

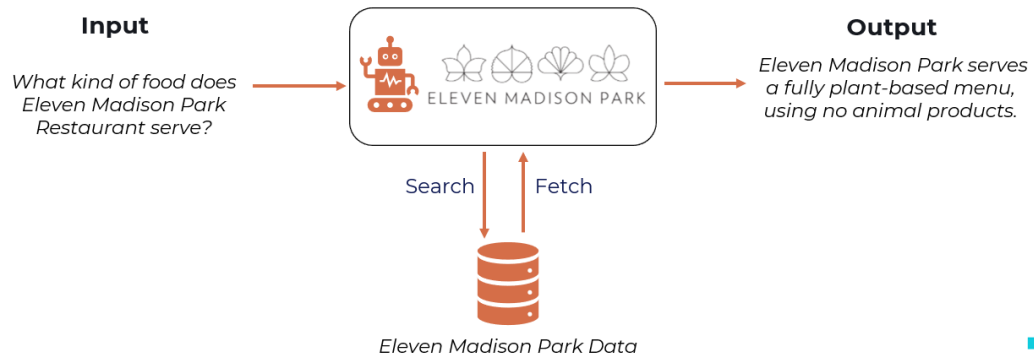
# TASK 1: PROJECT OVERVIEW & KEY LEARNING OBJECTIVES



## Project Overview



- In this project, we'll build an AI assistant that can answer questions about a specific topic (the Eleven Madison Park restaurant) based on provided documents in a database.
- This technique is called Retrieval-Augmented Generation (RAG).



© All rights reserved for Dr. Ryan Ahmed @Stemplicity Inc.

## Key Learning Outcomes



© All rights reserved for Dr. Ryan Ahmed @Stemplicity Inc.

Understand the core principles of Retrieval-Augmented Generation (RAG).

Use LangChain to streamline and orchestrate the RAG workflow.

Load, preprocess, and split text documents for optimal performance.

Generate high-quality embeddings using OpenAI models.

Store and search embeddings efficiently with ChromaDB as your vector database.

Build a powerful Retrieval-Based Question Answering (QA) chain using LangChain.

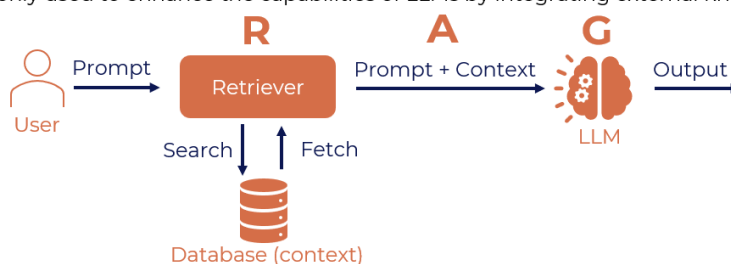
Design an interactive Gradio UI that displays answers along with source citations for transparency.

## TASK 2: UNDERSTAND RETRIEVAL AUGMENTED GENERATION (RAG)

### Retrieval Augmented Generation (RAG)



- RAG is a technique that combines retrieval-based systems with generative AI models.
- It's commonly used to enhance the capabilities of LLMs by integrating external knowledge sources.

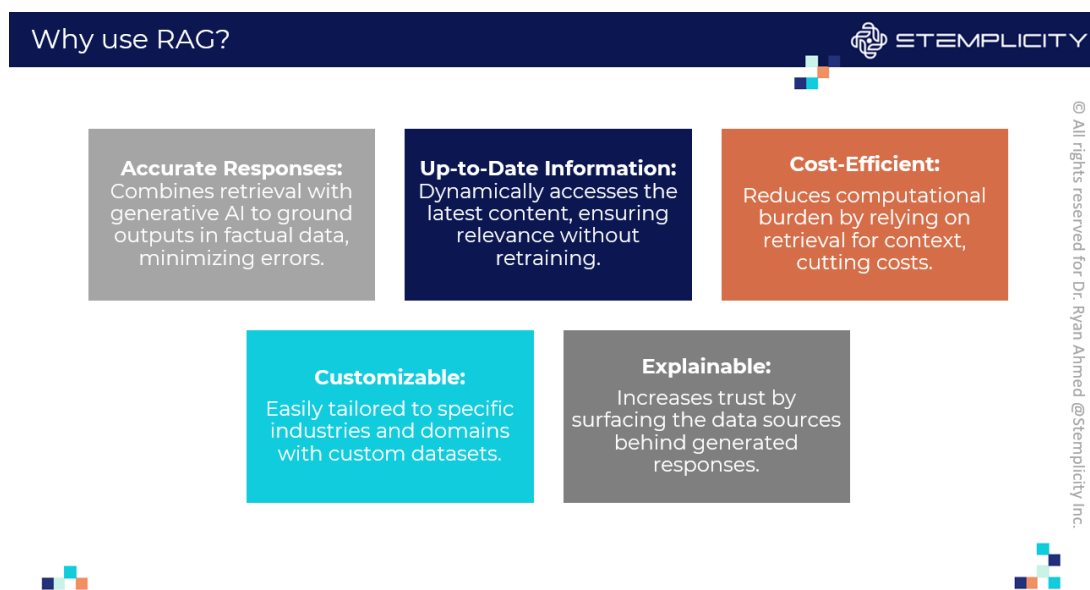
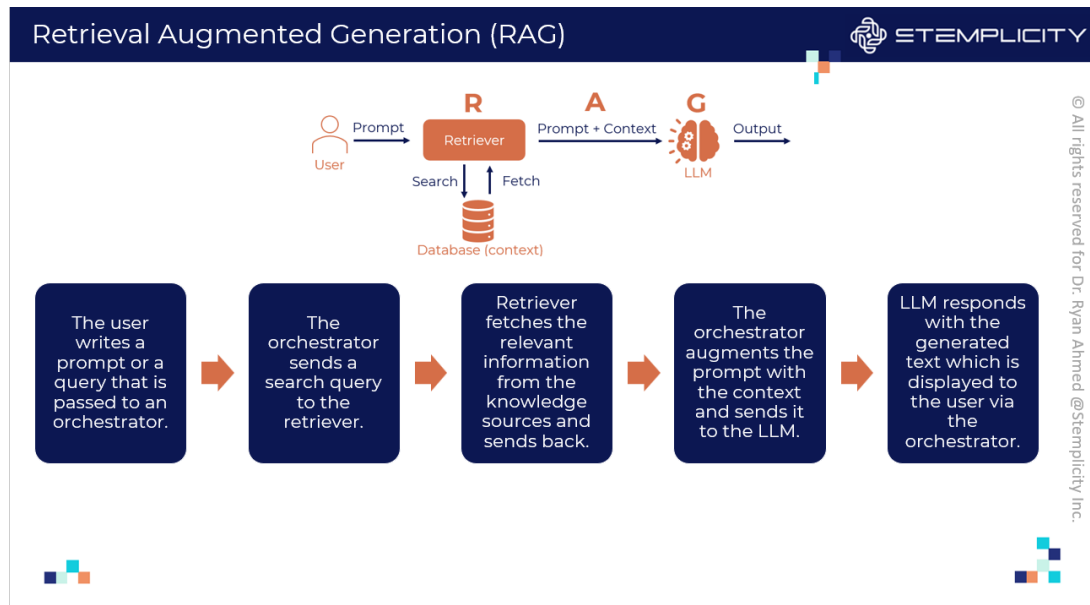


**R** Lookup the external source to retrieve the relevant information

**A** Add the retrieved information to the user prompt

**G** Use LLM to generate a response to the user prompt with the context

© All rights reserved for Dr. Ryan Ahmed @Stemplicity Inc.



## Fine-tuning Vs. RAG

Aspect	Fine-Tuning	Retrieval-Augmented Generation (RAG)
<b>What It Is</b>	Training the base model (e.g., GPT) on a specific dataset to adapt it to a particular use case.	Combines GPT's language generation with a retrieval mechanism that fetches external information.
<b>How It Works</b>	Provide a labeled dataset tailored to your task. Retrain the model using this data, updating its weights.	A document or database is indexed using a vector database. When queried, retrieves the most relevant chunks using similarity search. GPT generates responses by combining retrieved information with general knowledge.
<b>Use Case</b>	Useful for well-structured, high-quality datasets and niche understanding. Requires compute resources and training expertise.	Ideal for applications requiring up-to-date or domain-specific knowledge without retraining.
<b>Advantages</b>	Deeply integrates the knowledge from the dataset.	Does not require retraining the model; integrates external knowledge. Easier to maintain; data can be updated without modifying the model. Cost-effective compared to fine-tuning.
<b>Drawbacks</b>	Expensive and time-consuming. Locks the model into a specific knowledge set, limiting flexibility for dynamic or broad queries.	Relies on the quality of the retrieval system and indexed data. Retrieval may fail if documents are poorly indexed or irrelevant data is fetched.

© All rights reserved for Dr. Ryan Ahmed @Stemplicity Inc.

# TASK 3: LANGCHAIN 101

## What is LangChain?



- LangChain is a framework designed to help developers build applications using large language models more effectively.
- Instead of just sending one prompt and getting one reply, LangChain lets you chain together different components (like prompts, memory, tools, retrieval from documents, etc.) to build more complex, smarter apps.

© All rights reserved for Dr. Ryan Ahmed @Stemplicity Inc.



