

# OpenAI Master Class

- Make sure you have access to the following Azure resources within your tenant:
  - Azure Open AI (Have the necessary quotas available to spin up models).
  - Azure Databricks (Alternatively local anaconda or vs-code with python installed).
  - Azure DevOps.
  - Azure Cognitive Search.
  - Azure Redis for Cache.

## Module 1 – OpenAI and Generative AI

- Introduction to Azure OpenAI. Theoretical content contained within the slide deck.

## Module 2 – Azure OpenAI studio Playground

- Introduction to Azure OpenAI Studio Playground. Theoretical content contained within the slide deck.
- **Exercise 1 – Setup and login to Azure OpenAI studio:**
  - Setup Azure OpenAI studio.
- **Exercise 2 – Interface with the Azure OpenAI studio Completion Playground:**
  - Deploy and interface with the Azure Completion Playground.
- **Exercise 3 – Interface with Azure OpenAI studio Chat Playground:**
  - Deploy and interface with the Azure Chat Playground.
- **Exercise 4 – Interface with DALL.E Playground:**
  - Test the DALL.E playground and its functionality.

## Module 3 – Azure OpenAI integration with Databricks

- Introduction to Azure OpenAI and Databricks Integration. Theoretical content contained within the slide deck.
- **Exercise 1 – Setup Databricks Environment:**
  - Launch Databricks Workspace.
  - Create local library using the following settings:
    - maven coordinates: [com.microsoft.azure:synapseml\\_2.12:0.9.5](https://mmlspark.azureedge.net/maven)
    - Repository: <https://mmlspark.azureedge.net/maven>
  - Ensure you have the requirements.txt file in your home directory to install the relevant libraries when running the notebooks.
  - Setup .ini file containing the required key and base for the Azure OpenAI subscription.
  - Reference Module 3 notebook on the Git Repository to test the connection ([TNG\\_AzureOpenAI\\_MasteryCourse/MODULE 3 - NOTEBOOK SETUP.py at main · jeanpierrevanniekerk/TNG\\_AzureOpenAI\\_MasteryCourse \(github.com\)](https://github.com/jeanpierrevanniekerk/TNG_AzureOpenAI_MasteryCourse/blob/main/MODULE%203%20-%20NOTEBOOK%20SETUP.py)).

## Module 4 – APIs and SDKs

- Introduction to Azure OpenAI APIs and SDKs. Theoretical content contained within the slide deck.
- **Exercise 1 – Login to Azure CLI and then interact with an Azure OpenAI LLM:**
  - 1) Make sure you have the necessary permissions to access the Azure CLI
  - 2) `az login`
  - 3) `export accessToken=$(az account get-access-token --resource https://cognitiveservices.azure.com | jq -r .accessToken)`
  - 4) `curl https://tngpocazureopenai-services.openai.azure.com/openai/deployments/ChatGPT/completions?api-version=2022-12-01 \`  
`-H "Content-Type: application/json" \`  
`-H "Authorization: Bearer $accessToken" \`  
`-d '{ "prompt": "Tell me a funny story.", "max_tokens": 5 }'`
  - 5) `curl https://tngpocazureopenai-services.openai.azure.com/openai/deployments/ChatGPT/completions?api-version=2022-12-01 \`  
`-H "Content-Type: application/json" \`  
`-H "Authorization: Bearer $accessToken" \`  
`-d '{ "prompt": "Tell me a funny story.", "max_tokens": 500}'`
- **Exercise 2 – Access Azure OpenAI LLM functionality using Python SDK:**
  - 1) Access Module 4 – SDK notebook on the Git Repository  
([TNG AzureOpenAI MasteryCourse/MODULE 4 - PYTHON SDK.py at main · jeanpierrevanniekerk/TNG\\_AzureOpenAI\\_MasteryCourse \(github.com\)](#))

## Module 5 – Prompt Engineering

- Introduction to Azure OpenAI Prompt Engineering. Theoretical content contained within the slide deck.
- **Exercise 1 – Test various prompt engineering techniques:**
  - 1) Access Module 5 – Prompt Engineering Notebook on Git Repository  
([TNG AzureOpenAI MasteryCourse/MODULE 5 - PROMPT ENGINEERING.py at main · jeanpierrevanniekerk/TNG\\_AzureOpenAI\\_MasteryCourse \(github.com\)](#))

## Module 6 – Model Fine Tuning

- Introduction to Azure OpenAI Model Fine Tuning. Theoretical content contained within the slide deck.
- **Exercise 1 – Use general model which has not been fine-tuned using CLI:**  
`curl https://tngpocazureopenai-services.openai.azure.com/openai/deployments/ChatGPT/completions?api-version=2022-12-01 \`  
`-H "Content-Type: application/json" \`  
`-H "Authorization: Bearer $accessToken" \`  
`-d '{ "prompt": "When I go to the store, I want an", "max_tokens": 500}'`
- **Exercise 2 – Setup fine-tuned model using Azure OpenAI Python SDK:**
  - 1) Access Module 6 – Model Fine Tuning Notebook on Git Repository  
([TNG AzureOpenAI MasteryCourse/MODULE 6 - MODEL FINE TUNING.py at main · jeanpierrevanniekerk/TNG\\_AzureOpenAI\\_MasteryCourse \(github.com\)](#)). Setup model and train the model.

