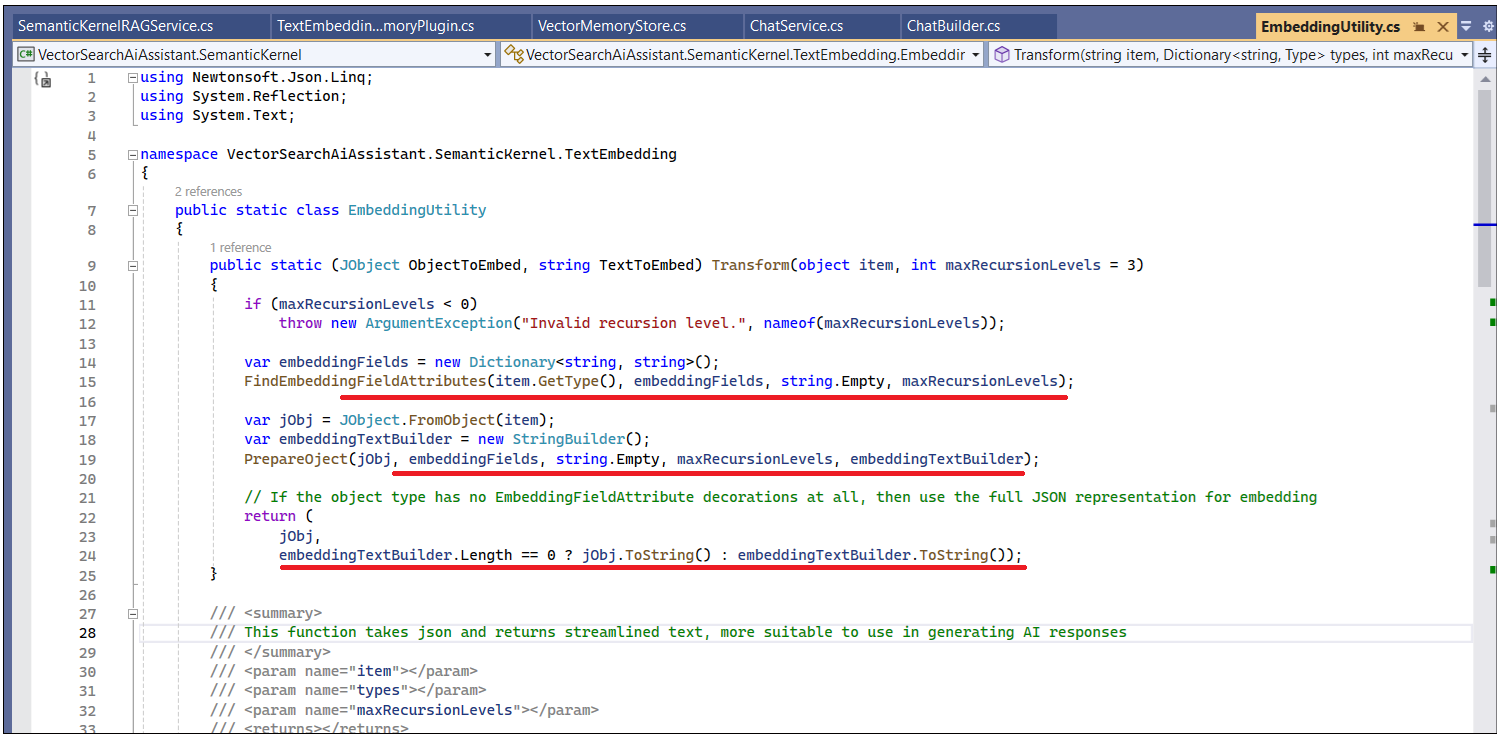
1. await \_memoryStore.UpsertAsync will send the vectorized object to the Azure Cognitive Search index.