## LangChain Features



### Chains

You can link together multiple steps or calls to a model.  
Example: Ask a question → Search Wikipedia → Summarize → Translate the summary.

### Agents

LLMs that decide which tools to use and when.  
Think of them like smart assistants that can use calculators, databases, or search engines when needed.

### Tools/Plugins

Connect your LLM to Google search, Python code execution, file reading, etc.  
Example: "What's 347 \* 65?" → LLM uses a calculator tool to answer accurately.

### Memory

Keeps track of the conversation history or user preferences.  
Makes interactions more contextual and human-like.

### Retrieval-Augmented Generation (RAG)

Connect LLM to your documents (databases, websites).  
• It retrieves relevant info from them to answer questions accurately.

© All rights reserved for Dr. Ryan Ahmed @Stemplicity Inc.

# TASK 4. SETUP, GATHER RAG TOOLS, & LOAD THE DATA

```
In [1]: # We start by installing the libraries we need and setting up our 0
# This cell installs the necessary libraries. Please run it once

# VERY IMPORTANT: Microsoft Visual C++ 14.0 or greater is required
# https://visualstudio.microsoft.com/visual-cpp-build-tools/

# The '-q' flag makes the installation less verbose
print("Installing necessary libraries...")
!pip install -q langchain langchain-openai openai chromadb gradio p
print("Libraries installed successfully!")
```

Installing necessary libraries...  
Libraries installed successfully!

```
In [2]: # Let's install and import OpenAI Package
!pip install --upgrade openai
from openai import OpenAI

# Let's import os, which stands for "Operating System"
import os

# This will be used to load the API key from the .env file
from dotenv import load_dotenv
load_dotenv()

# Get the OpenAI API keys from environment variables
openai_api_key = os.getenv("OPENAI_API_KEY")

# Let's configure the OpenAI Client using our key
openai_client = OpenAI(api_key=openai_api_key)
print("OpenAI client successfully configured.")

# Let's view the first few characters in the key
print(openai_api_key[:15])
```

Requirement already satisfied: openai in c:\users\ryana\anaconda3\lib\site-packages (1.76.0)  
 Requirement already satisfied: anyio<5,>=3.5.0 in c:\users\ryana\anaconda3\lib\site-packages (from openai) (4.9.0)  
 Requirement already satisfied: distro<2,>=1.7.0 in c:\users\ryana\anaconda3\lib\site-packages (from openai) (1.9.0)  
 Requirement already satisfied: httpx<1,>=0.23.0 in c:\users\ryana\anaconda3\lib\site-packages (from openai) (0.28.1)  
 Requirement already satisfied: jiter<1,>=0.4.0 in c:\users\ryana\anaconda3\lib\site-packages (from openai) (0.9.0)  
 Requirement already satisfied: pydantic<3,>=1.9.0 in c:\users\ryana\anaconda3\lib\site-packages (from openai) (2.8.2)  
 Requirement already satisfied: sniffio in c:\users\ryana\anaconda3\lib\site-packages (from openai) (1.3.0)  
 Requirement already satisfied: tqdm>4 in c:\users\ryana\anaconda3\lib\site-packages (from openai) (4.66.5)  
 Requirement already satisfied: typing-extensions<5,>=4.11 in c:\users\ryana\anaconda3\lib\site-packages (from openai) (4.11.0)  
 Requirement already satisfied: idna>=2.8 in c:\users\ryana\anaconda3\lib\site-packages (from anyio<5,>=3.5.0->openai) (3.7)  
 Requirement already satisfied: certifi in c:\users\ryana\anaconda3\lib\site-packages (from httpx<1,>=0.23.0->openai) (2025.1.31)  
 Requirement already satisfied: httpcore==1.\* in c:\users\ryana\anaconda3\lib\site-packages (from httpx<1,>=0.23.0->openai) (1.0.2)  
 Requirement already satisfied: h11<0.15,>=0.13 in c:\users\ryana\anaconda3\lib\site-packages (from httpcore==1.\*->httpx<1,>=0.23.0->openai) (0.14.0)  
 Requirement already satisfied: annotated-types>=0.4.0 in c:\users\ryana\anaconda3\lib\site-packages (from pydantic<3,>=1.9.0->openai) (0.6.0)  
 Requirement already satisfied: pydantic-core==2.20.1 in c:\users\ryana\anaconda3\lib\site-packages (from pydantic<3,>=1.9.0->openai) (2.20.1)  
 Requirement already satisfied: colorama in c:\users\ryana\anaconda3\lib\site-packages (from tqdm>4->openai) (0.4.6)  
 OpenAI client successfully configured.  
 sk-proj-H3dZxa9

```
In [3]: # Let's import Langchain components
from langchain_openai import OpenAIEmbeddings, OpenAI
from langchain.vectorstores import Chroma
from langchain.text_splitter import RecursiveCharacterTextSplitter
from langchain.document_loaders import TextLoader
from langchain.chains import RetrievalQAWithSourcesChain
```

```
In [4]: # Define the path to your data file
# Ensure 'eleven_madison_park_data.txt' is in the same folder as the
DATA_FILE_PATH = "eleven_madison_park_data.txt"
print(f>Data file path set to: {DATA_FILE_PATH}")
```

Data file path set to: eleven\_madison\_park\_data.txt

```
In [5]: # Let's load Eleven Madison Park Restaurant data, which has been saved
# The data is saved in "eleven_madison_park_data.txt", Langchain's
print(f>Attempting to load data from: {DATA_FILE_PATH}")
```

```
# Initialize the TextLoader with the file path and specify UTF-8 en
# Encoding helps handle various characters correctly
loader = TextLoader(DATA_FILE_PATH, encoding = "utf-8")

# Load the document(s) using TextLoader from LangChain, which loads
raw_documents = loader.load()
print(f"Successfully loaded {len(raw_documents)} document(s).")
```

Attempting to load data from: eleven\_madison\_park\_data.txt

Successfully loaded 1 document(s).

In [6]: 

```
# Let's display a few characters of the loaded content to perform a
print(raw_documents[0].page_content[:500] + "...")
```

Source: <https://www.elevenmadisonpark.com/>

Title: Eleven Madison Park

Content:

Book on Resy

---END OF SOURCE---

Source: <https://www.elevenmadisonpark.com/careers>

Title: Careers – Eleven Madison Park

Content:

Join Our Team Eleven Madison Park ▼ All Businesses Eleven Madison Pa  
rk Clemente Bar Daniel Humm Hospitality Filter Categories Culinary P  
astray Wine & Beverage Dining Room Office & Admin Other Job Types Ful  
l Time Part Time Compensation Salary Hourly Apply filters OPEN OPPOR  
TUNITIES Staff Acco...

#### PRACTICE OPPORTUNITY:

- Display the last 750 characters in the loaded document
- Perform a sanity check by comparing the printed characters to `eleven_madison_park_data.txt` file
- Print the email and the phone number of the restaurant

In [ ]:

## TASK 5. SPLITTING DOCUMENTS (CHUNKING) WITH LANGCHAIN TEXT SPLITTER

Large documents are hard for AI models to process efficiently and make it difficult to find specific answers. We need to split the loaded document into smaller, manageable "chunks". We'll use Langchain's

`RecursiveCharacterTextSplitter`.

- **Why chunk?** Smaller pieces are easier to embed, store, and retrieve accurately.
- **chunk\_size** : Max characters per chunk.

- **chunk\_overlap** : Characters shared between consecutive chunks (helps maintain context).

```
In [11]: # Let's split the document into chunks
print("\nSplitting the loaded document into smaller chunks...")

# Let's initialize the splitter, which tries to split the document
# sentences (.), and spaces (' ').
text_splitter = RecursiveCharacterTextSplitter(chunk_size = 1000,
                                              chunk_overlap = 150,

# Split the raw document(s) into smaller Document objects (chunks)
documents = text_splitter.split_documents(raw_documents)

# Check if splitting produced any documents
if not documents:
    raise ValueError("Error: Splitting resulted in zero documents.
print(f"Document split into {len(documents)} chunks.")
```

