**Step 3: Generate Embeddings**

1. **Integrate OpenAI API:**
   * Add the OpenAI NuGet package to your project.
   * Set up the API key and necessary configurations.
2. **Create Embeddings:**
   * Write code to generate embeddings for your set of documents.
   * Use the OpenAI API to process words, phrases, sentences, paragraphs, and entire documents to create embeddings.

**Step 4: Perform Semantic Similarity Analysis**

1. **Word and Phrase Similarity Analysis:**
   * Select pairs of phrases to analyze (e.g., “Angela Merkel” vs. “Government”, “Cristiano Ronaldo” vs. “Government”).
   * Compute the similarity scores for these pairs using embeddings and a similarity metric (e.g., cosine similarity).
2. **Document-Level Comparisons:**
   * Choose a set of documents on related and unrelated topics.
   * Compute the similarity scores between these documents.

**Step 5: Data Visualization**

1. **Generate CSV Files:**
   * Export the similarity metrics into CSV files for further analysis.
2. **Create Illustrative Diagrams:**
   * Use external tools like Microsoft Excel or Tableau to create diagrams that visualize the similarity metrics.
   * Highlight key insights and variations in similarity across different domains and contexts.

**Step 6: Develop a Reproducible Codebase**

1. **Write Clean, Well-Documented Code:**
   * Ensure your code is easy to understand and maintain.
   * Include comments and documentation to explain the methods and algorithms used.
2. **Export Results:**
   * Implement functionality to export similarity metrics in CSV format.

**Step 7: Document the Methodology**

1. **Describe the Methods and Algorithms:**
   * Clearly document the methods and algorithms used for semantic similarity computation.
   * Explain the choice of metrics (e.g., cosine similarity) and any pre-trained models or algorithms applied (e.g., Word2Vec, BERT).

**Step 8: Demonstration and Validation**

1. **Use Recognized Examples:**
   * Use examples involving widely recognized names, phrases, and documents to illustrate your findings.
2. **Showcase Versatility:**
   * Demonstrate the versatility of semantic similarity metrics across diverse scenarios.

**Final Thoughts**

* **Iterate and Improve:**
  + Regularly review and refine your code and documentation.
  + Ensure your findings are clear and well-supported by your analysis.

Good luck with your project! If you have any specific questions or need further assistance, feel free to ask.

**Step-by-Step Guide to Creating Embeddings**

#### Step 3: Write Code to Generate Embeddings

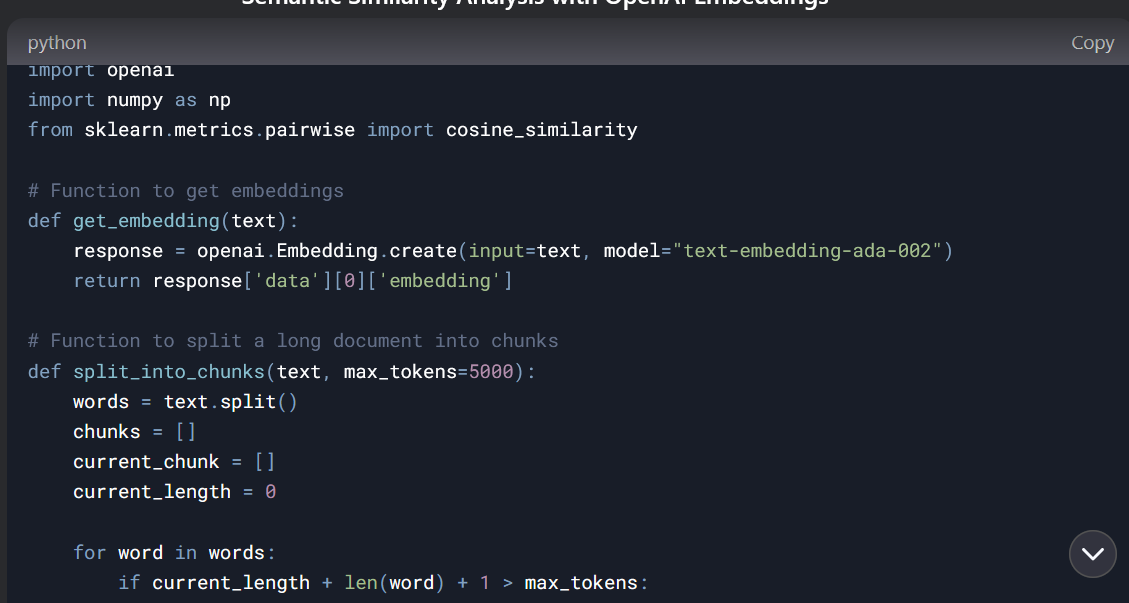
1. **Create a Helper Class for OpenAI API Integration:**
   * Create a new class named OpenAIHelper.
   * Add methods to interact with the OpenAI API and generate embeddings.
2. **Example Code:**