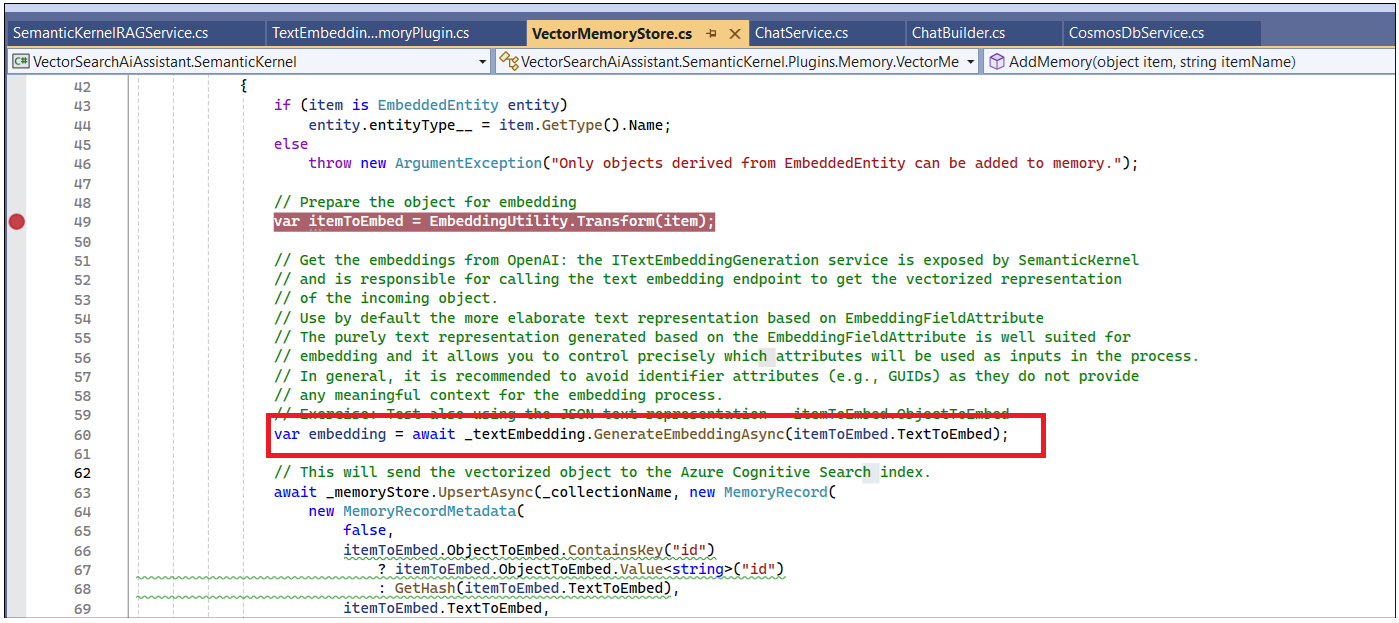
1. Open EmbeddingUtility function,



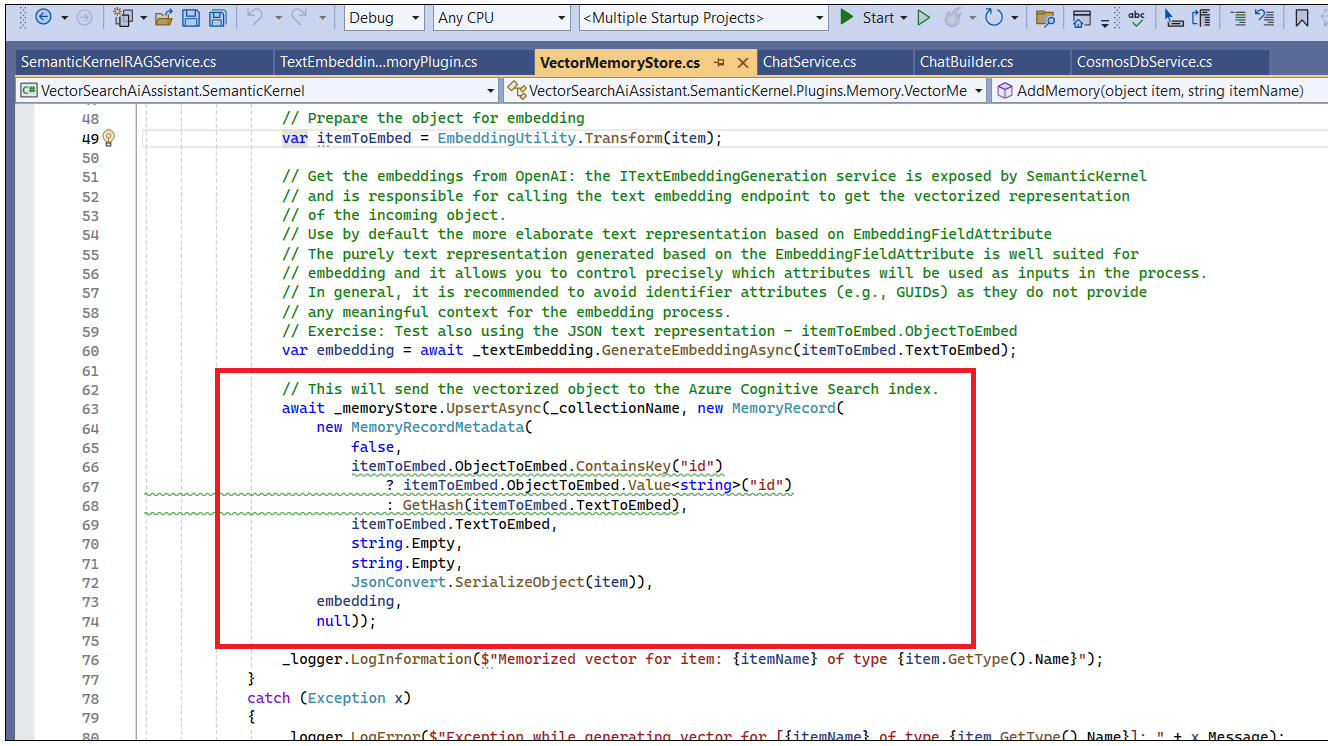
1. In **EmbeddingUtility** function ,check the attributes of the fields to be embedded and generate a string out of it. if the object type has no **EmbeddingFieldAttribute** decorations at all, then use the full JSON representation for embedding



