

Lab 3: Data Ingestion

This document describes how to use a Python notebook to populate the Azure Cosmos DB for MongoDB collection with sample data

Pre-requisites

Ensure that you have the following software installed on your system before proceeding with the lab:

- Visual Studio Code: A cross-platform code editor that supports Python development. You can download it from <https://code.visualstudio.com/>
- Python 3.10.11: The latest version of the Python programming language. You can download it from <https://www.python.org/downloads/release/python-31011/>

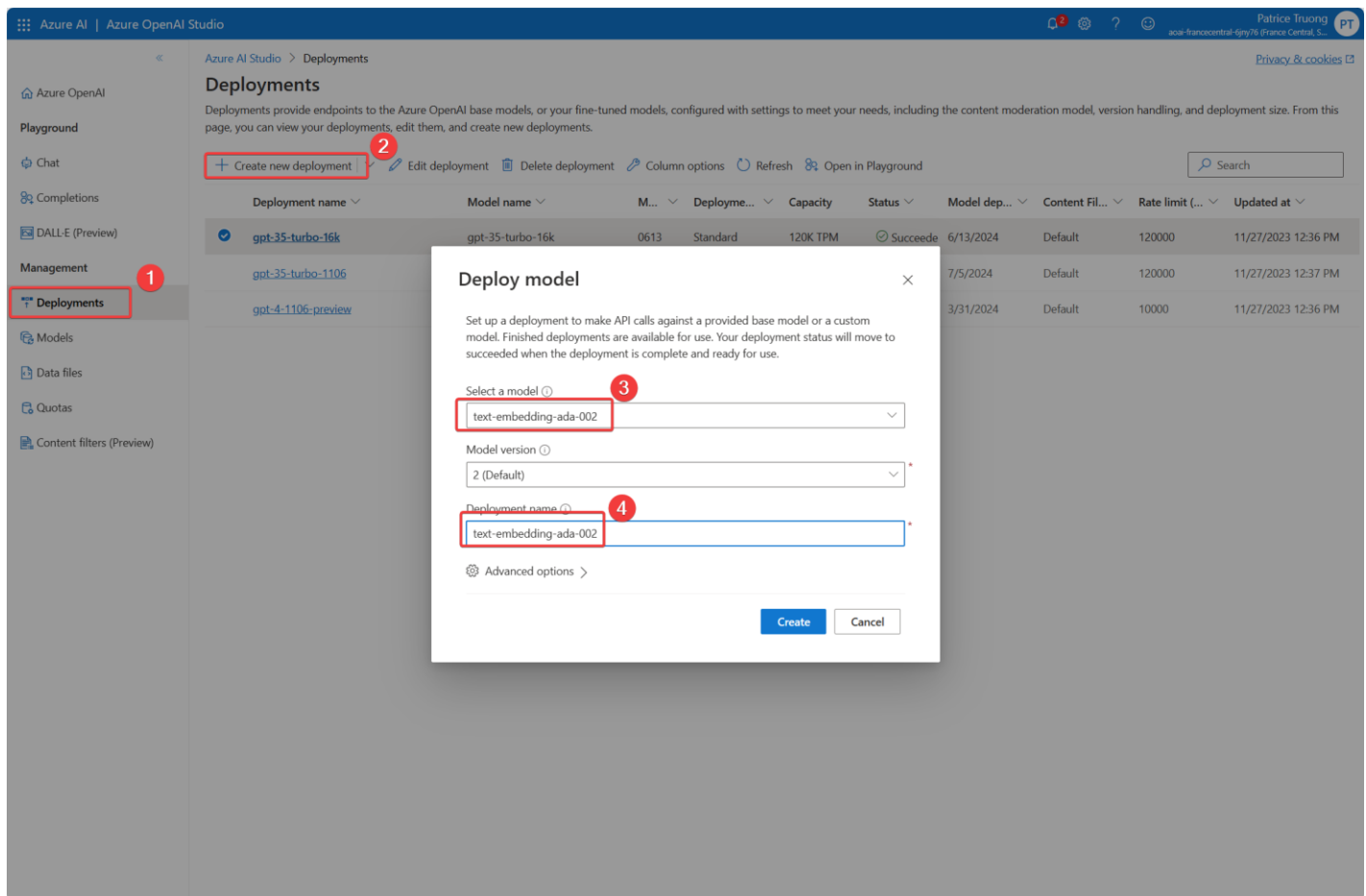
Note: If you are using a different version of Python, make sure that it is compatible with the libraries and packages used in this lab.