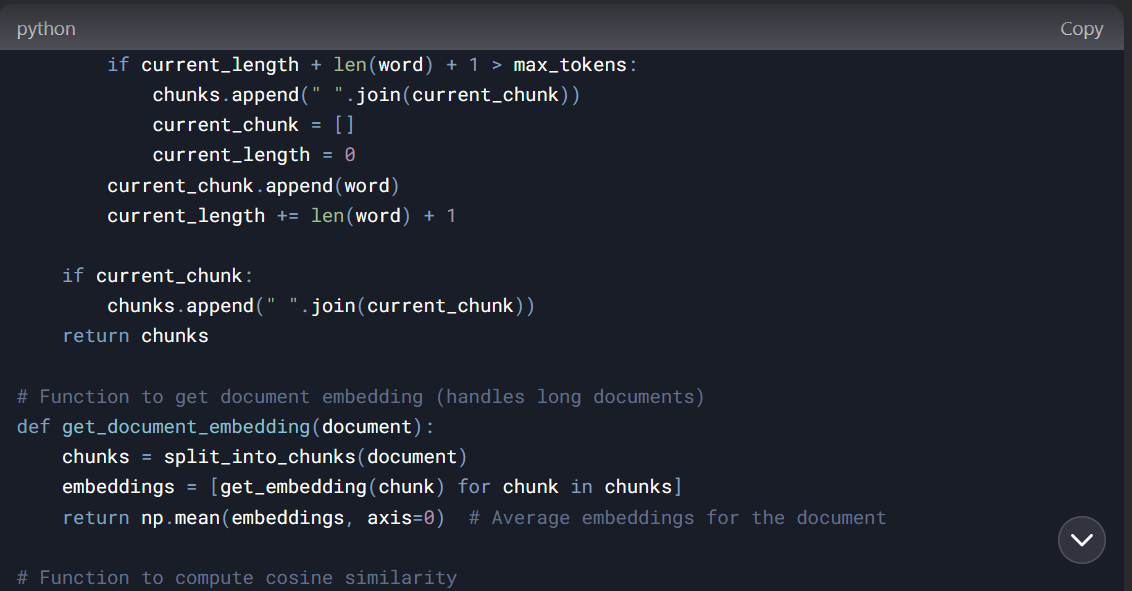
**Example Workflow for Long Documents**

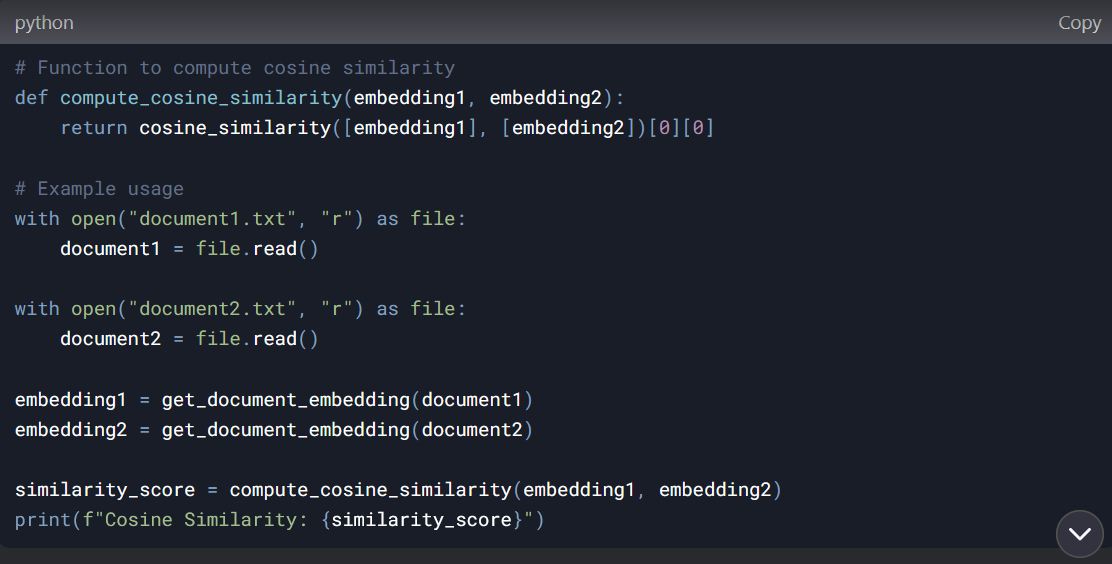
1. **Input**:
   * Two long documents (e.g., .txt files).
2. **Preprocessing**:
   * Read the content of the files.
   * If the content exceeds the token limit, split it into smaller chunks.
3. **Generate Embeddings**:
   * For each chunk, generate an embedding using the OpenAI API.
   * If the document was split into multiple chunks, average the embeddings to get a single embedding vector for the entire document.
4. **Compute Similarity**:
   * Compute the cosine similarity between the embeddings of the two documents.
5. **Visualization**:
   * Plot the similarity score on a chart, as described in your original approach.

**Example Code for Long Documents**

Here’s how you can handle long documents:

****

****

****

Enter Source Input (write done when you finish)

G:\FUAS\SE\semantic-similarity-analysis\SemanticSimilarity\SemanticSimilarity\Input\Sources\file1.txt  
G:\FUAS\SE\semantic-similarity-analysis\SemanticSimilarity\SemanticSimilarity\Input\Sources\file2.txt

G:\FUAS\SE\semantic-similarity-analysis\SemanticSimilarity\SemanticSimilarity\Input\Sources\file2.txt  
done

Enter Reference input (write done when you finish):

G:\FUAS\SE\semantic-similarity-analysis\SemanticSimilarity\SemanticSimilarity\Input\Sources\file1.txt  
G:\FUAS\SE\semantic-similarity-analysis\SemanticSimilarity\SemanticSimilarity\Input\Sources\file2.txt

G:\FUAS\SE\semantic-similarity-analysis\SemanticSimilarity\SemanticSimilarity\Input\Sources\file2.txt

Done