Splitting the loaded document into smaller chunks...  
Document split into 38 chunks.

```
In [12]: # Let's display the Python list containing document chunks
documents
```

```
Out[12]: [Document(metadata={'source': 'eleven_madison_park_data.txt'}, page_content='Source: https://www.elevenmadisonpark.com/\nTitle: Eleven Madison Park\nContent:\nBook on Resy\n---END OF SOURCE---'),
  Document(metadata={'source': 'eleven_madison_park_data.txt'}, page_content='Source: https://www.elevenmadisonpark.com/careers\nTitle: Careers – Eleven Madison Park\nContent:'),
  Document(metadata={'source': 'eleven_madison_park_data.txt'}, page_content='Join Our Team Eleven Madison Park ▾ All Businesses Eleven Madison Park Clemente Bar Daniel Humm Hospitality Filter Categories Culinary Pastry Wine & Beverage Dining Room Office & Admin Other Job Types Full Time Part Time Compensation Salary Hourly Apply filters OPEN OPPORTUNITIES Staff Accountant – Part Time Eleven Madison Park Part Time • Hourly ($20 – $25) Host/Reservationist Eleven Madison Park Full Time • Hourly ($24) Sous Chef Eleven Madison Park Full Time • Salary ($72K – $75K) Pastry Cook Eleven Madison Park Full Time • Hourly ($18 – $20) Kitchen Server Eleven Madison Park Full Time • Hourly ($16) plus tips Dining Room Manager Eleven Madison Park Full Time • Salary ($72K – $75K) Porter Manager Eleven Madison Park Full Time • Salary ($70K – $75K) Senior Sous Chef Eleven Madison Park Full Time • Salary ($85K – $95K) Maitre D Eleven Madison Park Full Time • Hourly ($16) plus tips Even if you don't see the opportunity you're looking for, we would still love to hear from you. There"),
  Document(metadata={'source': 'eleven_madison_park_data.txt'}, page_content='Madison Park Full Time • Hourly ($16) plus tips Even if you don't see the opportunity you're looking for, we would still love to hear from you. There may be a place for you on our team as we grow. × Where Do You Want To Work? Eleven Madison Park Clemente Bar Daniel Humm Hospitality I Want to Work Here'),
  Document(metadata={'source': 'eleven_madison_park_data.txt'}, page_content='...')]
```



e\_content='---END OF SOURCE---'),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page\_content='Source: <https://www.elevenmadisonpark.com/giftcards>\nTitle: Gift Cards – Eleven Madison Park\nContent:\nGift Cards Purchase Gift Cards Here Please note that all gift card sales are final and nonrefundable.\n---END OF SOURCE---'),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page\_content='Source: <https://www.elevenmadisonpark.com/press-and-accolades>\nTitle: Press and Accolades – Eleven Madison Park\nContent:\nAccolades World’s 50 Best Best of the Best The New York Times Four Stars Michelin Guide Three Stars Wine Spectator Grand Award The World of Fine Wine Best Overall Wine List James Beard Foundation Outstanding Chef, Outstanding Service, Outstanding Pastry Chef, Outstanding Restaurant, Best Chef: NYC, Outstanding Wine Service, Rising Star Chef Recent Press Time: “Is Cooking for the 1% for a Reason” Washington Post: “The Joy of Plant-Based Eating” Interview: “Daniel Humm Is Begging You to Eat More Plants” Financial Times: “Episcurean escapes: Alain Ducasse and Daniel Humm’s transatlantic adventure” The Strategist: “What Chef Daniel Humm Can’t Live Without”\n---END OF SOURCE---'),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page\_content='Source: <https://www.elevenmadisonpark.com/accessibility-statement>\nTitle: Accessibility Statement – Eleven Madison Park\nContent:\nAccessibility Statement Eleven Madison Park is committed to providing digital accessibility for everyone. Our ongoing accessibility effort works towards conforming to the Web Content Accessibility Guidelines (WCAG) version 2.1, level AA criteria. These guidelines not only help make web content accessible to users with sensory, cognitive and mobility disabilities, but ultimately to all users, regardless of ability. Our ongoing accessibility efforts work toward making ElevenMadisonPark.com as accessible as possible. Eleven Madison Park welcomes comments on how to improve the site’s accessibility for users with disabilities. Please email: [info@elevenmadisonpark.com](mailto:info@elevenmadisonpark.com) .\n---END OF SOURCE---\n\nSource: <https://www.elevenmadisonpark.com/cart>\nTitle: Eleven Madison Park\nContent:\nShopping Cart You have nothing in your shopping cart. Continue Shopping\n---END OF SOURCE---'),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page\_content='Source: <https://www.elevenmadisonpark.com/faq>\nTitle: FAQs – Eleven Madison Park\nContent:'),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page\_content='FAQs We are located at 11 Madison Avenue, on the northeast corner of East 24th and Madison Avenue, directly across the street from Madison Square Park. We offer three menus, all 100% plant-based: Full Tasting Menu : An eight- to nine-course experience priced at \$365 per guest. This menu typically lasts about two to three hours and features a mix of plated and communal dishes. 5-Course Menu : Priced at \$285 per guest, this menu highlights selections from the Full Tasting Menu and lasts approximately two hours. Bar Tasting Menu : Available in our lounge for \$225 per guest, this menu includes four to five courses and is designed to last around two hours. Note : These durations are estimates based on tables of two. Larger parties may require additional time. We open up reservations on the first of every month for the following month—for example, on October 1st, all of November will be made available. If you find that the reservation you were hoping for is booked, we do

hold a waitlist on'),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page\_content='on October 1st, all of November will be made available. If you find that the reservation you were hoping for is booked, we do hold a waitlist on Resy and encourage you to add your name to it, and if something becomes available, we will be in touch. All sales are final and non-refundable. We do not offer cancellation or rescheduling options at this time. Eleven Madison Park does not include gratuity for the dining experience. Guests are welcome to leave a desired gratuity at the conclusion of their experience at their discretion. For any questions on our gratuity policies, please contact our guest relations team at [info@elevenmadisonpark.com](mailto:info@elevenmadisonpark.com). Not a problem! We will be verifying all food allergies, aversions, and dietary restrictions prior to you joining us, but also at the table when you arrive for your meal. Yes. We offer wine pairings, starting at \$125 per guest, as well as a full wine list that our wine team can help you select from and may also be viewed on our website. You are'),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page\_content='starting at \$125 per guest, as well as a full wine list that our wine team can help you select from and may also be viewed on our website. You are also welcome to bring your own special bottle of wine for your meal for a \$75 per 750ml bottle corkage (4 bottle maximum) fee. We do not have a dress code. Many of our guests dress up for the occasion but wear whatever will make you most comfortable. You can reach our reservations team at any time at [info@elevenmadisonpark.com](mailto:info@elevenmadisonpark.com). We can accommodate up to seven guests at a table in the main dining room. If you are interested in booking a table for a group larger than this, please contact our Private Dining and Special Events team at [events@elevenmadisonpark.com](mailto:events@elevenmadisonpark.com). Yes, you can. Reservations for our Bar Tasting menu are available at our lounge tables and are \$225 per guest; you may also order a number of items a la carte. Seats at the bar counter are on a first-come, first-serve basis. You may order our Bar Tasting menu in addition to'),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page\_content='a number of items a la carte. Seats at the bar counter are on a first-come, first-serve basis. You may order our Bar Tasting menu in addition to cocktails, light snacks, and our a la carte menu. Unfortunately, Resy does not have the capability to accept gift cards as a form of payment at this time. Please secure your reservation using an alternate payment method, and our accounting team will be happy to help redeem your Eleven Madison Park gift card and credit the form of payment you used to make the booking. Contact Marcia Regen ( [mregen@hummhospitality.com](mailto:mregen@hummhospitality.com) ) for help. You can always also bring Eleven Madison Park gift cards into the restaurant when you dine to be used toward any beverage purchased on site. To purchase a Braille gift card, please contact Marcia Regen ( [mregen@hummhospitality.com](mailto:mregen@hummhospitality.com) ).'),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page\_content='---END OF SOURCE---'),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page\_content='Source: <https://www.elevenmadisonpark.com/ourrestaurant>\nTitle: About - Eleven Madison Park\nContent:'),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page\_content='Welcome to Eleven Madison Park Eleven Madison Park is a

fine dining restaurant in the heart of New York City. Overlooking Madison Square Park—one of Manhattan’s most beautiful green spaces—we sit at the base of a historic Art Deco building on the corner of East 24th Street and Madison Avenue. Since opening in 1998, we underwent a full-scale renovation and redesign in the summer of 2017. Chef Daniel Humm has owned the restaurant since 2011, during which time we have evolved considerably in both cuisine and experience. In 2021, we transitioned to a fully plant-based menu, using no animal products. That same year, we partnered with Magic Farms, which grows produce exclusively for our seasonal menus. Guests can enjoy a full tasting menu, a five-course menu, or a bar menu. The bar also offers à la carte snacks, as well as wine and cocktails. Reservations are available via Resy, and bar seating is open for both walk-ins and reservations. Hours: Monday to Wednesday: 5:30 pm to 10 pm),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page\_content='Reservations are available via Resy, and bar seating is open for both walk-ins and reservations. Hours: Monday to Wednesday: 5:30 pm to 10 pm Thursday to Friday: 5 pm to 11 pm Saturday: 12 pm to 2 pm, 5 pm to 11 pm Sunday: 12 pm to 2 pm, 5 pm to 11 pm View Wine List View Cocktail List Due to the hyper-seasonal nature of our menu, all courses are subject to change. Therefore, we do not share our food menus online. Book on Resy'),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page\_content='---END OF SOURCE---'),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page\_content='Source: <https://www.elevenmadisonpark.com/magic-farms>\nTitle: Magic Farms – Eleven Madison Park\nContent:'),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page\_content='About Magic Farms As part of our transition to a plant-based menu, it was important for us to build a deeper connection to the ingredients we use in our kitchen. In 2021, we partnered with Chef Daniel Humm’s friend, Maciek Kobielski, to create Magic Farms on his family’s land in Hoosick Falls, New York. What began as a passion project for Maciek has become a full-time career. The farm now spans four acres of fields and seven high tunnels, growing produce exclusively for our seasonal menus. At the restaurant, this partnership has profoundly influenced our craft, fostering an ongoing dialogue between chef and farmer that shapes every menu. Our team collaborates with Maciek to plan menus around his harvest cycles and crop rotations, mapping out seasons in advance. Maciek grows each vegetable to our specifications using pesticide-free, regenerative techniques, with harvests peaking in the warmer months. This ensures access to the freshest, most flavorful produce year-round and allows each'),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page\_content='techniques, with harvests peaking in the warmer months. This ensures access to the freshest, most flavorful produce year-round and allows each menu to reflect the best of the Northeast’s seasons. Book on Resy Maciek and Chef Humm Cherry Tomato Sunset over Magic Farms Delivery to the Restaurant Squash Blossoms Chefs Mattia and Josh visiting the farm. Badger Flame Beets Garlic Bulbs Sunflower Harvest A Tunnel in Spring Eggplants Greens Growing Radicchio Snow Day Maciek on His Tractor Radish Harvest Lettuces Growing Item 1 of 17'),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page