- **Exercise 3 – Use fine-tuned model using CLI:**

```
curl https://tngpocazureopenai-services.openai.azure.com/openai/deployments/ChatGPT/completions?api-version=2022-12-01 \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $accessToken" \
-d '{"prompt": "When I go to the store, I want an ", "max_tokens": 500}'
```

## Module 7 – Embedding Models

- Introduction to Azure OpenAI Embedding Models. Theoretical content contained within the slide deck.
- **Exercise 1 – Setup Redis for cache database and utilise Embedding models:**
  - 1) Access Module 7 – Embedding Models Notebook to write to Redis for cache database. Also run embedding models to embed text into vectors. Analyse and evaluate output ([TNG\\_AzureOpenAI\\_MasteryCourse/MODULE 7 - EMBEDDING MODELS.py at main · jeanpierrevanniekerk/TNG\\_AzureOpenAI\\_MasteryCourse \(github.com\)](#)).

## Module 8 – Codex Models

- Introduction to Azure OpenAI Codex Models. Theoretical content contained within the slide deck.
- **Exercise 1 – Run some code queries to test codex capabilities:**
  - 1) Access Module 8 – Codex Notebook to run through the various techniques of how the codex models can be used to improve coding productivity and performance ([TNG\\_AzureOpenAI\\_MasteryCourse/MODULE 8 - CODEX.py at main · jeanpierrevanniekerk/TNG\\_AzureOpenAI\\_MasteryCourse \(github.com\)](#)).

## Module 9 – DALL.E

- Introduction to Azure OpenAI DALL.E Models. Theoretical content contained within the slide deck.
- **Exercise 1 – Generate an image using the Azure CLI:**

```
curl https://tngpocazureopenai-services.openai.azure.com/openai/images/generations:submit?api-version=2023-06-01-preview \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $accessToken" \
-d '{"prompt": "An avocado chair", "size": "512x512", "n": 3, "response_format": "url"}'
```

- **Exercise 2 – Retrieve an image using the Azure CLI:**

```
curl -X GET "https://tngpocazureopenai-services.openai.azure.com/openai/operations/images/88ef2a2e-9a18-497b-988a-eeed86132dbb?api-version=2023-06-01-preview" -H "Content-Type: application/json" -H "Authorization: Bearer $accessToken"
```

- **Exercise 3 – Generate an image using the Azure Python SDKs:**
  - 1) Access Module 9 – DALL.E Notebook allowing the user to generate images using the Python SDK ([TNG\\_AzureOpenAI\\_MasteryCourse/MODULE 9 - DALL.E.py at main · jeanpierrevanniekerk/TNG\\_AzureOpenAI\\_MasteryCourse \(github.com\)](#)).

## **Module 10 – Grounding your model using your own data**

- Introduction to Azure OpenAI Grounding Models. Theoretical content contained within the slide deck.
- **Exercise 1 – Use the Azure OpenAI Studio to ground a model:**
  - 1) Generate a text file and copy some text in there.
  - 2) Upload it to the Azure OpenAI Studio during model grounding.
  - 3) Ask questions related to the text in the text file.
- **Exercise 2 – Use the Azure CLI to access and interface with the grounding model:**

```
curl -i -X POST https://tngpocazureopenai-services.openai.azure.com/openai/deployments/ChatGPT/extensions/chat/completions?api-version=2023-06-01-preview \
-H "Content-Type: application/json" \
-H "api-key: 7079b53b72df4f04bf94a302697561e9" \
-H "chatgpt_url: https://tngpocazureopenai-services.openai.azure.com/openai/deployments/ChatGPT/extensions/chat/completions?api-version=2023-06-01-preview" \
-H "chatgpt_key: 7079b53b72df4f04bf94a302697561e9" \
-d '{"dataSources": [{"type": "AzureCognitiveSearch", "parameters": {"endpoint": "https://tngcognitivesearch.search.windows.net/indexes/useyourrowndata/docs?api-version=2023-07-01-Preview&search=*&key\":\"n9ZqMO9M3zdLfplmh30FI9JFV2k8vhc0mTdhLFNRQfAzSeD9y1Ej\", \"indexName\": \"useyourrowndata\"}}}], \"messages\": [{\"role\": \"user\", \"content\": \"Is there a module that touches on Pandas code?\"}]}'
```

