1.0

Installing python in local system

Steps

Download python from browser , click [Download Python | Python.org](https://www.python.org/downloads/) to open download page

A screenshot of a computer

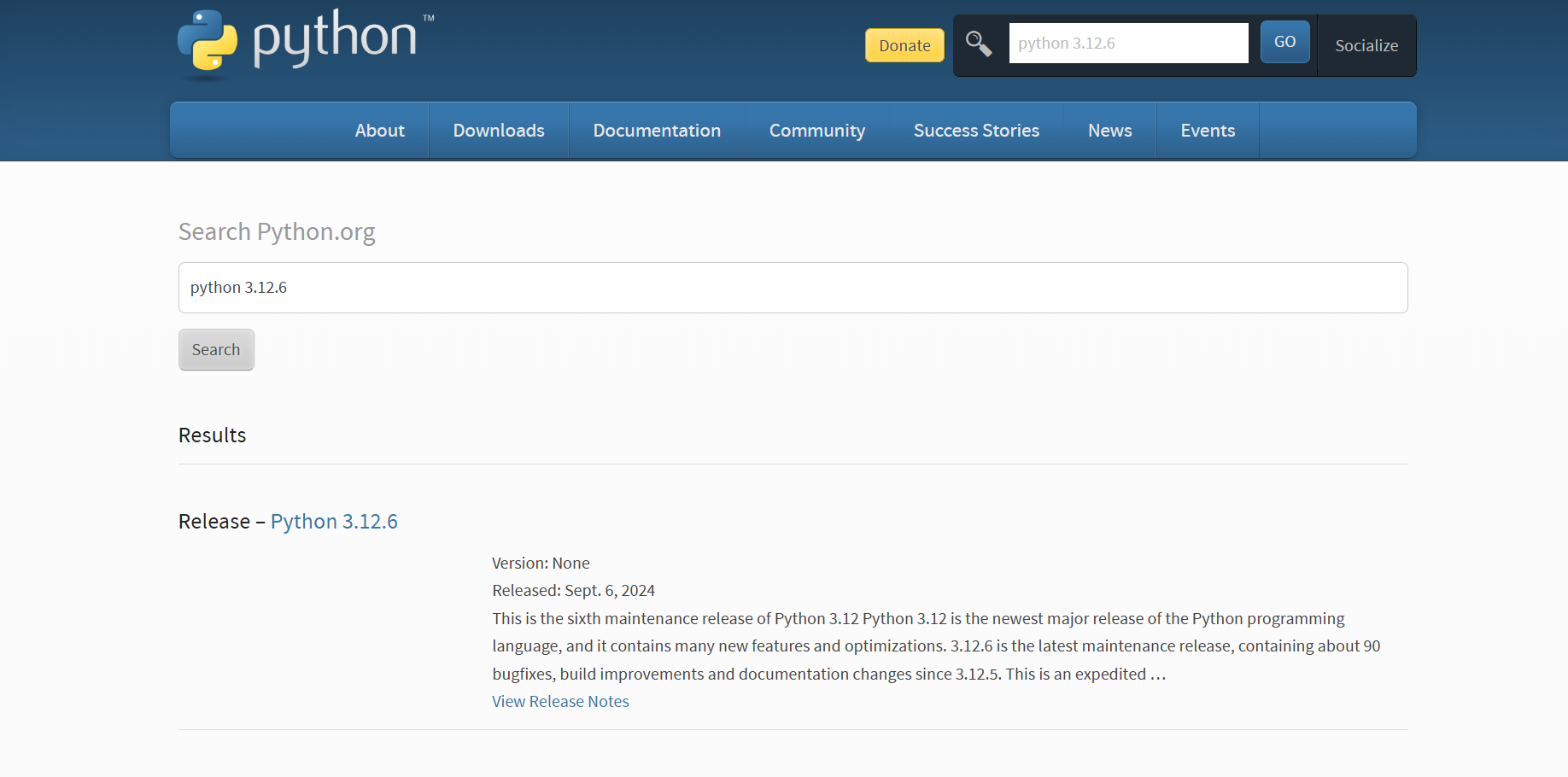
AI-generated content may be incorrect.

Now search for python 3.12.6 and Click on go button

A screenshot of a computer

AI-generated content may be incorrect.

Now new page will open 🡪 click on the python 3.12.6



Scroll down until u see this files block 🡪 click windows installer (64-bit)

A screenshot of a computer

AI-generated content may be incorrect.

Now you can see python installer will start download wait until it completes

Now install the downloaded file

1.1

Setting path in environment

Search for IDLE A screenshot of a computer

AI-generated content may be incorrect.

Click on file location



Copy the path



Now search for 🡪 Edit the system environment variable

A screenshot of a computer

AI-generated content may be incorrect.

Open it

Now u can see this interface

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Click on environment variables

You can see Path below system variables , select it and click edit

A screenshot of a computer program

AI-generated content may be incorrect.

Click on New 🡪 paste that copied address of python 🡪 click ok

A screenshot of a computer program

AI-generated content may be incorrect.