e\_content='---END OF SOURCE---'),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page\_content='Source: <https://www.elevenmadisonpark.com/team>\nTitle: Team – Eleven Madison Park\nContent:'),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page\_content='Daniel Humm is a chef, author, speaker, and owner of Daniel Humm Hospitality , the New York-based hospitality group behind the highly acclaimed Eleven Madison Park, Clemente Bar , and direct-to-consumer lifestyle brand Eleven Madison Home . A native of Switzerland, Chef Humm earned his first Michelin star at the age of 24. He is consistently listed as one of the world's best chefs, with both he and Eleven Madison Park receiving numerous accolades: four stars from The New York Times, seven James Beard Foundation Awards (including Outstanding Chef and Outstanding Restaurant in America), a number one spot on the world's 50 Best Restaurants list, and three Michelin stars for over 12 years in a row, since 2012. At the height of the COVID-19 pandemic, Chef Humm transformed Eleven Madison Park's Michelin-starred restaurant and its back-of-house into a commissary kitchen in partnership with Rethink Food , a not-for-profit organization co-founded by Chef Humm. He and his team prepared over'),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page\_content='into a commissary kitchen in partnership with Rethink Food , a not-for-profit organization co-founded by Chef Humm. He and his team prepared over 1,000,000 meals over the course of the pandemic for frontline workers and underserved communities and distributed them to churches, shelters, and food banks. In 2021, he reopened Eleven Madison Park with a completely plant-based menu and became the first and only plant-based restaurant in Michelin Guide history to receive a three-star rating in October 2022. Gwendal Poullennec, the International Director of the Michelin Guides, has called Chef Humm and Eleven Madison Park a North Star that young chefs can fix their eyes upon as they navigate their gastronomic journey. Chef Humm uses his global platform to advocate for equitable, sustainable food systems. He wants to inspire people both within and beyond the fine-dining world and build awareness around the transformative power of food. As a leading figure in the culinary realm, he ignited a'),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page\_content='and beyond the fine-dining world and build awareness around the transformative power of food. As a leading figure in the culinary realm, he ignited a conversation about challenging the definition of luxury in the food world and how it can be more purpose-driven at the United Nations Climate Change Conference in Glasgow, Scotland, in 2021. He discussed the impact of our food systems on the environment at the 2023's Global Citizen Summit, joining activists and world leaders such as Canadian Prime Minister Justin Trudeau and French President Macron in setting a global agenda for action on the most urgent issues facing humanity and the planet. In September 2024, he was appointed UNESCO Goodwill Ambassador for Food Education by Audrey Azoulay, UNESCO's Director-General. Chef Humm is the author of Eleven Madison Park: The Cookbook , I Love New York: Ingredients and Recipes , The NoMad Cookbook, Eleven Madison Park: The Next Chapter , and Eat More Plants . His latest book, titled Eleven'),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page

e\_content='New York: Ingredients and Recipes , The NoMad Cookbook, Eleven Madison Park: The Next Chapter , and Eat More Plants . His latest book, titled Eleven Madison Park: The Plant-Based Chapter , was released in November 2024. Dominique Roy's passion for cooking began at a young age, baking bread alongside his mother in his hometown of Gatineau, Canada. After attending culinary school at 17 and pastry school at 19, Dom worked for renowned chefs in Canada and France, including Georges Blanc and Jerome Ferrer. He participated in many different cooking competitions, including representing Team Canada in the Luxembourg Culinary World Cup in 2014. Dom always knew he wanted to work for one of the best restaurants in the world, and after staging at Maaemo in Oslo and Mirazur in France, he arrived in New York City from Montréal. He started with Eleven Madison Park in 2015, holding various positions, including Sous Chef and Culinary Research & Development. He also led the opening team of Davies &'),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page\_content='Madison Park in 2015, holding various positions, including Sous Chef and Culinary Research & Development. He also led the opening team of Davies & Brook in London in 2019. During the COVID-19 pandemic in 2020, Dom spearheaded the Eleven Madison Park and Rethink Food partnership, leading the program that provided hundreds of thousands of meals to both food-insecure residents of New York City as well as frontline workers. In 2021, he assumed the title of Chef de Cuisine, leading the reopening and building the vision for EMP's future, and in 2023 he became a partner at Daniel Humm Hospitality. Laura Cronin grew up having a love for food at an early age, working in bakeries in her hometown of Berkeley Heights, New Jersey. After graduating culinary school at Johnson & Wales University in Rhode Island, she moved across the country to pursue a career on the baking team at the San Francisco Baking Institute under the tutelage of Michel Suas. A year later, Laura relocated to Paris and enrolled'),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page\_content='on the baking team at the San Francisco Baking Institute under the tutelage of Michel Suas. A year later, Laura relocated to Paris and enrolled at the Ecole Gregoire-Ferrandi, which led to an apprenticeship with Chef Angelo Musa. In 2011, Laura moved back to the States and worked in several restaurants including Zero Zero and Perbacco, where she was named San Francisco Chronicle's Rising Star Chef and Zagat's 30 under 30 list. In 2017, Laura moved to New York City to join the Eleven Madison Park team as a Pastry Sous Chef leading the team through the restaurant's reopening and working closely with Executive Pastry Chef Mark Welker, she assumed the Pastry Chef role in 2019. Originally from New Jersey, Daniel Di Stefano is the Executive Culinary Director for Daniel Humm Hospitality. Danny joined the team in 2009, starting as a commis and working his way up to sous chef in 2011. He has been instrumental in launching brands such as Made Nice and Eleven Madison Home. Following the pandemic,')

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page\_content='his way up to sous chef in 2011. He has been instrumental in launching brands such as Made Nice and Eleven Madison Home. Following the pandemic, Danny returned as a Managing Partner to spearhead our at-home program and strategic partnerships, and in 2024, Danny assumed the role of Executive Culinary Director. In this

role, he leads culinary operations and works closely with Chef Humm and the rest of the culinary development team on menus, partnerships, events, collaborations, and new projects. Josh Harnden was born in Seattle, Washington, where he was raised with a plant-based diet, which fueled his passion for food and the culinary arts from a young age. Immersed in the restaurant industry for over two decades, he worked his way through the Seattle culinary scene—including at the renowned Canlis—before moving to New York in 2011 to help open and manage Chef Daniel Humm's restaurant at NoMad Hotel. After helping NoMad earn its first Michelin star, Josh was asked to join the team at'),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page\_content='open and manage Chef Daniel Humm's restaurant at NoMad Hotel. After helping NoMad earn its first Michelin star, Josh was asked to join the team at Eleven Madison Park, where he has held a number of different management positions, including leading research and development. In this role, he helped to launch a range of pop-ups and openings for Daniel Humm Hospitality, as well as developing recipes for the NoMad and Eleven Madison Park cookbooks, including this one. He is now a partner and the Creative Culinary Director for Daniel Humm Hospitality. Andrew Chandler was essentially born into restaurants, spending much of his childhood watching his mother command the dining rooms of some of the best restaurants in his home state of Wisconsin. He followed in her footsteps, taking on his first jobs bussing tables and washing dishes as a teenager. After high school, he graduated from University of Wisconsin-Platteville with a bachelor's degree in Psychology. He went on to spend several years'),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page\_content='After high school, he graduated from University of Wisconsin-Platteville with a bachelor's degree in Psychology. He went on to spend several years working with children with special needs, while simultaneously working full time in some of the best restaurants in Milwaukee, including Riverside and Braise. Eventually it became clear that his true calling was working in the dining room, and in 2012 he moved across the country to start as a Kitchen Server at Eleven Madison Park. Andrew has worked through nearly every position in the restaurant, making the transition to management in 2016. Within Make it Nice he has led the team through the pop-up EMP Summer House, the subsequent reopening of Eleven Madison Park, and the opening of Davies and Brook in London. For our General Manager, Gabriel Di Bella, an understanding of delicious food and wine runs in the family. The son of an Italian chef, he grew up helping out in his parents' restaurant in France. As a young adult, he studied at the'),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page\_content='wine runs in the family. The son of an Italian chef, he grew up helping out in his parents' restaurant in France. As a young adult, he studied at the wine school of Tain-l'Hermitage in the heart of the Rhône Valley, before joining the wine team at Alain Ducasse in Monaco. He later spent time at Marcus at The Berkeley and the Birley Clubs in London. Gabriel came on board at Daniel Humm Hospitality as the opening Wine Director for Davies & Brook in 2019, and in 2021, he joined us at Eleven Madison Park as our Wine Director. Today, he leads the team as our General Manager. Our Wine Director, Adam Waddell, was born and raised on the West Coast, where