- Azure OpenAI account registered in the Azure subscription used for this lab

Deploy Azure OpenAI models

Follow these steps to deploy the Azure OpenAI models (GPT 3.5 Turbo and Text-embedding-ada-002)

- Login to the Azure Portal
- Connect to the Azure OpenAI account
- In the left menu, select “Model deployments”
- Click “Manage deployments”
- Select the “Deployments” section
- Click on the “Create deployment” button
- Select the “text-embedding-ada-002” model in the dropdown list
- In the deployment name, type text-embedding-ada-002
- Click on the “Create” button to deploy the model

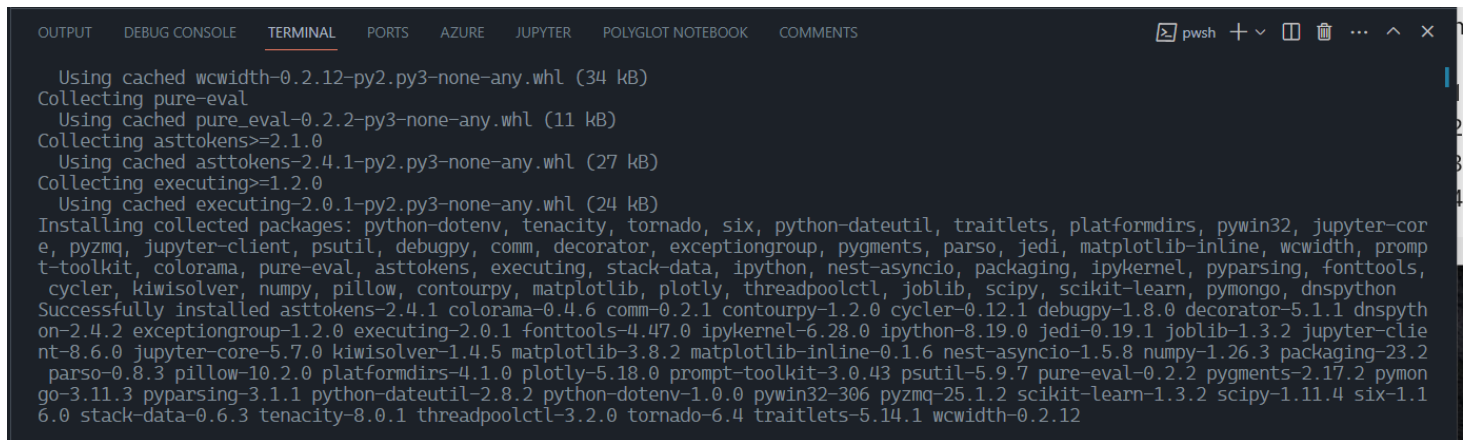


- Repeat the previous steps to deploy the gpt-3.5-turbo model

Prepare Python virtual environment

Run these commands to create a Python virtual environment and install the required libraries:

1. Create a “lab3” folder and navigate to this folder
2. Open Visual Studio Code
3. Using the menu: File > Open Folder > Code
4. Open a new Powershell Terminal > New Terminal
5. Type this command to create a virtual environment: `python -m venv .venv`
6. Activate the virtual environment with `.venv\scripts\activate`
7. Install the required libraries with `pip install -r requirements.txt`



```
Using cached wcwidth-0.2.12-py2.py3-none-any.whl (34 kB)
Collecting pure-eval
Using cached pure_eval-0.2.2-py3-none-any.whl (11 kB)
Collecting asttokens>=2.1.0
Using cached asttokens-2.4.1-py2.py3-none-any.whl (27 kB)
Collecting executing>=1.2.0
Using cached executing-2.0.1-py2.py3-none-any.whl (24 kB)
Installing collected packages: python-dotenv, tenacity, tornado, six, python-dateutil, traitlets, platformdirs, pywin32, jupyter-core, pyzmq, jupyter-client, psutil, debugpy, comm, decorator, exceptiongroup, pygments, parso, jedi, matplotlib-inline, wcwidth, prompt-toolkit, colorama, pure-eval, asttokens, executing, stack-data, ipython, nest-asyncio, packaging, ipykernel, pyparsing, fonttools, cycler, kiwisolver, numpy, pillow, contourpy, matplotlib, plotly, threadpoolctl, joblib, scipy, scikit-learn, pymongo, dnspython
Successfully installed asttokens-2.4.1 colorama-0.4.6 comm-0.2.1 contourpy-1.2.0 cycler-0.12.1 debugpy-1.8.0 decorator-5.1.1 dnspython-2.4.2 exceptiongroup-1.2.0 executing-2.0.1 fonttools-4.47.0 ipykernel-6.28.0 ipython-8.19.0 jedi-0.19.1 joblib-1.3.2 jupyter-client-8.6.0 jupyter-core-5.7.0 kiwisolver-1.4.5 matplotlib-3.8.2 matplotlib-inline-0.1.6 nest-asyncio-1.5.8 numpy-1.26.3 packaging-23.2 parso-0.8.3 pillow-10.2.0 platformdirs-4.1.0 plotly-5.18.0 prompt-toolkit-3.0.43 psutil-5.9.7 pure-eval-0.2.2 pygments-2.17.2 pymongo-3.11.3 pyparsing-3.1.1 python-dateutil-2.8.2 python-dotenv-1.0.0 pywin32-306 pyzmq-25.1.2 scikit-learn-1.3.2 scipy-1.11.4 six-1.16.0 stack-data-0.6.3 tenacity-8.0.1 threadpoolctl-3.2.0 tornado-6.4 traitlets-5.14.1 wcwidth-0.2.12
```

Modify environment variables

Rename the “.env template” file to “.env” and modify the variables to reflect your environment

COSMOSDB_MONGODB_HOST	The name of your Cosmos DB for MongoDB cluster	yourclustername.mongocluster.cosmos.azure.com
COSMOSDB_MONGODB_USERNAME	Cluster admin name	sa
COSMOSDB_MONGODB_PASSWORD	Cluster admin password	yourpassword
COSMOSDB_MONGODB_DATABASE	Database name	database_usern e.g. database_user01
COSMOSDB_MONGODB_PRODUCTS	Products collection name	products_username
COSMOSDB_MONGODB_CUSTOMERS	Customers collection name	customers_username
OPENAI_API_BASE	Azure OpenAI account url	<a href="https://<team_name>openai.openai.azure.com/">https://<team_name>openai.openai.azure.com/
OPENAI_API_KEY	Azure OpenAI account key	
OPENAI_EMBEDDING_MODEL	Name of your embedding model deployment	Defaults to text-embedding-ada-002
OPENAI_CHAT_MODEL	Name of your chat model deployment	Defaults to gpt-35-turbo

