Guide: Running the Azure OpenAI Connection Test

This document outlines two methods to set up your environment and run the `test_connection.py` script.

Prerequisites:

- Ensure you have the `test_connection.py` file ready.
- You must have your Azure OpenAI credentials (API Key, Endpoint URL, Deployment Name, API Version).

Method 1: Virtual Environment (Recommended)

This method creates an isolated environment for your project, ensuring no conflicts with system-wide packages.

1. Verify and Install Python

Ensure Python 3.10 or greater is installed (Python 3.12 is recommended).

Check current version
python3 --version

# If not installed or version is < 3.10, install Python 3.12
sudo apt update
sudo apt install python3.12

2. Install Virtual Environment Tool

If you haven't already, install the `venv` module for your Python version.

Bash
# Replace python3.12 with your specific version if different
sudo apt install python3.12-venv

3. Create and Activate the Environment

Create a virtual environment named `venv` and activate it.

Create the environment
python3 -m venv venv

Activate the environment
source venv/bin/activate

Note: Once activated, your terminal prompt should show `(venv)`.

4. Install Dependencies via Pip

Install the OpenAI library specifically for this environment.

pip install openai

5. Update Configuration

Open `test_connection.py` in your text editor and update the configuration variables with your specific Azure details:

# Look for these variables in the script and update them:
OPENAI_BASE_URL= "https://your-resource-name.openai.azure.com/"
OPENAI_API_KEY= "your_actual_api_key_here"
OPENAI_API_VERSION= "2023-05-15" # Update to your specific version
LLM_MODEL= "gpt-4o-mini"   # Your deployment name

6. Run the Program

Run the script within the active virtual environment.

Bash
python3 test_connection.py

Expected Output

When the program runs successfully, you will see the following output in your terminal:

Connecting to Azure OpenAI at: https://your-resource.openai.azure.com/

✅ CONNECTION SUCCESSFUL!
Response: Pong! How can I assist you today?

To exit the virtual environment later, simply run the command `deactivate`.

Method 2: System-Wide Installation (Linux/Debian)

This method installs the OpenAI library globally on your system using the package manager.

1. Verify and Install Python

Ensure Python 3.10 or greater is installed (Python 3.12 is recommended).

```
# Check current version
python3 --version

# If not installed or version is < 3.10, install Python 3.12
sudo apt update
sudo apt install python3.12
```

2. Install Dependencies

Install the OpenAI library using the system package manager.

```
sudo apt install python3-openai
```

3. Update Configuration

Open `test_connection.py` and update the Azure configuration details (Endpoint, Key, Model, Version) as shown in Method 1.

4. Run the Program

Execute the script using Python 3.

```
python3 test_connection.py
```

Expected Output

When the program runs successfully, you will see the following output in your terminal:

Connecting to Azure OpenAI at: https://your-resource.openai.azure.com/

✅ CONNECTION SUCCESSFUL!
Response: Pong! How can I assist you today?

Appendix: Sample Program

Save the following code as **`test_connection.py`**.

```
import os
from openai import AzureOpenAI

# ========================================
# CONFIGURATION
# Update these values with your Azure OpenAI details
# ========================================
# --- STATIC VARIABLES ---
OPENAI_BASE_URL= "https://<your-resource-name>.openai.azure.com/"
OPENAI_API_KEY= "<your_actual_api_key_here>"
OPENAI_API_VERSION= "2023-05-15"  # Update to your specific version
LLM_MODEL= "gpt-4o-mini"      # Your deployment name
# ========================================

def test_azure_connection():
    print(f"Connecting to Azure OpenAI at: {AZURE_ENDPOINT}")

    try:
        # Initialize the Azure OpenAI Client
        client = AzureOpenAI(
            azure_endpoint=AZURE_ENDPOINT,
            api_key=API_KEY,
            api_version=API_VERSION
        )

        # Create a simple completion request to test the connection
        response = client.chat.completions.create(
            model=DEPLOYMENT_NAME,
            messages=[
```

```
            {"role": "system", "content": "You are a helpful assistant."},
            {"role": "user", "content": "Hello! Is this connection working?"}
        ]
    )

    # Print the response
    print("\n--- Connection Successful! ---")
    print("Response from Azure OpenAI:")
    print(response.choices[0].message.content)

except Exception as e:
    print("\n--- Connection Failed ---")
    print(f"Error: {e}")
    print("Please check your API Key, Endpoint, and Deployment Name.")

if __name__ == "__main__":
    test_azure_connection()
```

Troubleshooting

- **Module Not Found:** If you see `ModuleNotFoundError: No module named 'openai'`, ensure you activated the virtual environment (Method 2) or that the `apt install` completed successfully (Method 1).
- **Authentication Error:** If the script fails with an auth error, double-check that your **API Key** and **Azure Endpoint** are copied correctly without extra spaces.