1.2

Verify Installation

Open command prompt 🡪 inside it type command

Python –version

Output should show python with version

Like below image

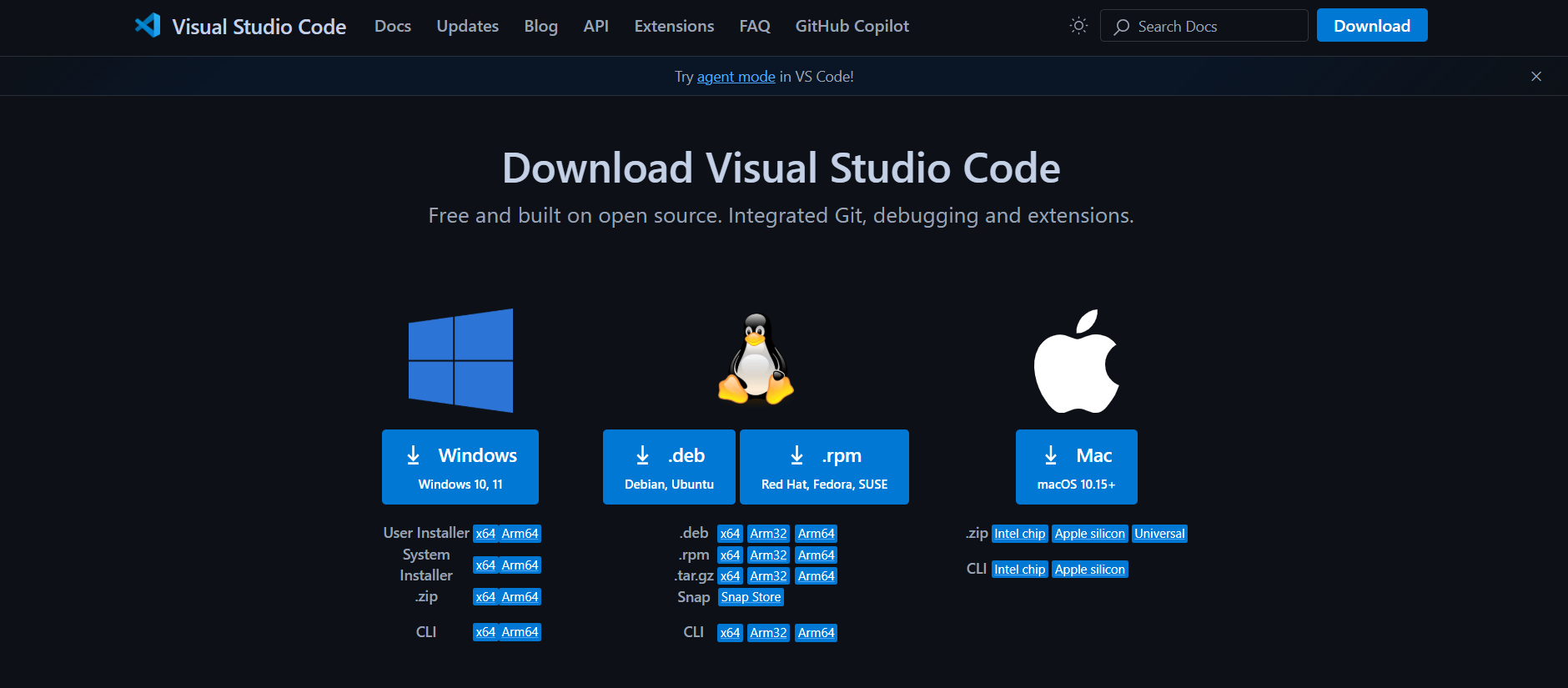
A screenshot of a computer program

AI-generated content may be incorrect.

1.3 Downlaod Visual studio code

Link🡪[Download Visual Studio Code - Mac, Linux, Windows](https://code.visualstudio.com/download)

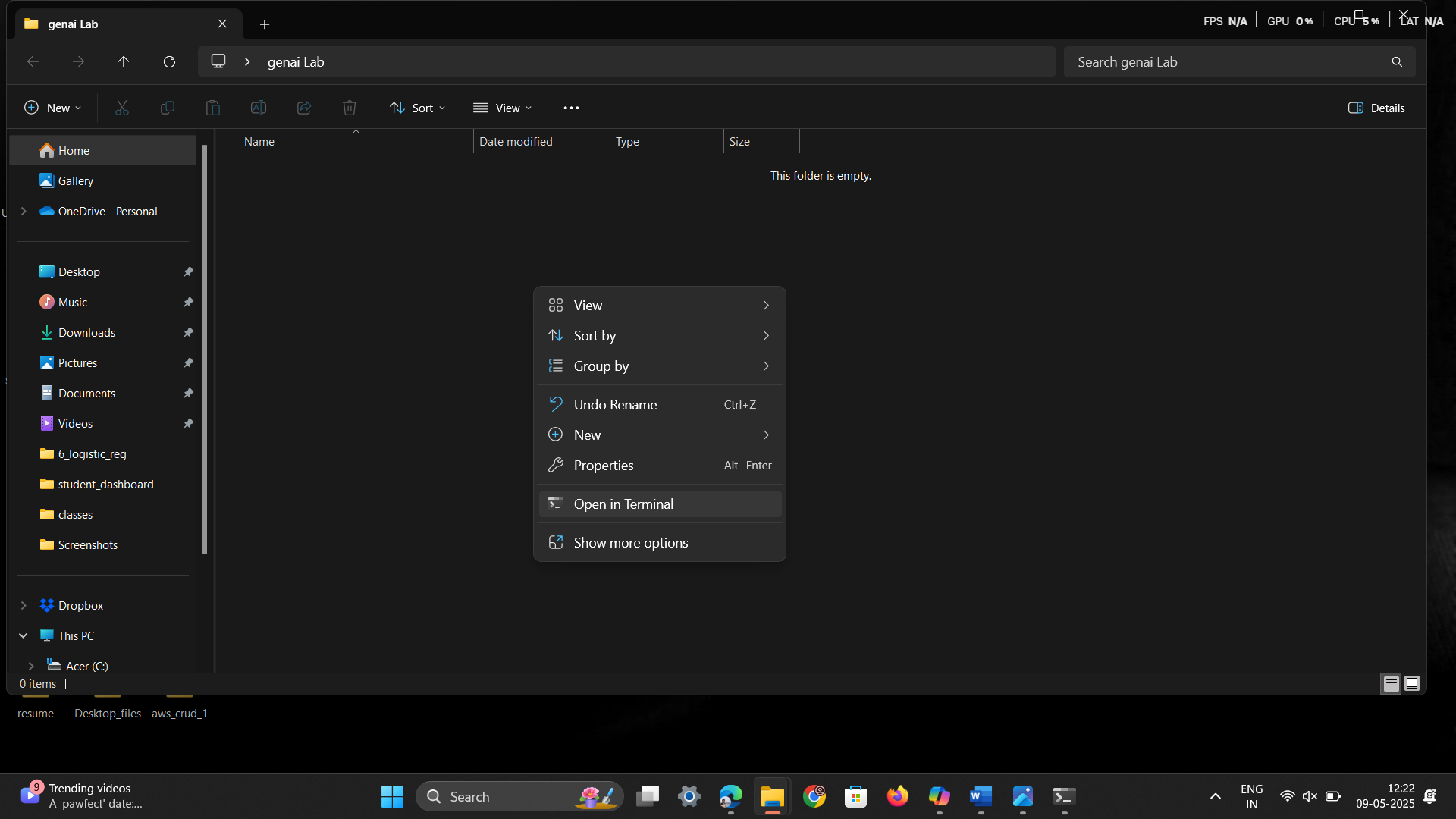
Select the suitable OS



Before starting the Lab programs

Create folder

Open that folder in command prompt

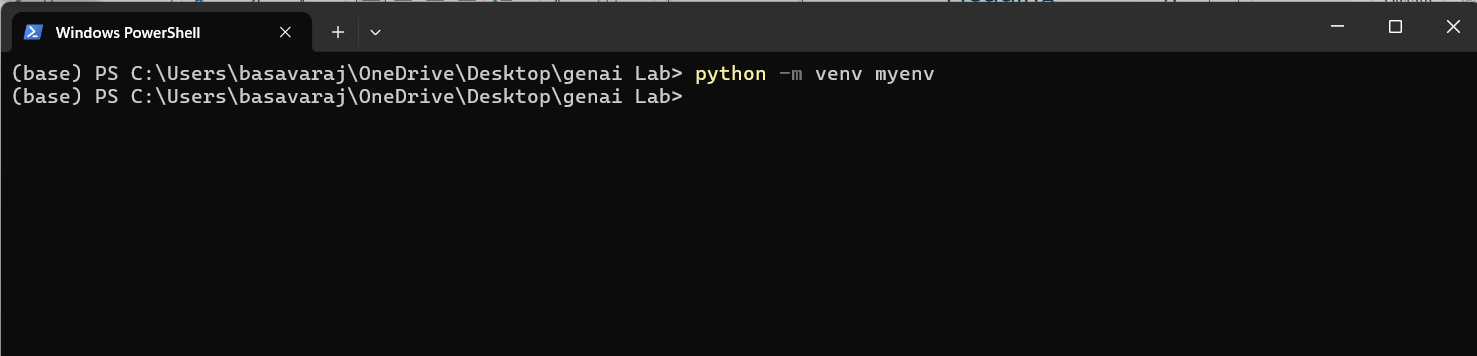


1.4 Create a Virtual Environment

Run the following command in your terminal or command prompt:

🡪python -m venv myenv

Replace myenv with your preferred environment name.