re he started violin studies at the age of four. After coming to New York in 2010 to continue studies in Classical Violin Performance, he soon developed a new-found passion for wine and hospitality. Since then, he has worked at many well-known New York City restaurants, such as Jean-Georges, Ai Fiori, and Gabriel Kreuther. Adam was the first'),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page\_content='Since then, he has worked at many well-known New York City restaurants, such as Jean-Georges, Ai Fiori, and Gabriel Kreuther. Adam was the first Head Sommelier at SAGA by Jamal James Kent and joined Eleven Madison Park in 2023 as Associate Wine Director. In 2025, he was named Wine Director and now leads the program with dedication and expertise. Sebastian Tollius has been perfecting his craft in the hospitality industry for almost 15 years. Before joining us at Eleven Madison Park in 2019, he worked in some of the best bars and hotels around the world, including in Switzerland, Thailand, and Spain. Bringing a range of innovative concepts and a culinary approach, Sebastian expertly led our Beverage team through our 2021 re-opening and plant-based transition, and he has transformed our cocktail program through new plant-based techniques and ingredients. He and his team now work closely with our chefs in the kitchen to create a brand-new cocktail menu each season that complements and'),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page\_content='and ingredients. He and his team now work closely with our chefs in the kitchen to create a brand-new cocktail menu each season that complements and highlights ingredients from the tasting menu – constantly pushing boundaries when it comes to both technique and flavor. Mattia Rancati, Senior Sous Chef Stefano Casale, Senior Sous Chef Davide Peli, Sous Chef Luis Garcia, Sous Chef Marissa Mazzella, Sous Chef Pascal Rauwolf, Executive Pastry Sous Chef Orianna Mendez, Junior Pastry Sous Chef Pooja Harsora, Junior Sous Chef Youngbo Lee, Junior Sous Chef Alexander Mies, Purchasing Manager James Gale, Service Director Harry Basley, Dining Room Manager Eliazar Cervantes, Restaurant Operations Manager Marina Dos Santos Malta, Guest Relations Manager Caitlin McBride, Private Dining Room Manager'),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page\_content='---END OF SOURCE---'),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page\_content='Source: <https://www.elevenmadisonpark.com/events>\nTitle: Events – Eleven Madison Park\nContent:\nEvents & Private Dining Our restaurant offers a variety of stunning spaces to suit any event, from intimate gatherings to grand celebrations. Each space has its own unique character, seamlessly blending art, design, and exceptional hospitality. Whether you envision a seated dinner, a cocktail reception, or a corporate meeting, our thoughtfully designed venues can be tailored to your needs. For more information about hosting an event with us, please review our Private Events Deck, or contact the Private Dining and Special Events Department using the form linked below. Learn More Inquire\n---END OF SOURCE---'),

Document(metadata={'source': 'eleven\_madison\_park\_data.txt'}, page\_content='Source: <https://www.elevenmadisonpark.com/contact>\nTitle: Contact – Eleven Madison Park\nContent:\nContact Us Visit Resy Note: We open up reservations on the first of every month for the following month – for example, on October 1st, all of November will

l be made available. If you find that the reservation you were hoping for is booked, we do hold a waitlist and encourage you to add your name to it, and if something does become available, we will be in touch. Please contact [careers@elevenmadisonpark.com](mailto:careers@elevenmadisonpark.com) or visit Culinary Agents . To stay up to date about future Eleven Madison Park news and events, please sign up for our Newsletter . Please contact [events@elevenmadisonpark.com](mailto:events@elevenmadisonpark.com) Please contact [press@elevenmadisonpark.com](mailto:press@elevenmadisonpark.com) Thank you for thinking of Eleven Madison Park for your inquiry. At this time, all of our non-profit efforts are focused on our partnership with Rethink Food . Email: [info@elevenmadisonpark.com](mailto:info@elevenmadisonpark.com) Phone: 212.889.0905 Ext. 3\n---END OF SOURCE---']}]

```
In [13]: # Let's display an example chunk and its metadata
print("\n--- Example Chunk (Chunk 2) ---")
print(documents[2].page_content)
print("\n--- Metadata for Chunk 2 ---")
print(documents[2].metadata) # Should show {'source': 'eleven_madis
```

--- Example Chunk (Chunk 2) ---

Join Our Team Eleven Madison Park ▾ All Businesses Eleven Madison Park Clemente Bar Daniel Humm Hospitality Filter Categories Culinary Pastry Wine & Beverage Dining Room Office & Admin Other Job Types Full Time Part Time Compensation Salary Hourly Apply filters OPEN OPPORTUNITIES Staff Accountant – Part Time Eleven Madison Park Part Time • Hourly (\$20 – \$25) Host/Reservationist Eleven Madison Park Full Time • Hourly (\$24) Sous Chef Eleven Madison Park Full Time • Salary (\$72K – \$75K) Pastry Cook Eleven Madison Park Full Time • Hourly (\$18 – \$20) Kitchen Server Eleven Madison Park Full Time • Hourly (\$16) plus tips Dining Room Manager Eleven Madison Park Full Time • Salary (\$72K – \$75K) Porter Manager Eleven Madison Park Full Time • Salary (\$70K – \$75K) Senior Sous Chef Eleven Madison Park Full Time • Salary (\$85K – \$95K) Maitre D Eleven Madison Park Full Time • Hourly (\$16) plus tips Even if you don't see the opportunity you're looking for, we would still love to hear from you. There

--- Metadata for Chunk 2 ---

```
{'source': 'eleven_madison_park_data.txt'}
```

### PRACTICE OPPORTUNITY:

- Change `chunk_size` to `500` in the `RecursiveCharacterTextSplitter` and re-run the cell. How many chunks do you get now? Is it more or less than before? Change it back to `1000`
- Change `chunk_overlap` to `0` and re-run the cell. Does the number of chunks change drastically? What problem might setting overlap to `0` cause? Change it back to `150`
- Print the last example chunk and its metadata and re-run the cell. What information is stored in the `metadata` ? Why is the `'source'` important?

In [ ]:



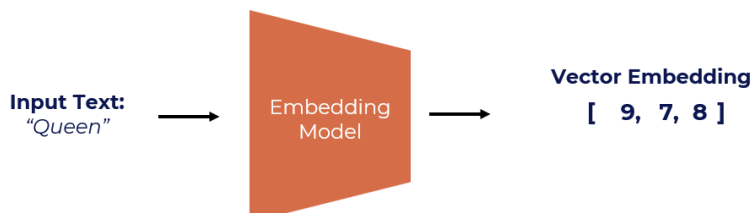
# TASK 6. EMBEDDINGS AND VECTOR STORE CREATION

## What is Embedding?



- In the context of Large Language Models (LLMs), embedding refers to a way of representing words, phrases, sentences, and documents as dense vectors of numbers.
- These vectors capture the semantic meaning of the text, allowing the model to understand and work with language more effectively.

© All rights reserved for Dr. Ryan Ahmed @Stemplicity Inc.

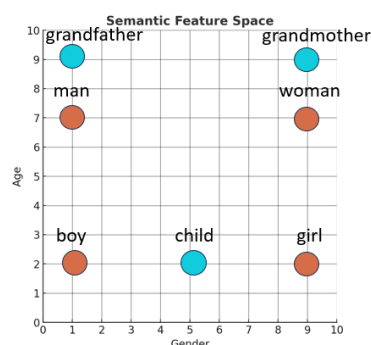


<https://www.cs.cmu.edu/~dst/WordEmbeddingDemo/tutorial.html>

## Embedding Example



- Assume that we want to represent the words "man", "woman", "boy", and "girl" on a **semantic feature space**.
- On the X-axis, we have "gender", and on the Y-axis, we have "Age". These are called **semantic features**.
- Note that two words refer to males, two to females, two to adults, and two to children.



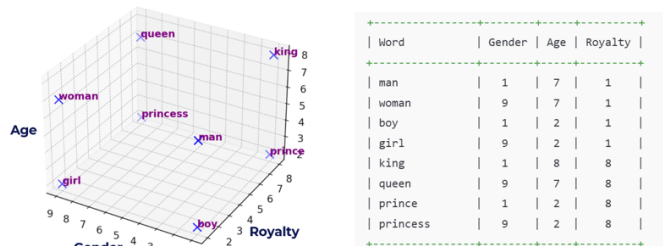
© All rights reserved for Dr. Ryan Ahmed @Stemplicity Inc.

Word	Coordinates	Gender	Age
man	[ 1, 7 ]		
woman	[ 9, 7 ]		
boy	[ 1, 2 ]		
girl	[ 9, 2 ]		
child	[ 5, 2 ]		
grandmother	[ 9, 9 ]		
grandfather	[ 1, 9 ]		

## Let's View Embeddings in 3D Space!



- Let's now look at the words "king," "queen," "prince," and "princess."
- While these words share similar gender & age attributes with "man," "woman," "boy," and "girl," they have different meanings.
- To distinguish "man" from "king," "woman" from "queen," we need to introduce a new semantic feature that sets them apart. We'll call this feature **"Royalty"**. Let's view them in a 3D space.



© All rights reserved for Dr. Ryan Ahmed @Stemplicity Inc.