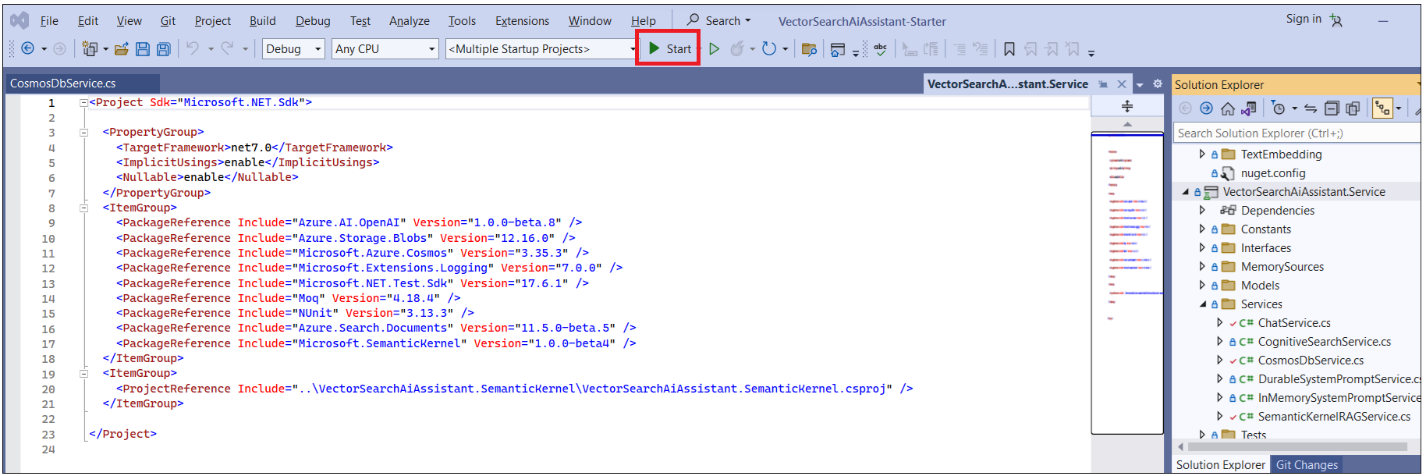
1. We are converting string into an embeddings by calling OpenAI. GenerateEmbeddingAsync function is used to return an array of points.



1. Finally, we will send the vectorized object to the Azure Cognitive Search index. Once its done then you can search documents in chat interface.



1. Run the VectorSearchAiAssistant.Service project to vectorize and add the new items to the index. Running this project will initialize the Cosmos DB change feed handlers and start the vectorization process. The vectorization process will take a few minutes to complete.



1. Check the Cognitive Search index to validate that the new items were added. Ask in Interface the same question ++**provide product details of FR-M94B-46++**

A screenshot of a computer

Description automatically generated

## **Task 2: Experiment using prompts**

1. Click on **Create a New chat** and ask below questions to respond with a single number or with one or two words

How many racking socks products do you have?

How many racing socks do you have? Respond with a single number.

How many racing socks do you have? Respond with a single number and nothing else.