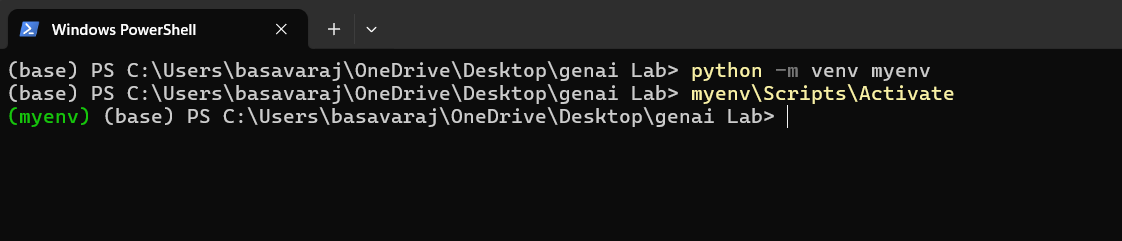


Now Virtual Environment file is created

1.5 Activate the Virtual Environment

Command 🡪 myenv\Scripts\Activate

After running that command, you can see (myenv) our virtual environment is running



For first time u can see the error

To overcome this error

Open PowerShell, run as administrator

Handling Remote Execution Policy Restrictions (Windows)

Run this command 🡪 Set-ExecutionPolicy Unrestricted -Scope CurrentUser

A screenshot of a computer program

AI-generated content may be incorrect.

Now close the terminal open the folder 🡪 open in terminal🡪 run activation command(myenv\Scripts\Activate)

Still face error

Try this command

🡪 Set-ExecutionPolicy RemoteSigned -Scope CurrentUser

Program -1

Open created folder in terminal , u can see that folder path in terminal

Type command 🡪 code .

Folder will open in Vs code

Open terminal in vs code

Activate the virtual environment , command 🡪 myenv\Scripts\Activate

Remember 🡪 virtual environment should be activated inside that only u should install lib and run the programs

To check the activation check the below image (myven) should appear before the folder path

A close-up of a computer screen

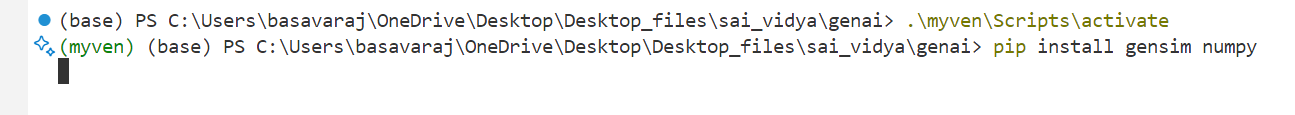
AI-generated content may be incorrect.

Click on new file name it as 1\_program.py

Install the required libraries

To install run the below command in terminal

* pip install gensim numpy



Type the code

# Import required libraries

from gensim.models import Word2Vec

from gensim.parsing.preprocessing import STOPWORDS

# Step 1: Define corpus

corpus = [

    'king is a strong man', 'queen is a wise woman', 'boy is a young man',

    'girl is a young woman', 'prince is a young', 'prince will be strong',

    'princess is young', 'man is strong', 'woman is pretty',

    'prince is a boy', 'prince will be king', 'princess is a girl',

    'princess will be queen'

]

# Step 2: Tokenize and remove stopwords

statements\_listt = [sentence.split() for sentence in corpus]

documents = [[word for word in doc if word not in STOPWORDS] for doc in statements\_listt]

# Show the cleaned documents

print("Cleaned Documents:")

print(documents)

# Step 3: Train the Word2Vec model

model = Word2Vec(documents, vector\_size=3, window=3, min\_count=1)

# Step 4: Vector operations

vector1 = model.wv['king']

vector2 = model.wv['man']

sum\_vector = vector1 + vector2

diff\_vector = vector1 - vector2

print("\nSum vector (king + man):", sum\_vector)

print("Difference vector (king - man):", diff\_vector)

# Step 5: Cosine similarity

similarity = model.wv.similarity('king', 'queen')

print(f"\nCosine Similarity between 'king' and 'queen': {similarity}")

# Step 6: Most similar words

similar\_words = model.wv.most\_similar('king', topn=5)

print("\nMost Similar words to 'king':", similar\_words)

# Step 7: Analogy example

analogy\_vector = model.wv['king'] - model.wv['man'] + model.wv['woman']

most\_similar = model.wv.most\_similar(positive=[analogy\_vector], topn=1)

print("\nAnalogy Result (king - man + woman):", most\_similar)

save the file

to run the code run the below code in terminal

🡪 python 1\_program.py

A white background with colorful text

AI-generated content may be incorrect.

Output :

Cleaned Documents:

[['king', 'strong', 'man'], ['queen', 'wise', 'woman'], ['boy', 'young', 'man'], ['girl', 'young', 'woman'], ['prince', 'young'], ['prince', 'strong'], ['princess', 'young'], ['man', 'strong'], ['woman', 'pretty'], ['prince', 'boy'], ['prince', 'king'], ['princess', 'girl'], ['princess', 'queen']]

Sum vector (king + man): [ 0.04106534 -0.03225286 -0.2932088 ]

Difference vector (king - man): [ 0.34350622 -0.46918964 0.03080189]

Cosine Similarity between 'king' and 'queen': 0.8843974471092224

Most Similar words to 'king': [('young', 0.9887178540229797), ('queen', 0.8843973875045776), ('girl', -0.036878012120723724), ('princess', -0.08420105278491974), ('wise', -0.20955052971839905)]

Analogy Result (king - man + woman): [('queen', 0.9522483944892883)]

Program 2

Open created folder in terminal , u can see that folder path in terminal

Type command 🡪 code .

Folder will open in Vs code

Open terminal in vs code

Activate the virtual environment , command 🡪 myenv\Scripts\Activate

Remember 🡪 virtual environment should be activated inside that only u should install lib and run the programs

To check the activation check the below image (myven) should appear before the folder path

A close-up of a computer screen

AI-generated content may be incorrect.

Click on new file name it , 2\_gen\_ai\_lab\_embedding.py

Install the required libraries

To install run the below command in terminal

🡪 pip install gensim matplotlib scikit-learn pandas



Type the code

import gensim

from gensim.models import Word2Vec

import re

import pandas as pd

import matplotlib.pyplot as plt

from sklearn.decomposition import PCA

# Sample domain-specific corpus (Technology)

technology\_corpus = [

    "Artificial intelligence is transforming various industries.",

    "Machine learning algorithms improve predictive analytics.",

    "Cloud computing enables scalable infrastructure for businesses.",

    "Cybersecurity is crucial for protecting sensitive data.",

    "Blockchain technology ensures secure and decentralized transactions.",

    "The Internet of Things connects smart devices seamlessly.",

    "Big data analytics helps organizations make data-driven decisions.",

    "Quantum computing has the potential to revolutionize cryptography.",

    "Edge computing brings computation closer to data sources.",

    "Natural language processing enhances human-computer interactions."