## **MODULE 11 – Pandas vs Pyspark with Azure OpenAI**

- Introduction to Azure OpenAI Pandas vs Pyspark. Theoretical content contained within the slide deck.
- **Exercise 1 – Identify how one would use Pandas and Pyspark to interface with the Azure OpenAI SDKs:**
  - 1) Access Module 11 – Pandas vs Pyspark Notebook to get a view of how one would leverage Pyspark to scale these LLM solutions ([TNG AzureOpenAI MasteryCourse/MODULE 11 - PANDAS vs PYSARK.py at main · jeanpierrevanniekerk/TNG AzureOpenAI MasteryCourse \(github.com\)](#)).

## **MODULE 12 – Azure OpenAI Practical Examples**

- Introduction to Azure OpenAI Practical Examples. Theoretical content contained within the slide deck.
- **Exercise 1 – Use the Azure OpenAI Example Notebooks to get a good understanding of some practical examples:**
  - 1) Access Module 12 – Example notebooks to get a view of how we would practically implement the Azure OpenAI LLMs:
    - a. Data Exploration and Embeddings.
      - i. [TNG AzureOpenAI MasteryCourse/MODULE 12 - DATA EXPLORATION.py at main · jeanpierrevanniekerk/TNG AzureOpenAI MasteryCourse \(github.com\)](#)
      - ii. [TNG AzureOpenAI MasteryCourse/MODULE 12 - EMBEDDINGS.py at main · jeanpierrevanniekerk/TNG AzureOpenAI MasteryCourse \(github.com\)](#)
    - b. Visualize Embeddings and Classification Documents.
      - i. [TNG AzureOpenAI MasteryCourse/MODULE 12 - VISUALIZE EMBEDDING.py at main · jeanpierrevanniekerk/TNG AzureOpenAI MasteryCourse \(github.com\)](#)
      - ii. [TNG AzureOpenAI MasteryCourse/MODULE 12 - CLASSIFICATION DOCUMENTS.py at main · jeanpierrevanniekerk/TNG AzureOpenAI MasteryCourse \(github.com\)](#)
    - c. Document Summarization.

- i. [TNG AzureOpenAI MasteryCourse/MODULE 12 - DOCUMENT SUMMARIZATION.py at main · jeanpierrevanniekerk/TNG AzureOpenAI MasteryCourse \(github.com\)](#)
- d. Key Information.
  - i. [TNG AzureOpenAI MasteryCourse/MODULE 12 - KEY INFORMATION.py at main · jeanpierrevanniekerk/TNG AzureOpenAI MasteryCourse \(github.com\)](#)
- e. Key Word Extraction.
  - i. [TNG AzureOpenAI MasteryCourse/MODULE 12 - KEY WORD EXTRACTION.py at main · jeanpierrevanniekerk/TNG AzureOpenAI MasteryCourse \(github.com\)](#)
- f. Semantic Search.
  - i. [TNG AzureOpenAI MasteryCourse/MODULE 12 - SEMANTIC SEARCH.py at main · jeanpierrevanniekerk/TNG AzureOpenAI MasteryCourse \(github.com\)](#)
- g. Information Retrieval.
  - i. [TNG AzureOpenAI MasteryCourse/MODULE 12 - INFORMATION RETRIEVAL.py at main · jeanpierrevanniekerk/TNG AzureOpenAI MasteryCourse \(github.com\)](#)

## **MODULE 13 – Azure OpenAI MLOps**

- Introduction to Azure OpenAI MLOps using Databricks and Azure DevOps. Theoretical content contained within the slide deck.
- **Exercise 1 – Take the “Data Exploration” notebook through the MLOps lifecycle:**
  - 1) Create 2 Databricks Environments (1 Dev and 1 Prod).
  - 2) Setup an Azure DevOps repository.
  - 3) Link the repository with your Dev Databricks Environment.
  - 4) Setup the Azure DevOps pipelines and releases.
  - 5) Push Dev notebook to Prod Databricks Environment.
  - 6) Setup scheduling and notification functionalities.

## **MODULE 14 – Advanced Use Cases**

- Getting some exposure to advances Azure OpenAI use cases. Theoretical content contained within the slide deck.

## **MODULE 15 – Summary and Conclusion**

- Summarizing the content covered in the Azure OpenAI Mastery course.
- Discussion potential next steps and how TNG can help expedite Azure OpenAI implementations in your organisation.