Now, we convert our text chunks into **embeddings** (numerical vectors) using OpenAI. Similar text chunks will have similar vectors. We then store these vectors in a **vector store** (ChromaDB) for fast searching.

- **Embeddings:** Text -> Numbers (Vectors) representing meaning.
- **Vector Store:** Database optimized for searching these vectors.

Tensorflow Ebeddings projector (it's fun!): <https://projector.tensorflow.org/>

```
In [22]: # Let's initialize our embeddings model. Note that we will use OpenAI
print("Initializing OpenAI Embeddings model...")

# Create an instance of the OpenAI Embeddings model
# Langchain handles using the API key we loaded earlier
embeddings = OpenAIEmbeddings(openai_api_key = openai_api_key)

print("OpenAI Embeddings model initialized.")

# Let's Create ChromaDB Vector Store
print("\nCreating ChromaDB vector store and embedding documents...")

# Now the chunks from 'documents' are being converted to a vector u
# The vectors are then stored as a vector in ChromaDB
# You could add `persist_directory='./my_chroma_db'` to save it to
# You will need to specify: (1) The list of chunked Document object
vector_store = Chroma.from_documents(documents = documents, embeddi

# Verify the number of items in the store
vector_count = vector_store._collection.count()
print(f"ChromaDB vector store created with {vector_count} items.")

if vector_count == 0:
    raise ValueError("Vector store creation resulted in 0 items. Ch
```

```
Initializing OpenAI Embeddings model...
OpenAI Embeddings model initialized.
```

```
Creating ChromaDB vector store and embedding documents...
ChromaDB vector store created with 38 items.
```

```
In [23]: # Let's retrieve the first chunk of stored data from the vector store
stored_data = vector_store._collection.get(include=["embeddings", "documents"])

# Display the results
print("First chunk text:\n", stored_data['documents'][0])
print("\nEmbedding vector:\n", stored_data['embeddings'][0])
print(f"\nFull embedding has {len(stored_data['embeddings'][0])} dimensions.")
```

First chunk text:

```
Source: https://www.elevenmadisonpark.com/
Title: Eleven Madison Park
Content:
Book on Resy
---END OF SOURCE---
```

Embedding vector:

```
[ 0.02330522 -0.01571015 -0.00706136 ... -0.02464633 -0.01022939
 -0.06158162]
```

Full embedding has 1536 dimensions.

#### PRACTICE OPPORTUNITY:

- Using Tensorflow Embeddings Projector, explore the nearest points to "Italy"
- Choose 2 additional different words that you like
- Tensorflow Ebeddings projector: <https://projector.tensorflow.org/>

In [ ]:

## TASK 7. TESTING THE RETRIEVAL

Before building the full Q&A chain, let's test if our vector store can find relevant chunks based on a sample question. We'll use the `similarity_search` method.

```
In [29]: # Let's perform a similarity search in our vector store
print("\n--- Testing Similarity Search in Vector Store ---")
test_query = "What different menus are offered?"
print(f"Searching for documents similar to: '{test_query}'")

# Perform a similarity search. 'k=2' retrieves the top 2 most similar documents.
try:
    similar_docs = vector_store.similarity_search(test_query, k = 2)
    print(f"\nFound {len(similar_docs)} similar documents:")
```

```

# Display snippets of the retrieved documents and their sources
for i, doc in enumerate(similar_docs):
    print(f"\n--- Document {i+1} ---")
    # Displaying the first 700 chars for brevity
    content_snippet = doc.page_content[:700].strip() + "..."
    source = doc.metadata.get("source", "Unknown Source") # Get source
    print(f"Content Snippet: {content_snippet}")
    print(f"Source: {source}")

except Exception as e:
    print(f"An error occurred during similarity search: {e}")

```

--- Testing Similarity Search in Vector Store ---

Searching for documents similar to: 'What different menus are offered?'

Found 2 similar documents:

--- Document 1 ---

Content Snippet: FAQs We are located at 11 Madison Avenue, on the northeast corner of East 24th and Madison Avenue, directly across the street from Madison Square Park. We offer three menus, all 100% plant-based: Full Tasting Menu : An eight- to nine-course experience priced at \$365 per guest. This menu typically lasts about two to three hours and features a mix of plated and communal dishes. 5-Course Menu : Priced at \$285 per guest, this menu highlights selections from the Full Tasting Menu and lasts approximately two hours. Bar Tasting Menu : Available in our lounge for \$225 per guest, this menu includes four to five courses and is designed to last around two hours. Note : These durations are estimates based on...

Source: eleven\_madison\_park\_data.txt

--- Document 2 ---

Content Snippet: Reservations are available via Resy , and bar seating is open for both walk-ins and reservations. Hours: Monday to Wednesday: 5:30 pm to 10 pm Thursday to Friday: 5 pm to 11 pm Saturday: 12 pm to 2 pm, 5 pm to 11 pm Sunday: 12 pm to 2 pm, 5 pm to 11 pm View Wine List View Cocktail List Due to the hyper-seasonal nature of our menu, all courses are subject to change. Therefore, we do not share our food menus online. Book on Resy...

Source: eleven\_madison\_park\_data.txt

### PRACTICE OPPORTUNITY:

- Change the `test_query` variable to ask one of the following questions. Re-run the cell for each question. Do the retrieved document snippets seem relevant to your question?
  - "Who is Daniel Humm?"
  - "Is there a dress code?"
  - "Tell me about the partnership with Magic Farms."
  - "How do I make a reservation?"
- Adjust the value of `k` to `k=1` and then to `k=5` . Re-run the cell. How does changing `k` affect the number of documents retrieved?

In [ ]:

In [ ]:

## TASK 8. BUILDING & TESTING THE RAG CHAIN USING LANGCHAIN

Now we assemble the core RAG logic using Langchain's `RetrievalQAWithSourcesChain`. This chain combines:

1. A **Retriever**: Fetches relevant documents from our `vector_store`.
2. An **LLM**: Generates the answer based on the question and retrieved documents (we'll use OpenAI).

This specific chain type automatically handles retrieving documents, formatting them with the question for the LLM, and tracking the sources.

```
In [50]: # --- 1. Define the Retriever ---
# The retriever uses the vector store to fetch documents
# We configure it to retrieve the top 'k' documents
retriever = vector_store.as_retriever(search_kwargs={"k": 3})
print("Retriever configured successfully from vector store.")

# --- 2. Define the Language Model (LLM) from OpenAI---
# Temperature controls the model's creativity; 'temperature=0' aims
# You might need to specify a more powerful model, such as "gpt-3.5
llm = OpenAI(temperature = 1.3, openai_api_key = openai_api_key)
print("OpenAI LLM successfully initialized.")

# --- 3. Create the RetrievalQAWithSourcesChain ---
# This chain type is designed specifically for Q&A with source trac
# chain_type="stuff": Puts all retrieved text directly into the pro
#                               Suitable if the total text fits within the L
#                               Other types like "map_reduce" handle larger
qa_chain = RetrievalQAWithSourcesChain.from_chain_type(llm = llm,
                                                        chain_type =
                                                        retriever =
                                                        return_sourc
                                                        verbose = Tr

print("RetrievalQAWithSourcesChain created")
```

Retriever configured successfully from vector store.  
 OpenAI LLM successfully initialized.  
 RetrievalQAWithSourcesChain created

```
In [52]: # --- Test the Full Chain ---
print("\n--- Testing the Full RAG Chain ---")
chain_test_query = "What kind of food does Eleven Madison Park serv
print(f"Query: {chain_test_query}")
```

```

# Run the query through the chain. Use invoke() for Langchain >= 0.
# The input must be a dictionary, often with the key 'question'.
try:
    result = qa_chain.invoke({"question": chain_test_query})

    # Print the answer and sources from the result dictionary
    print("\n--- Answer ---")
    print(result.get("answer", "No answer generated."))

    print("\n--- Sources ---")
    print(result.get("sources", "No sources identified."))

    # Optionally print snippets from the source documents returned
    if "source_documents" in result:
        print("\n--- Source Document Snippets ---")
        for i, doc in enumerate(result["source_documents"]):
            content_snippet = doc.page_content[:250].strip()
            print(f"Doc {i+1}: {content_snippet}")

except Exception as e:
    print(f"\nAn error occurred while running the chain: {e}")
    # Consider adding more specific error handling if needed

```

--- Testing the Full RAG Chain ---

Query: What kind of food does Eleven Madison Park serve?

> Entering new RetrievalQAWithSourcesChain chain...

> Finished chain.

--- Answer ---

Eleven Madison Park serves a discounted plant-based menu and farm-sourced à la carte snacks, reservations are required.

--- Sources ---

<https://www.elevenmadisonpark.com/faq>

--- Source Document Snippets ---

Doc 1: Welcome to Eleven Madison Park Eleven Madison Park is a fine dining restaurant in the heart of New York City. Overlooking Madison Square Park—one of Manhattan's most beautiful green spaces—we sit at the base of a historic Art Deco building on the cor

Doc 2: Source: <https://www.elevenmadisonpark.com/ourrestaurant>

Title: About – Eleven Madison Park

Content:

Doc 3: Source: <https://www.elevenmadisonpark.com/faq>

Title: FAQs – Eleven Madison Park

Content:

### PRACTICE OPPORTUNITY:

- Look at the `result` dictionary printed by the chain test. What are the key pieces of information it contains?