]

# Basic text preprocessing function (tokenization & lowercasing)

def simple\_tokenize(text):

    return re.findall(r'\b\w+\b', text.lower())

# Preprocess corpus manually

preprocessed\_corpus = [simple\_tokenize(sentence) for sentence in technology\_corpus]

# Train Word2Vec model

model = Word2Vec(sentences=preprocessed\_corpus, vector\_size=50, window=5, min\_count=1, workers=4)

# Select 10 domain-specific words

selected\_words = ["ai", "machine", "cloud", "cybersecurity", "blockchain", "iot", "data", "quantum", "edge", "nlp"]

# Filter selected words to include only words present in model.wv

selected\_words = [word for word in selected\_words if word in model.wv]

# Extract word embeddings for selected words

word\_vectors = [model.wv[word] for word in selected\_words if word in model.wv]

# Reduce dimensionality using PCA

pca = PCA(n\_components=2)

reduced\_vectors = pca.fit\_transform(word\_vectors)

# Create DataFrame for visualization

df\_embeddings = pd.DataFrame(reduced\_vectors, columns=["x", "y"], index=selected\_words)

# Plot embeddings

plt.figure(figsize=(10, 6))

plt.scatter(df\_embeddings["x"], df\_embeddings["y"], marker='o')

for word, (x, y) in zip(df\_embeddings.index, reduced\_vectors):

    plt.text(x, y, word, fontsize=12)

plt.xlabel("PCA Component 1")

plt.ylabel("PCA Component 2")

plt.title("Word Embeddings Visualization (Technology Domain)")

plt.show()

# Function to get semantically similar words

def get\_similar\_words(word, top\_n=5):

    if word in model.wv:

        return model.wv.most\_similar(word, topn=top\_n)

    else:

        return f"Word '{word}' not in vocabulary."

# Example usage

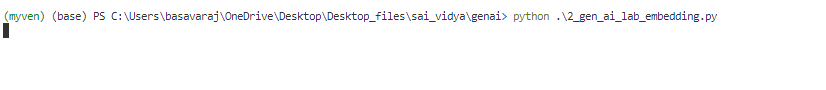
input\_word = "technology"

similar\_words = get\_similar\_words(input\_word)

print(f"Top 5 words similar to '{input\_word}':", similar\_words)

To run this program , type below command in terminal

🡪 python 2\_gen\_ai\_lab\_embedding.py



Output image:

A graph with words on it

AI-generated content may be incorrect.

Program 3

Open created folder in terminal , u can see that folder path in terminal

Type command 🡪 code .

Folder will open in Vs code

Open terminal in vs code

Activate the virtual environment , command 🡪 myenv\Scripts\Activate

Remember 🡪 virtual environment should be activated inside that only u should install lib and run the programs

To check the activation check the below image (myven) should appear before the folder path

A close-up of a computer screen

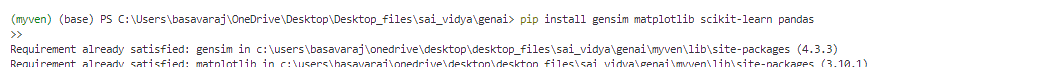
AI-generated content may be incorrect.

Click on new file name it , 3\_gen\_ai\_lab\_embedding.py

Install the required libraries

To install run the below command in terminal

🡪 pip install gensim matplotlib scikit-learn pandas



Type the code

import gensim

from gensim.models import Word2Vec

import re

import pandas as pd

import matplotlib.pyplot as plt

from sklearn.decomposition import PCA

# Sample domain-specific corpus (Medical)

medical\_corpus = [

    "The patient was diagnosed with diabetes and prescribed metformin.",

    "Hypertension can be managed with lifestyle changes and medication.",

    "The MRI scan showed abnormalities in the brain tissue.",

    "Patients with asthma should avoid exposure to allergens.",

    "The new drug showed promising results in clinical trials.",

    "Surgery was performed to remove the tumor successfully.",

    "Doctors recommend regular exercise to prevent cardiovascular diseases.",

    "The vaccine helps in boosting the immune response against infections.",

    "Genetic factors play a significant role in the onset of Alzheimer's disease.",

    "Antibiotics should be used responsibly to prevent resistance."

]

# Basic text preprocessing function (tokenization & lowercasing)

def simple\_tokenize(text):

    return re.findall(r'\b\w+\b', text.lower())

# Preprocess corpus manually

preprocessed\_corpus = [simple\_tokenize(sentence) for sentence in medical\_corpus]

print("preprocessed corpus")

print(preprocessed\_corpus)

# Train Word2Vec model

model = Word2Vec(sentences=preprocessed\_corpus, vector\_size=50, window=5, min\_count=1, workers=4)

# Extract word embeddings for visualization

words = list(model.wv.index\_to\_key)  # Get vocabulary words

print("vocabulary words")

print(words)

word\_vectors = [model.wv[word] for word in words]  # Get corresponding embeddings

print("word vectors")

print(word\_vectors)

# Reduce dimensionality using PCA

pca = PCA(n\_components=2)

reduced\_vectors = pca.fit\_transform(word\_vectors)

# Create DataFrame for visualization

df\_embeddings = pd.DataFrame(reduced\_vectors, columns=["x", "y"], index=words)

# Plot embeddings

plt.figure(figsize=(10, 6))

plt.scatter(df\_embeddings["x"], df\_embeddings["y"], marker='o')

for word, (x, y) in zip(df\_embeddings.index, reduced\_vectors):

    plt.text(x, y, word, fontsize=12)

plt.xlabel("PCA Component 1")

plt.ylabel("PCA Component 2")

plt.title("Word Embeddings Visualization (Medical Domain)")

plt.show()

#3 d plot

# Import necessary libraries

import matplotlib.pyplot as plt

from mpl\_toolkits.mplot3d import Axes3D  # Import for 3D plotting

from sklearn.decomposition import PCA

# ... (rest of your code)

# Perform PCA with 3 components

pca = PCA(n\_components=3)  # Change n\_components to 3

reduced\_vectors = pca.fit\_transform(word\_vectors)

# Create DataFrame for visualization

df\_embeddings = pd.DataFrame(reduced\_vectors, columns=["x", "y", "z"], index=words) #add z

# Plot embeddings in 3D

fig = plt.figure(figsize=(10, 8))  # Adjust figure size if needed

ax = fig.add\_subplot(111, projection='3d')  # Create a 3D subplot

ax.scatter(df\_embeddings["x"], df\_embeddings["y"], df\_embeddings["z"], marker='o')

for word, (x, y, z) in zip(df\_embeddings.index, reduced\_vectors):

    ax.text(x, y, z, word, fontsize=10)  # Adjust fontsize if needed

ax.set\_xlabel("PCA Component 1")

ax.set\_ylabel("PCA Component 2")

ax.set\_zlabel("PCA Component 3")  # Add z-axis label

ax.set\_title("Word Embeddings Visualization (3D)")

plt.show()

Run the code

Command

🡪python 3\_gen\_ai\_lab\_embedding.py



Output

A diagram of words and numbers

AI-generated content may be incorrect.

Program 4

Open created folder in terminal , u can see that folder path in terminal

Type command 🡪 code .

Folder will open in Vs code

Open terminal in vs code

Activate the virtual environment , command 🡪 myenv\Scripts\Activate

Remember 🡪 virtual environment should be activated inside that only u should install lib and run the programs

To check the activation check the below image (myven) should appear before the folder path

A close-up of a computer screen

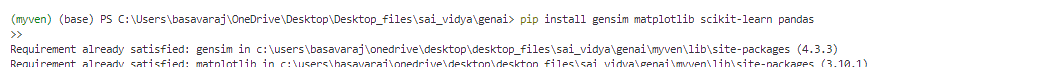
AI-generated content may be incorrect.