A screenshot of a computer

Description automatically generated

1. Click on **Create a New chat** and ask below questions to respond with a bulleted list or formatted a certain way

Which products are for racing? Format the response as a JSON file.

A screenshot of a computer

Description automatically generated

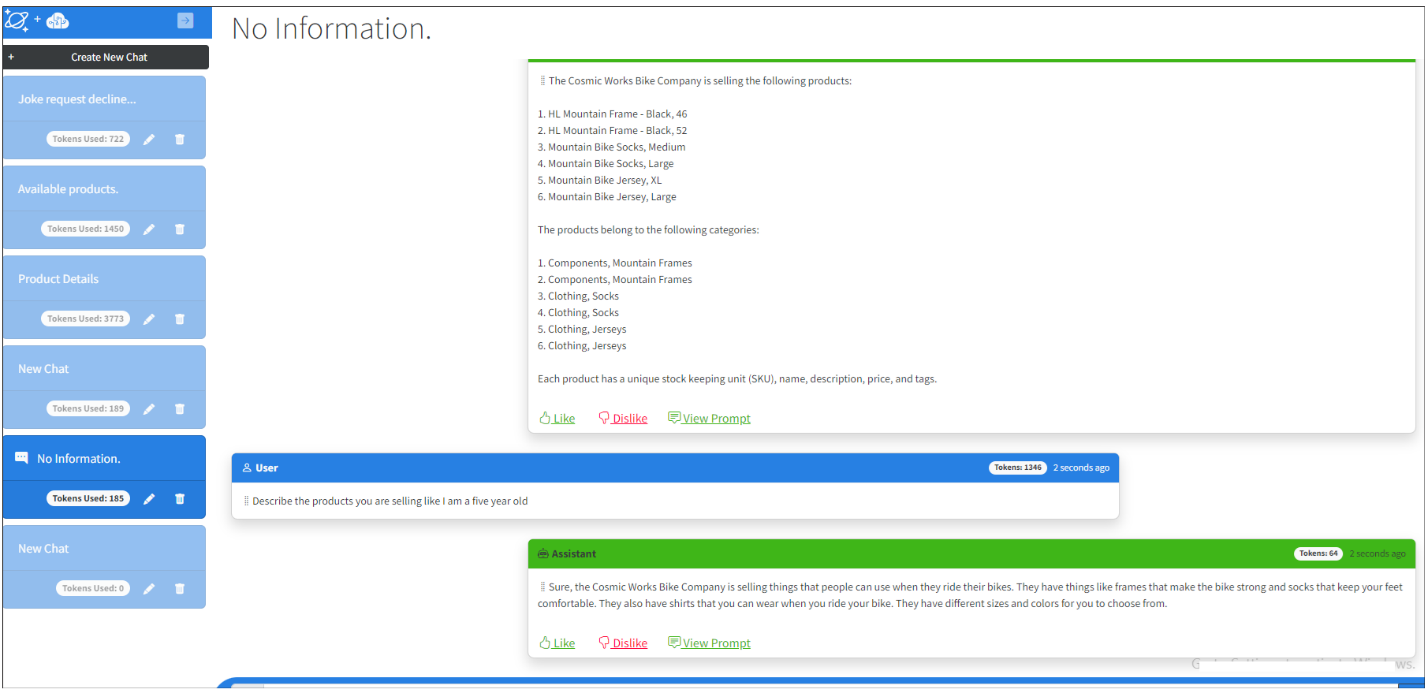
1. Click on **Create a New chat** and ask below questions to respond using simpler syntax (e.g. explain it like I'm five)

**Describe the products you are selling**

**+Describe the products you are selling like I am a five year old**

**A screenshot of a computer

Description automatically generated**

****

1. Challenge the model with prompts that require reasoning or a chain of thought. For example, ask it to calculate aggregates on the data or go further and give some word problems like those you had in school.

**Which are the products categories that are available?**

**Which are the products categories that are available? Which category has the least products?**

**What is the average number of products per category of products?**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

1. Challenge the model to explain its reasoning. Ask below question

**Which are the products categories that are available? Which category has the least products? Explain how you reached the conclusion.**

A screenshot of a computer

Description automatically generated

1. Request that the model list its sources. Ask below question

Which are the product categories? Provide details on the sources of data you are using to answer.

A screenshot of a computer

Description automatically generated

**Summary**

• We created new data and then chat history showing how it responded using the new data as context.

• located the new product or customer data which was loaded in the Cognitive Search Index and in Cosmos DB.