- Change `temperature=0` to `temperature=1.3` in the `llm = OpenAI(...)` line. Re-run the cell and ask the *same* `chain_test_query`. Does the wording of the answer change slightly? Change it back to `0`.
- Set `verbose=True` and `return_source_documents=True` in the `RetrievalQAWithSourcesChain.from_chain_type(...)` line. Re-run the cell. What extra information do you see printed? Set it back to `False`.

In [ ]:

## TASK 9. CREATING A GRADIO INTERFACE FOR OUR RAG CHAIN

Let's wrap our RAG chain in a user-friendly web interface using Gradio. Users will type a question, click a button, and see the answer along with the sources the AI used.

```
In [59]: # Gradio for UI
import gradio as gr
```

```
In [60]: # --- Define the Function for Gradio ---

# This function takes the user's input, runs the chain, and formats
# Ensure the `qa_chain` variable is accessible in this scope.
def ask_elevenmadison_assistant(user_query):
    """
    Processes the user query using the RAG chain and returns format
    """
    print(f"\nProcessing Gradio query: '{user_query}'")
    if not user_query or user_query.strip() == "":
        print("--> Empty query received.")
        return "Please enter a question.", "" # Handle empty input

    try:
        # Run the query through our RAG chain
        result = qa_chain.invoke({"question": user_query})

        # Extract answer and sources
        answer = result.get("answer", "Sorry, I couldn't find an an
        sources = result.get("sources", "No specific sources identi

        # Basic formatting for sources (especially if it just retur
        if sources == DATA_FILE_PATH:
            sources = f"Retrieved from: {DATA_FILE_PATH}"
        elif isinstance(sources, list): # Handle potential list of
            sources = ", ".join(list(set(sources))) # Unique, comm

    print(f"--> Answer generated: {answer[:100].strip()}...")
```


```

        print(f"--> Sources identified: {sources}")

        # Return the answer and sources to be displayed in Gradio o
        return answer.strip(), sources

    except Exception as e:
        error_message = f"An error occurred: {e}"
        print(f"--> Error during chain execution: {error_message}")
        # Return error message to the user interface
        return error_message, "Error occurred"

# --- Create the Gradio Interface using Blocks API ---
print("\nSetting up Gradio interface...")

with gr.Blocks(theme=gr.themes.Soft(), title="Eleven Madison Park Q
    # Title and description for the app
    gr.Markdown(
        """
        # Eleven Madison Park – AI Q&A Assistant 
        Ask questions about the restaurant based on its website dat
        The AI provides answers and cites the source document.
        *(Examples: What are the menu prices? Who is the chef? Is i
        """
    )

    # Input component for the user's question
    question_input = gr.Textbox(
        label = "Your Question:",
        placeholder = "e.g., What are the opening hours on Saturday
        lines = 2, # Allow a bit more space for longer questions
    )

    # Row layout for the output components
    with gr.Row():
        # Output component for the generated answer (read-only)
        answer_output = gr.Textbox(label="Answer:", interactive=False)
        # Output component for the sources (read-only)
        sources_output = gr.Textbox(label="Sources:", interactive=False)

    # Row for buttons
    with gr.Row():
        # Button to submit the question
        submit_button = gr.Button("Ask Question", variant="primary")
        # Clear button to reset inputs and outputs
        clear_button = gr.ClearButton(components=[question_input, a

    # Add some example questions for users to try
    gr.Examples(
        examples=[
            "What are the different menu options and prices?",
            "Who is the head chef?",
            "What is Magic Farms?",
        ],
        inputs=question_input, # Clicking example loads it into th
        # We could potentially add outputs=[answer_output, sources_
        # but that requires running the chain for each example befo

```



```
        cache_examples=False, # Don't pre-compute results for exam
    )

    # --- Connect the Submit Button to the Function ---
    # When submit_button is clicked, call 'ask_emp_assistant'
    # Pass the value from 'question_input' as input
    # Put the returned values into 'answer_output' and 'sources_out
    submit_button.click(fn = ask_elevenmadison_assistant, inputs =

print("Gradio interface defined.")

# --- Launch the Gradio App ---
print("\nLaunching Gradio app... (Stop the kernel or press Ctrl+C i
# demo.launch() # Launch locally in the notebook or browser
demo.launch()
```

Setting up Gradio interface...

Gradio interface defined.

Launching Gradio app... (Stop the kernel or press Ctrl+C in terminal to quit)

\* Running on local URL: http://127.0.0.1:7860

To create a public link, set `share=True` in `launch()`.

Out[60]:

**PRACTICE OPPORTUNITY:**

- Expand on the example questions to include the ones below. Rerun the app and test them out.
  - "Do I need a reservation for the bar?"
  - "What is the dress code?"
  - "Can I buy gift cards?"
- Change the text on the submit button from `Ask Question` to `Ask Eleven Madison Park AI 🤖`. Where do you make this change?
- We already added a `gr.ClearButton` in the code above. Test it out! Ask a question, get an answer, then click the "Clear All" button. Does it clear the question, answer, and sources fields?

In [ ]:

In [ ]:

In [ ]:

In [ ]:

## PRACTICE OPPORTUNITY SOLUTIONS

**PRACTICE OPPORTUNITY SOLUTION:**

- Display the last 750 characters in the loaded document
- Perform a sanity check by comparing the printed characters to `eleven_madison_park_data.txt` file
- Print the email and the phone number of the restaurant

```
In [ ]: # Let's display the last characters of the loaded content
print(raw_documents[0].page_content[-750:] + "...")
```

**PRACTICE OPPORTUNITY SOLUTION:**

- Change `chunk_size` to `500` in the `RecursiveCharacterTextSplitter` and re-run the cell. How many chunks do you get now? Is it more or less than before? Change it back to `1000`
- Change `chunk_overlap` to `0` and re-run the cell. Does the number of chunks change drastically? What problem might setting overlap to 0 cause? Change it back to `150`
- Print the last example chunk and its metadata and re-run the cell.

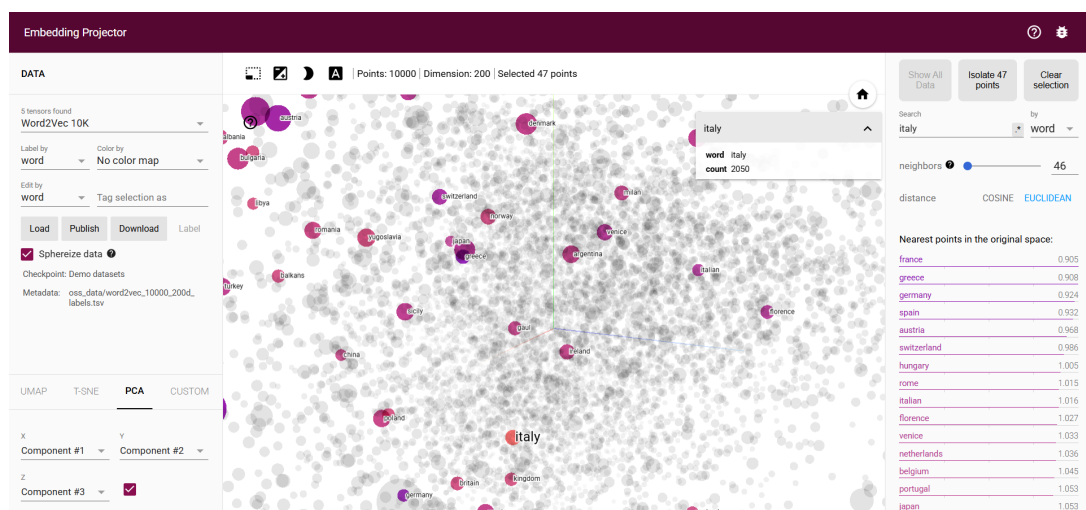
## What information is stored in the metadata ? Why is the 'source' important?

```
In [ ]: # Let's display an example chunk and its metadata
print("\n--- Example Chunk (Last Chunk) ---")
print(documents[-1].page_content)
print("\n--- Metadata for Last Chunk---")
print(documents[-1].metadata)
```

1. Changing `chunk_size=500` will result in *more* chunks being created because the original document is being divided into smaller pieces. The exact number depends on the total text length and overlap.
2. Changing `chunk_overlap=0` might slightly change the total number of chunks, but usually not by a large amount compared to changing the chunk size. The potential problem with zero overlap is that if an important sentence or idea happens to fall exactly on the boundary between two chunks, it might get cut in half, and the full context could be lost in both resulting chunks. Overlap mitigates this by ensuring some context is repeated.
3. The `'source'` key is crucial because it links the chunk back to its origin file, allowing the RAG chain to later report this source when providing citations for its answer.

### PRACTICE OPPORTUNITY SOLUTION:

- Using Tensorflow Embeddings Projector, explore the nearest points to "Italy"
- Choose 2 additional different words that you like
- Tensorflow Ebeddings projector: <https://projector.tensorflow.org/>



### PRACTICE OPPORTUNITY SOLUTION:

- Change the `test_query` variable to ask one of the following

questions. Re-run the cell for each question. Do the retrieved document snippets seem relevant to your question?