Click on new file name it , 4\_gen\_ai\_lab\_embedding.py

Install the required libraries

To install run the below command in terminal

🡪 pip install gensim matplotlib scikit-learn pandas



Type the code

import gensim

from gensim.models import Word2Vec

import re

import pandas as pd

import matplotlib.pyplot as plt

from sklearn.decomposition import PCA

# Sample domain-specific corpus (Technology)

technology\_corpus = [

    "Artificial intelligence is transforming various industries.",

    "Machine learning algorithms improve predictive analytics.",

    "Cloud computing enables scalable infrastructure for businesses.",

    "Cybersecurity is crucial for protecting sensitive data.",

    "Blockchain technology ensures secure and decentralized transactions.",

    "The Internet of Things connects smart devices seamlessly.",

    "Big data analytics helps organizations make data-driven decisions.",

    "Quantum computing has the potential to revolutionize cryptography.",

    "Edge computing brings computation closer to data sources.",

    "Natural language processing enhances human-computer interactions."

]

# Basic text preprocessing function (tokenization & lowercasing)

def simple\_tokenize(text):

    return re.findall(r'\b\w+\b', text.lower())

# Preprocess corpus manually

preprocessed\_corpus = [simple\_tokenize(sentence) for sentence in technology\_corpus]

# Train Word2Vec model

model = Word2Vec(sentences=preprocessed\_corpus, vector\_size=50, window=5, min\_count=1, workers=4)

# Select 10 domain-specific words

selected\_words = ["ai", "machine", "cloud", "cybersecurity", "blockchain", "iot", "data", "quantum", "edge", "nlp"]

# Extract word embeddings for selected words

word\_vectors = [model.wv[word] for word in selected\_words if word in model.wv]

# Filter selected words to include only words present in model.wv

selected\_words = [word for word in selected\_words if word in model.wv]

# Reduce dimensionality using PCA

pca = PCA(n\_components=2)

reduced\_vectors = pca.fit\_transform(word\_vectors)

# Create DataFrame for visualization

df\_embeddings = pd.DataFrame(reduced\_vectors, columns=["x", "y"], index=selected\_words)

# Plot embeddings

plt.figure(figsize=(10, 6))

plt.scatter(df\_embeddings["x"], df\_embeddings["y"], marker='o')

for word, (x, y) in zip(df\_embeddings.index, reduced\_vectors):

    plt.text(x, y, word, fontsize=12)

plt.xlabel("PCA Component 1")

plt.ylabel("PCA Component 2")

plt.title("Word Embeddings Visualization (Technology Domain)")

plt.show()

# Function to get semantically similar words

def get\_similar\_words(word, top\_n=5):

    if word in model.wv:

        return [w[0] for w in model.wv.most\_similar(word, topn=top\_n)]

    else:

        return []

# Function to enrich a GenAI prompt

def enrich\_prompt(prompt):

    words = simple\_tokenize(prompt)

    enriched\_words = []

    for word in words:

        enriched\_words.append(word)

        similar\_words = get\_similar\_words(word, top\_n=2)

        enriched\_words.extend(similar\_words)

    return " ".join(enriched\_words)

# Example usage

original\_prompt = "Explain the impact of AI on businesses."

enriched\_prompt = enrich\_prompt(original\_prompt)

print("Original Prompt:", original\_prompt)

print("Enriched Prompt:", enriched\_prompt)

# Function to compare outputs from a GenAI model

def generate\_response(prompt):

    # Placeholder function to simulate a GenAI response

    return f"Generated response for: {prompt}"

original\_response = generate\_response(original\_prompt)

enriched\_response = generate\_response(enriched\_prompt)

print("\nOriginal Response:", original\_response)

print("Enriched Response:", enriched\_response)

Run the program

Command 🡪 python 4\_gen\_ai\_lab\_embedding.py



Output

A graph with words on it

AI-generated content may be incorrect.

Program 5

Open created folder in terminal , u can see that folder path in terminal

Type command 🡪 code .

Folder will open in Vs code

Open terminal in vs code

Activate the virtual environment , command 🡪 myenv\Scripts\Activate

Remember 🡪 virtual environment should be activated inside that only u should install lib and run the programs

To check the activation check the below image (myven) should appear before the folder path

A close-up of a computer screen

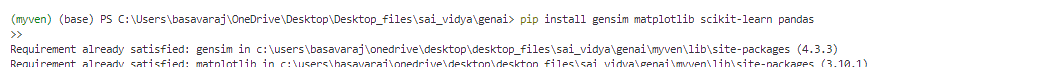
AI-generated content may be incorrect.

Click on new file name it , 5\_gen\_ai\_lab\_embedding.py

Install the required libraries

To install run the below command in terminal

🡪 pip install gensim matplotlib scikit-learn pandas



Type the code

import gensim

from gensim.models import Word2Vec

import re

import pandas as pd

import matplotlib.pyplot as plt

from sklearn.decomposition import PCA

import random

# Sample domain-specific corpus (Technology)

technology\_corpus = [

    "Artificial intelligence is transforming various industries.",

    "Machine learning algorithms improve predictive analytics.",

    "Cloud computing enables scalable infrastructure for businesses.",

    "Cybersecurity is crucial for protecting sensitive data.",

    "Blockchain technology ensures secure and decentralized transactions.",

    "The Internet of Things connects smart devices seamlessly.",

    "Big data analytics helps organizations make data-driven decisions.",

    "Quantum computing has the potential to revolutionize cryptography.",

    "Edge computing brings computation closer to data sources.",

    "Natural language processing enhances human-computer interactions."

]

# Basic text preprocessing function (tokenization & lowercasing)

def simple\_tokenize(text):

    return re.findall(r'\b\w+\b', text.lower())

# Preprocess corpus manually

preprocessed\_corpus = [simple\_tokenize(sentence) for sentence in technology\_corpus]

# Train Word2Vec model

model = Word2Vec(sentences=preprocessed\_corpus, vector\_size=50, window=5, min\_count=1, workers=4)

# Function to get semantically similar words

def get\_similar\_words(word, top\_n=5):

    if word in model.wv:

        return [w[0] for w in model.wv.most\_similar(word, topn=top\_n)]

    else:

        return []

# Function to generate a short paragraph using similar words

def generate\_paragraph(seed\_word):

    similar\_words = get\_similar\_words(seed\_word, top\_n=5)

    if not similar\_words:

        return f"No similar words found for '{seed\_word}'."

    paragraph = f"In a world driven by {seed\_word}, innovations continue to shape our future. "

    paragraph += "With advancements in " + ", ".join(similar\_words[:-1]) + f", and {similar\_words[-1]}, "

    paragraph += "society is witnessing a technological revolution that transforms industries and improves daily life."

    return paragraph

# Example usage

seed\_word = "intelligence"

generated\_paragraph = generate\_paragraph(seed\_word)

print("Seed Word:", seed\_word)

print("Generated Paragraph:", generated\_paragraph)

Run the program

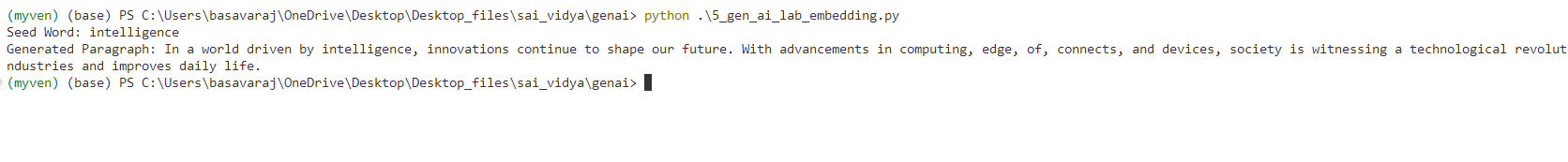
Command 🡪 python 5\_gen\_ai\_lab\_embedding.py



Output

Seed Word: intelligence

Generated Paragraph: In a world driven by intelligence, innovations continue to shape our future. With advancements in computing, edge, of, connects, and devices, society is witnessing a technological revolution that transforms industries and improves daily life.



Program 6

Open created folder in terminal , u can see that folder path in terminal

Type command 🡪 code .

Folder will open in Vs code

Open terminal in vs code

Activate the virtual environment , command 🡪 myenv\Scripts\Activate

Remember 🡪 virtual environment should be activated inside that only u should install lib and run the programs

To check the activation check the below image (myven) should appear before the folder path

A close-up of a computer screen

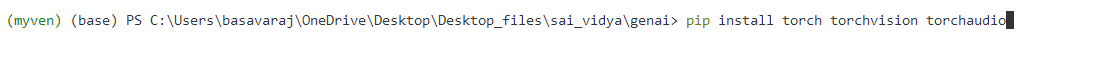
AI-generated content may be incorrect.

Click on new file name it , 6\_gen\_ai\_lab\_embedding.py

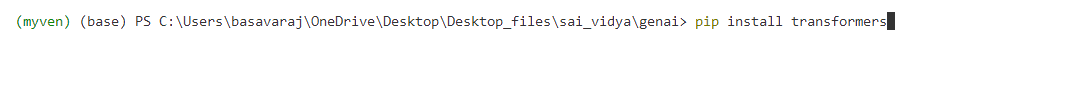
Install the required libraries

To install run the below command in terminal

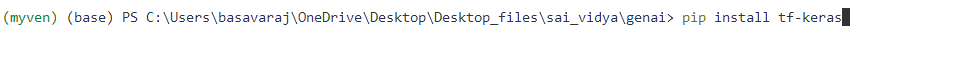
🡪pip install torch torchvision torchaudio



🡪pip install transformers



🡪pip install tf-keras



Type the code

from transformers import pipeline

# Load pre-trained sentiment analysis pipeline

sentiment\_pipeline = pipeline("sentiment-analysis")

# Function to analyze sentiment

def analyze\_sentiment(text):

    result = sentiment\_pipeline(text)

    return result[0]

# Example sentences for sentiment analysis

sentences = [

    "I love the new AI advancements, they are truly revolutionary!",

    "The new software update is frustrating and full of bugs.",

    "Customer service was amazing, very helpful and responsive.",

    "I'm disappointed with the product quality, not what I expected.",

    "This experience has been wonderful, highly recommend!"

]

# Analyze sentiment for each sentence

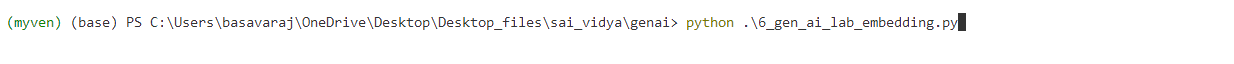
for sentence in sentences:

    sentiment\_result = analyze\_sentiment(sentence)

    print(f"Sentence: {sentence}\nSentiment: {sentiment\_result}\n")

Run the program

Command 🡪 python 6\_gen\_ai\_lab\_embedding.py



Output



Program 7

Open created folder in terminal , u can see that folder path in terminal

Type command 🡪 code .

Folder will open in Vs code

Open terminal in vs code

Activate the virtual environment , command 🡪 myenv\Scripts\Activate

Remember 🡪 virtual environment should be activated inside that only u should install lib and run the programs

To check the activation check the below image (myven) should appear before the folder path

A close-up of a computer screen

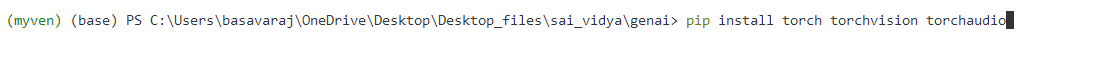
AI-generated content may be incorrect.

Click on new file name it , 7\_gen\_ai\_lab\_embedding.py

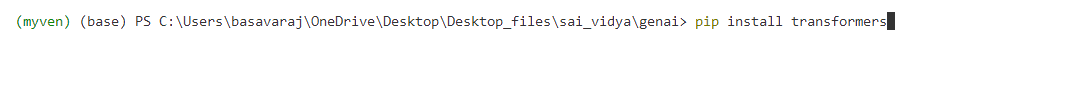
Install the required libraries

To install run the below command in terminal

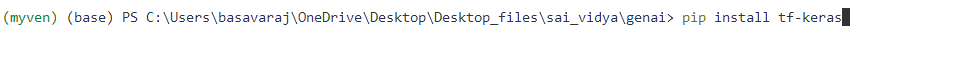
🡪pip install torch torchvision torchaudio



🡪pip install transformers



🡪pip install tf-keras



Type the code

from transformers import pipeline

# Load pre-trained summarization pipeline

summarization\_pipeline = pipeline("summarization")

# Function to summarize text

def summarize\_text(text, max\_length=100, min\_length=30):

    summary = summarization\_pipeline(text, max\_length=max\_length, min\_length=min\_length, do\_sample=False)

    return summary[0]['summary\_text']

# Example passage for summarization

long\_text = (

    "Artificial intelligence is rapidly transforming various industries, "

    "enabling automation, improving efficiency, and enhancing decision-making processes. "

    "With the rise of machine learning and deep learning models, businesses can now process large amounts of data "

    "more effectively, uncover hidden patterns, and gain valuable insights. "

    "However, ethical concerns, data privacy, and bias in AI algorithms remain significant challenges. "

    "As technology advances, organizations and policymakers must collaborate to ensure responsible AI development and deployment."

)

# Obtain summarized text

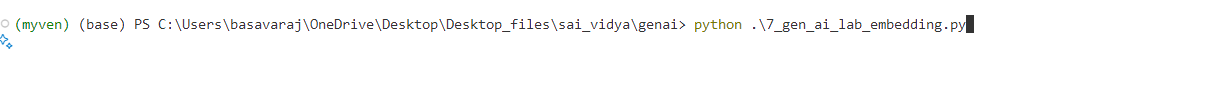
summary\_result = summarize\_text(long\_text)

print("Original Text:", long\_text)

print("Summarized Text:", summary\_result)

Run the program

Command🡪 python 7\_gen\_ai\_lab\_embedding.py



Output

Original Text: Artificial intelligence is rapidly transforming various industries, enabling automation, improving efficiency, and enhancing decision-making processes. With the rise of machine learning and deep learning models, businesses can now process large amounts of data more effectively, uncover hidden patterns, and gain valuable insights. However, ethical concerns, data privacy, and bias in AI algorithms remain significant challenges. As technology advances, organizations and policymakers must collaborate to ensure responsible AI development and deployment.

Summarized Text: Artificial intelligence is rapidly transforming various industries, enabling automation, improving efficiency, and enhancing decision-making processes . As technology advances, organizations and policymakers must collaborate to ensure responsible AI development and deployment .

A close-up of a text

AI-generated content may be incorrect.

Program 8

Open created folder in terminal , u can see that folder path in terminal

Type command 🡪 code .

Folder will open in Vs code

Open terminal in vs code

Activate the virtual environment , command 🡪 myenv\Scripts\Activate

Remember 🡪 virtual environment should be activated inside that only u should install lib and run the programs

To check the activation check the below image (myven) should appear before the folder path

A close-up of a computer screen

AI-generated content may be incorrect.