- "Who is Daniel Humm?"
  - "Is there a dress code?"
  - "Tell me about the partnership with Magic Farms."
  - "How do I make a reservation?"
- Adjust the value of `k` to `k=1` and then to `k=5`. Re-run the cell. How does changing `k` affect the number of documents retrieved?

```
In [ ]: # Let's perform a similarity search in our vector store
print("\n--- Testing Similarity Search in Vector Store ---")
test_query = "Tell me about the partnership with Magic Farms."
print(f"Searching for documents similar to: '{test_query}'")

# Perform a similarity search. 'k=2' retrieves the top 3 most similar
try:
    similar_docs = vector_store.similarity_search(test_query, k = 1)
    print(f"\nFound {len(similar_docs)} similar documents:")

    # Display snippets of the retrieved documents and their sources
    for i, doc in enumerate(similar_docs):
        print(f"\n--- Document {i+1} ---")
        # Displaying the first 700 chars for brevity
        content_snippet = doc.page_content[:700].strip() + "..."
        source = doc.metadata.get("source", "Unknown Source") # Get
        print(f"Content Snippet: {content_snippet}")
        print(f"Source: {source}")

except Exception as e:
    print(f"An error occurred during similarity search: {e}")
```

### PRACTICE OPPORTUNITY SOLUTION:

- Look at the `result` dictionary printed by the chain test. What are the key pieces of information it contains?
- Change `temperature=0` to `temperature=1.3` in the `llm = OpenAI(...)` line. Re-run the cell and ask the *same* `chain_test_query`. Does the wording of the answer change slightly? Change it back to `0`.
- Set `verbose=True` and `return_source_documents=True` in the `RetrievalQAWithSourcesChain.from_chain_type(...)` line. Re-run the cell. What extra information do you see printed? Set it back to `False`.

```
In [54]: result
```

```
Out[54]: {'question': 'What kind of food does Eleven Madison Park serve?',
          'answer': ' Eleven Madison Park serves a discounted plant-based menu and farm-sourced à la carte snacks, reservations are required.\n',
          'sources': 'https://www.elevenmadisonpark.com/faq',
          'source_documents': [Document(metadata={'source': 'eleven_madison_park_data.txt'}, page_content='Welcome to Eleven Madison Park Eleven Madison Park is a fine dining restaurant in the heart of New York City. Overlooking Madison Square Park—one of Manhattan’s most beautiful green spaces—we sit at the base of a historic Art Deco building on the corner of East 24th Street and Madison Avenue. Since opening in 1998, we underwent a full-scale renovation and redesign in the summer of 2017. Chef Daniel Humm has owned the restaurant since 2011, during which time we have evolved considerably in both cuisine and experience. In 2021, we transitioned to a fully plant-based menu, using no animal products. That same year, we partnered with Magic Farms , which grows produce exclusively for our seasonal menus. Guests can enjoy a full tasting menu, a five-course menu, or a bar menu. The bar also offers à la carte snacks, as well as wine and cocktails. Reservations are available via Resy , and bar seating is open for both walk-ins and reservations. Hours: Monday to Wednesday: 5:30 pm to 10 pm'),
                               Document(metadata={'source': 'eleven_madison_park_data.txt'}, page_content='Source: https://www.elevenmadisonpark.com/ourrestaurant\nTitle: About – Eleven Madison Park\nContent:'),
                               Document(metadata={'source': 'eleven_madison_park_data.txt'}, page_content='Source: https://www.elevenmadisonpark.com/faq\nTitle: FAQs – Eleven Madison Park\nContent:')]}}
```

1. **Chain Output:** The `result` dictionary from `RetrievalQAWithSourcesChain` typically contains:
  - `'question'` : The original input question.
  - `'answer'` : The textual answer generated by the LLM based on the retrieved context.
  - `'sources'` : A string listing the source(s) identified (often the filename from the metadata of the used documents).
  - `'source_documents'` : If `return_source_documents=True` , this key holds a list of the actual Langchain `Document` objects that were retrieved and passed to the LLM.
2. **Temperature Effect:** Changing `temperature=1.3` introduces more randomness into the LLM's generation process. While the answer should still be based on the retrieved context, the exact phrasing, sentence structure, or choice of words might vary slightly each time you run the query, compared to the more deterministic output with `temperature=0` .
3. **Verbose Mode:** Setting `verbose=True` causes Langchain to print detailed internal logs.

## PRACTICE OPPORTUNITY SOLUTION:

- Expand on the example questions to include the ones below. Rerun the app and test them out.
  - "Do I need a reservation for the bar?"
  - "What is the dress code?"
  - "Can I buy gift cards?"
- Change the text on the submit button from **Ask Question** to **Ask Eleven Madison Park AI 🤖**. Where do you make this change?
- We already added a **gr.ClearButton** in the code above. Test it out! Ask a question, get an answer, then click the "Clear All" button. Does it clear the question, answer, and sources fields?

```
In [65]: # --- Define the Function for Gradio ---

# This function takes the user's input, runs the chain, and formats
# Ensure the `qa_chain` variable is accessible in this scope.
def ask_elevenmadison_assistant(user_query):
    """
    Processes the user query using the RAG chain and returns format
    """
    print(f"\nProcessing Gradio query: '{user_query}'")
    if not user_query or user_query.strip() == "":
        print("--> Empty query received.")
        return "Please enter a question.", "" # Handle empty input

    try:
        # Run the query through our RAG chain
        result = qa_chain.invoke({"question": user_query})

        # Extract answer and sources
        answer = result.get("answer", "Sorry, I couldn't find an an
        sources = result.get("sources", "No specific sources identi

        # Basic formatting for sources (especially if it just retur
        if sources == DATA_FILE_PATH:
            sources = f"Retrieved from: {DATA_FILE_PATH}"
        elif isinstance(sources, list): # Handle potential list of
            sources = ", ".join(list(set(sources))) # Unique, comm

        print(f"--> Answer generated: {answer[:100].strip()}...")
        print(f"--> Sources identified: {sources}")

        # Return the answer and sources to be displayed in Gradio o
        return answer.strip(), sources

    except Exception as e:
        error_message = f"An error occurred: {e}"
        print(f"--> Error during chain execution: {error_message}")
        # Return error message to the user interface
        return error_message, "Error occurred"

# --- Create the Gradio Interface using Blocks API ---
print("\nSetting up Gradio interface...")
```

```

with gr.Blocks(theme=gr.themes.Soft(), title="EMP Q&A Assistant") as a
    # Title and description for the app
    gr.Markdown(
        """
        # Eleven Madison Park - AI Q&A Assistant 🗨️
        Ask questions about the restaurant based on its website data.
        The AI provides answers and cites the source document.
        *(Examples: What are the menu prices? Who is the chef? Is it open on Saturday?)
        """
    )

    # Input component for the user's question
    question_input = gr.Textbox(
        label = "Your Question:",
        placeholder = "e.g., What are the opening hours on Saturday",
        lines = 2, # Allow a bit more space for longer questions
    )

    # Row layout for the output components
    with gr.Row():
        # Output component for the generated answer (read-only)
        answer_output = gr.Textbox(label="Answer:", interactive=False)
        # Output component for the sources (read-only)
        sources_output = gr.Textbox(label="Sources:", interactive=False)

    # Row for buttons
    with gr.Row():
        # Button to submit the question
        submit_button = gr.Button("Ask Eleven Madison Park AI 🤖",)
        # Clear button to reset inputs and outputs
        clear_button = gr.ClearButton(components=[question_input, answer_output, sources_output])

    # Add some example questions for users to try
    gr.Examples(
        examples=[
            "What are the different menu options and prices?",
            "Who is the head chef?",
            "What is Magic Farms?",
            "Do I need a reservation for the bar?",
            "What is the dress code?",
            "Can I buy gift cards?"],
        inputs=question_input, # Clicking example loads it into the input
        # We could potentially add outputs=[answer_output, sources_output]
        # but that requires running the chain for each example before
        cache_examples=False, # Don't pre-compute results for examples
    )

    # --- Connect the Submit Button to the Function ---
    # When submit_button is clicked, call 'ask_emp_assistant'
    # Pass the value from 'question_input' as input
    # Put the returned values into 'answer_output' and 'sources_output'
    submit_button.click(fn = ask_elevenmadison_assistant, inputs = question_input, outputs = [answer_output, sources_output])

print("Gradio interface defined.")

```

```
# --- Launch the Gradio App ---  
print("\nLaunching Gradio app... (Stop the kernel or press Ctrl+C i  
# demo.launch() # Launch locally in the notebook or browser  
demo.launch())
```

Setting up Gradio interface...

Gradio interface defined.

Launching Gradio app... (Stop the kernel or press Ctrl+C in terminal to quit)

\* Running on local URL: <http://127.0.0.1:7861>

To create a public link, set `share=True` in `launch()`.

Out[65]:



## Summary



Retrieval-Augmented Generation (RAG) was presented as a robust approach for combining retrieval systems with generative language models to deliver accurate, context-aware outputs.

An end-to-end RAG pipeline was developed using LangChain for orchestration, OpenAI for embeddings, and ChromaDB for efficient vector search.

Key implementation techniques such as document preprocessing, embedding generation, and vector storage were demonstrated following best practices.

You can build powerful AI applications easily and quickly using an interactive Gradio interface which was designed to showcase answers alongside cited sources, ensuring transparency & enhancing user trust.



© All rights reserved for Dr. Ryan Ahmed @Stemplicity/ Inc.



# Thank you!