Create sample.txt file inside the same folder

Type this text inside the txt file🡪

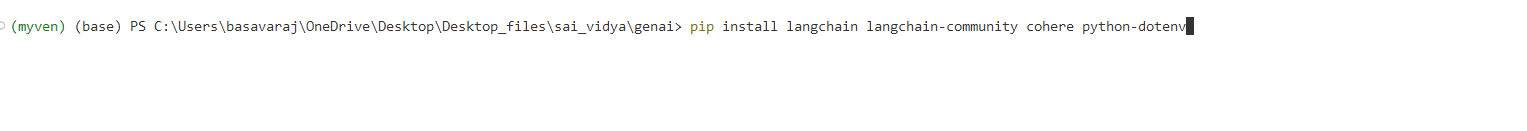
Climate change is a long-term alteration of temperature and typical weather patterns in a place. It could refer to a particular location or the planet as a whole. Climate change may cause weather patterns to be less predictable. These unexpected weather patterns can make it difficult to maintain and grow crops in regions that rely on farming because expected temperature and rainfall levels can no longer be relied on. Climate change has also been connected with other damaging weather events such as more frequent hurricanes, floods, downpours, and winter storms. The melting of glaciers, rising sea levels, and shifting wildlife populations are all effects of climate change.

Click on new file name it , 8\_gen\_ai\_lab\_embedding.py

Install the required libraries

To install run the below command in terminal

🡪 pip install langchain langchain-community cohere python-dotenv



Type the code

import os

from langchain\_core.prompts import PromptTemplate

from langchain\_community.llms import Cohere

# Set your Cohere API key

os.environ["COHERE\_API\_KEY"] = "Your\_cohere\_api\_key"

# Path to your local file

file\_path = "sample.txt"  # Change this to the correct local path if needed

# Read the file content

try:

    with open(file\_path, "r", encoding="utf-8") as file:

        document\_text = file.read()

    print("📄 File loaded successfully!\n")

    print(document\_text[:500])  # Show first 500 characters

except FileNotFoundError:

    print(f"❌ File not found: {file\_path}")

    exit(1)

except Exception as e:

    print(f"⚠️ An error occurred: {e}")

    exit(1)

# Load Cohere model

llm = Cohere(model="command", cohere\_api\_key=os.environ["COHERE\_API\_KEY"])

# Define prompt template

prompt = PromptTemplate(

    input\_variables=["input\_text"],

    template="Summarize this text:\n\n{input\_text}"

)

# Format and generate response

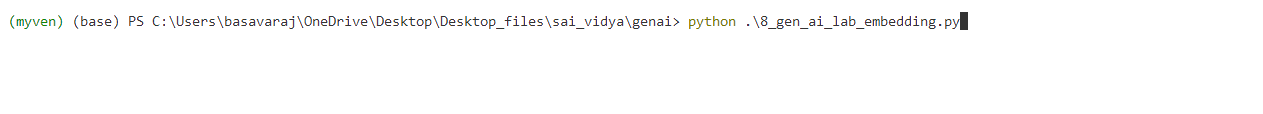
formatted\_prompt = prompt.format(input\_text=document\_text)

response = llm.invoke(formatted\_prompt)

print("\n📝 Summary:\n", response)

Run the code

Command🡪 python 8\_gen\_ai\_lab\_embedding.py



Output

📄 File loaded successfully!

Climate change is a long-term alteration of temperature and typical weather patterns in a place. It could refer to a particular location or the planet as a whole. Climate change may cause weather patterns to be less predictable. These unexpected weather patterns can make it difficult to maintain and grow crops in regions that rely on farming because expected temperature and rainfall levels can no longer be relied on. Climate change has also been connected with other damaging weather events such

C:\Users\basavaraj\OneDrive\Desktop\Desktop\_files\sai\_vidya\genai\8\_gen\_ai\_lab\_embedding.py:31: LangChainDeprecationWarning: The class `Cohere` was deprecated in LangChain 0.1.14 and will be removed in 1.0. An updated version of the class exists in the :class:`~langchain-cohere package and should be used instead. To use it run `pip install -U :class:`~langchain-cohere` and import as `from :class:`~langchain\_cohere import Cohere``.

llm = Cohere(model="command", cohere\_api\_key=os.environ["COHERE\_API\_KEY"])

📝 Summary:

Climate change refers to the long-term alteration of temperature and weather patterns, which has a range of effects including unpredictable weather patterns, extreme weather events, melting glaciers, rising sea levels, and shifting wildlife populations.

A close up of text

AI-generated content may be incorrect.

Program 9

Open created folder in terminal , u can see that folder path in terminal

Type command 🡪 code .

Folder will open in Vs code

Open terminal in vs code

Activate the virtual environment , command 🡪 myenv\Scripts\Activate

Remember 🡪 virtual environment should be activated inside that only u should install lib and run the programs

To check the activation check the below image (myven) should appear before the folder path

A close-up of a computer screen

AI-generated content may be incorrect.

Create sample.txt file inside the same folder

Type this text inside the txt file🡪

Climate change is a long-term alteration of temperature and typical weather patterns in a place. It could refer to a particular location or the planet as a whole. Climate change may cause weather patterns to be less predictable. These unexpected weather patterns can make it difficult to maintain and grow crops in regions that rely on farming because expected temperature and rainfall levels can no longer be relied on. Climate change has also been connected with other damaging weather events such as more frequent hurricanes, floods, downpours, and winter storms. The melting of glaciers, rising sea levels, and shifting wildlife populations are all effects of climate change.

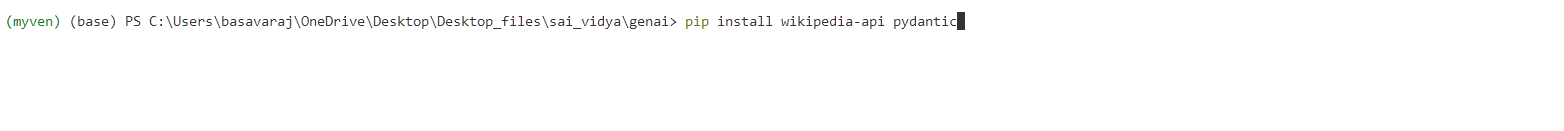
Click on new file name it , 9\_gen\_ai\_lab\_embedding.py

Install the required libraries

To install run the below command in terminal

🡪 pip install wikipedia-api pydantic

pip install wikipedia



Type the code

from pydantic import BaseModel

import wikipedia

import re

def extract\_info(page\_content: str, keyword: str) -> str:

    """Extract relevant information based on keyword search."""

    pattern = re.compile(rf'({keyword}.\*?)\n', re.IGNORECASE)

    match = pattern.search(page\_content)

    return match.group(1) if match else "Not Available"

class InstitutionInfo(BaseModel):

    name: str

    founder: str

    founded\_year: str

    branches: str

    employees: str

    summary: str

def fetch\_institution\_details(institution\_name: str) -> InstitutionInfo:

    try:

        page = wikipedia.page(institution\_name)

        content = page.content

        founder = extract\_info(content, "Founder")

        founded\_year = extract\_info(content, "Founded")

        branches = extract\_info(content, "Campuses|Branches|Locations")

        employees = extract\_info(content, "Employees|Staff")

        summary = " ".join(page.summary.split(".")[:2])  # First two sentences

        return InstitutionInfo(

            name=institution\_name,

            founder=founder,

            founded\_year=founded\_year,

            branches=branches,

            employees=employees,

            summary=summary

        )

    except wikipedia.exceptions.PageError:

        return InstitutionInfo(

            name=institution\_name,

            founder="Not Available",

            founded\_year="Not Available",

            branches="Not Available",

            employees="Not Available",

            summary="No information found on Wikipedia."

        )

if \_\_name\_\_ == "\_\_main\_\_":

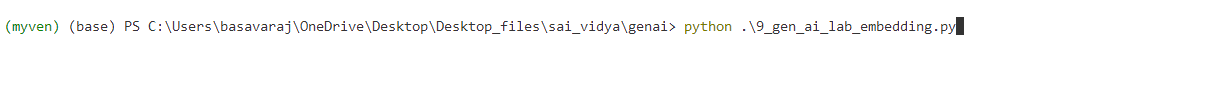
    institution\_name = input("Enter Institution Name: ")

    result = fetch\_institution\_details(institution\_name)

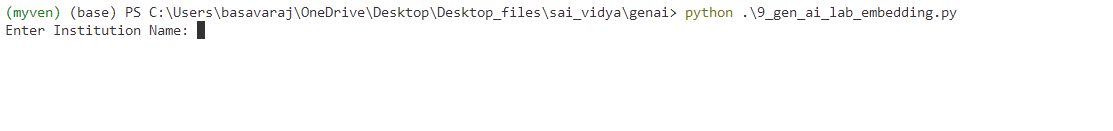
    print(result.model\_dump\_json(indent=4))

Run the program

Command🡪 python 9\_gen\_ai\_lab\_embedding.py



It as for institute name, enter the institute name



Output

A white background with text

AI-generated content may be incorrect.

Program 10

Open created folder in terminal , u can see that folder path in terminal

Type command 🡪 code .

Folder will open in Vs code

Open terminal in vs code

Activate the virtual environment , command 🡪 myenv\Scripts\Activate

Remember 🡪 virtual environment should be activated inside that only u should install lib and run the programs

To check the activation check the below image (myven) should appear before the folder path

A close-up of a computer screen

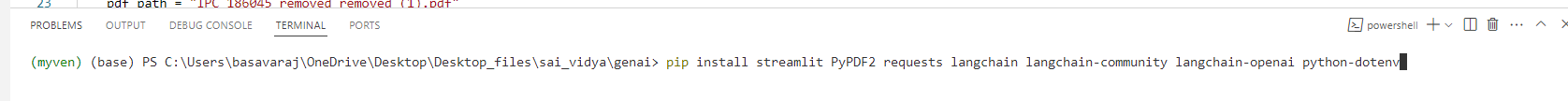
AI-generated content may be incorrect.

Click on new file name it , 9\_gen\_ai\_lab\_embedding.py

Install the required libraries

To install run the below command in terminal

🡪 pip install streamlit PyPDF2 requests langchain langchain-community langchain-openai python-dotenv



Paste the IPC\_186045\_removed\_removed (1).pdf file inside the same folder

Create 🡪 .env file

A screenshot of a computer

AI-generated content may be incorrect.

inside the .env file add your openai api key like below

OPENAI\_API\_KEY=your\_api\_key

Now type the program inside created , 9\_gen\_ai\_lab\_embedding.py file

Code:

import os

import requests

import PyPDF2

from dotenv import load\_dotenv

import streamlit as st

from langchain\_openai import ChatOpenAI, OpenAIEmbeddings

from langchain\_community.vectorstores import FAISS

from langchain.text\_splitter import CharacterTextSplitter

from langchain.chains import RetrievalQA

# Load environment variables from .env

load\_dotenv()

def extract\_text\_from\_pdf(pdf\_path):

    with open(pdf\_path, "rb") as file:

        reader = PyPDF2.PdfReader(file)

        text = "\n".join([page.extract\_text() for page in reader.pages if page.extract\_text()])

    return text

def download\_and\_extract\_ipc():

    pdf\_path = "IPC\_186045\_removed\_removed (1).pdf"

    if not os.path.exists(pdf\_path):

        url = "https://www.indiacode.nic.in/repealedfileopen?rfilename=A1860-45.pdf"

        response = requests.get(url, stream=True)

        if response.status\_code == 200:

            with open(pdf\_path, "wb") as file:

                for chunk in response.iter\_content(chunk\_size=1024):

                    file.write(chunk)

            st.success("IPC PDF downloaded successfully.")

        else:

            st.error("Failed to download the IPC PDF.")

            return None

    text = extract\_text\_from\_pdf(pdf\_path)

    if text.strip():

        with open("Indian\_Penal\_Code.txt", "w", encoding="utf-8") as f:

            f.write(text)

        return text

    else:

        st.error("No extractable text found in the PDF.")

        return None

@st.cache\_resource(show\_spinner="Embedding IPC text...")

def create\_retriever(text):

    text\_splitter = CharacterTextSplitter(chunk\_size=1000, chunk\_overlap=200)

    texts = text\_splitter.split\_text(text)

    embeddings = OpenAIEmbeddings()

    vector\_store = FAISS.from\_texts(texts, embeddings, normalize\_L2=True)

    return vector\_store.as\_retriever(search\_type="similarity", search\_kwargs={"k": 3})

def main():

    st.title("📘 IPC Chatbot")

    st.write("Ask anything about the Indian Penal Code (1860).")

    openai\_key = os.getenv("OPENAI\_API\_KEY")

    if not openai\_key:

        st.error("OpenAI API key not found. Please set it in your .env file.")

        return

    if "retriever" not in st.session\_state:

        text = download\_and\_extract\_ipc()

        if not text:

            return

        retriever = create\_retriever(text)

        st.session\_state.retriever = retriever

    llm = ChatOpenAI(model\_name="gpt-3.5-turbo")

    qa\_chain = RetrievalQA.from\_chain\_type(llm=llm, retriever=st.session\_state.retriever)

    user\_input = st.text\_input("Enter your query:")

    if user\_input:

        with st.spinner("Searching..."):

            response = qa\_chain.invoke(user\_input)

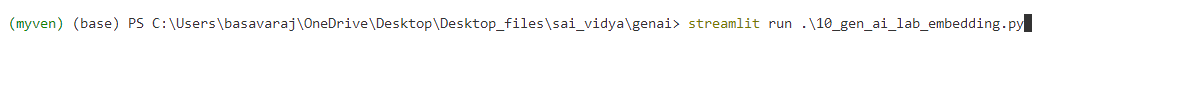
            st.markdown("\*\*Bot:\*\* " + response["result"])

if \_\_name\_\_ == "\_\_main\_\_":

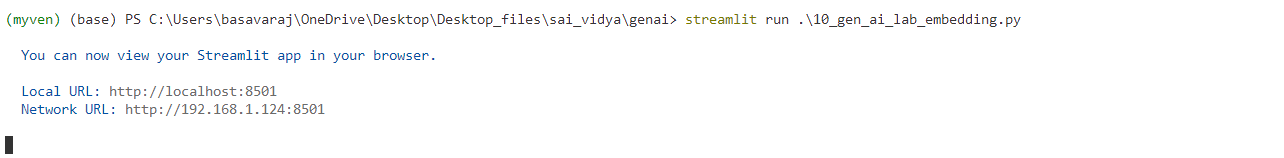
    main()

Run the program

Command 🡪 streamlit run 10\_gen\_ai\_lab\_embedding



Output



New stremlit page will open in browser that is IPC chatbot like below image

Ask chatbot about any ipc question

A screenshot of a computer

AI-generated content may be incorrect.