Your .env file should look like this:

```
COSMOSDB_MONGODB_HOST=yourclustername.mongocluster.cosmos.azure.com
COSMOSDB_MONGODB_USERNAME=sa
COSMOSDB_MONGODB_PASSWORD=your password
COSMOSDB_MONGODB_DATABASE=database_userXX
COSMOSDB_MONGODB_PRODUCTS=products_userXX
COSMOSDB_MONGODB_CUSTOMERS=customers_userXX

COSMOSDB_NOSQL_ACCOUNT=yournoqlaccount
COSMOSDB_NOSQL_DATABASE_NAME=database_teamXX
COSMOSDB_NOSQL_CONTAINER_NAME=conversations
COSMOSDB_NOSQL_KEY="xxxxxx"

AZURE_OPENAI_API_KEY=Xxxxxxx
AZURE_OPENAI_ENDPOINT=https://userXXopenai.openai.azure.com/
AZURE_OPENAI_EMBEDDING_MODEL=text-embedding-ada-002
AZURE_OPENAI_CHAT_MODEL=gpt-35-turbo
AZURE_OPENAI_API_VERSION=2023-12-01-preview

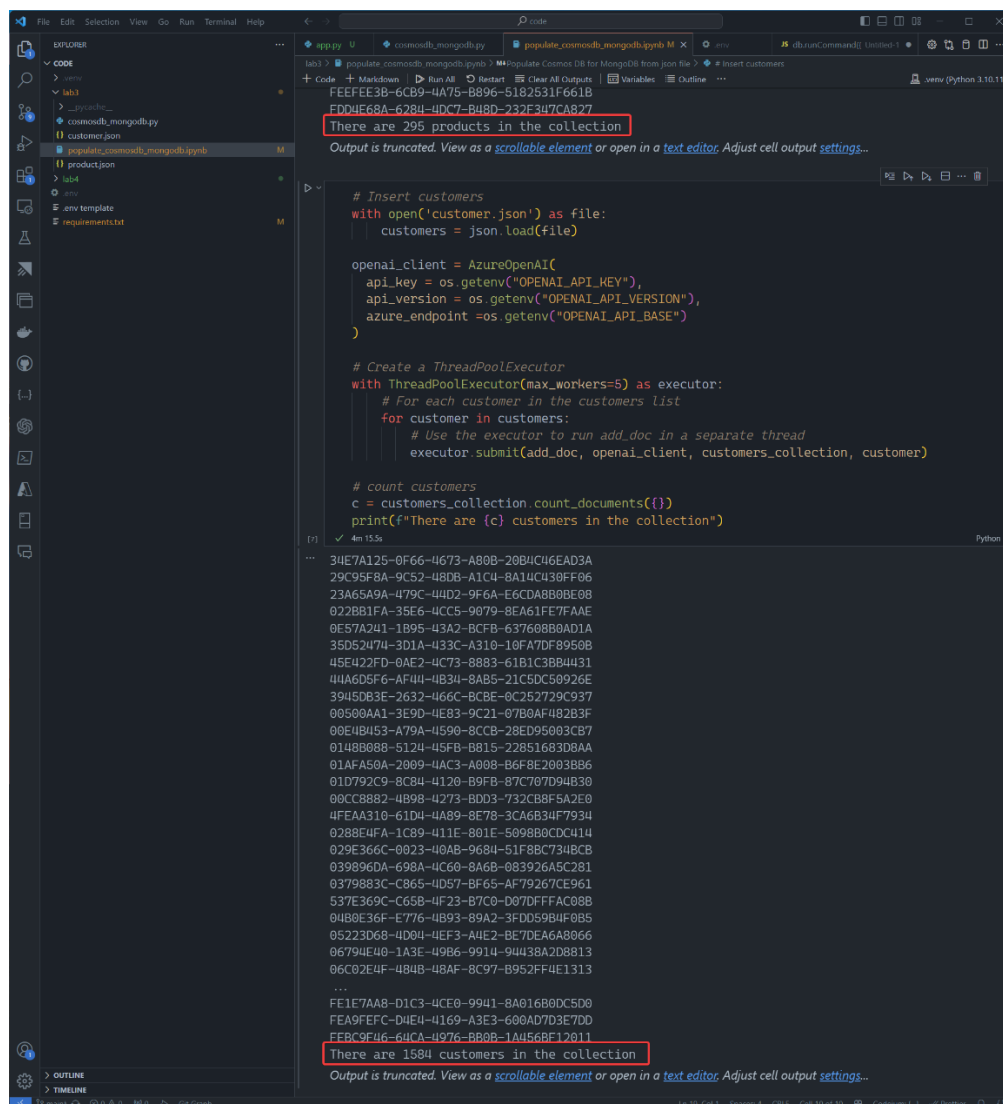
AZURE_SEARCH_SERVICE=https://yoursearch.search.windows.net
AZURE_SEARCH_KEY=xxxxxxxxxxxx
AZURE_SEARCH_API_VERSION=2023-11-01
```

Ingest data into Cosmos DB for MongoDB vCore

In this section, you will use the Python notebook to upload sample data to your Azure Cosmos DB for MongoDB vCore collection

1. You will find the customer.json and products.json in the lab3 folder on your local machine.
2. Open Visual Studio Code
3. Using the menu: File > Open Folder > Code
4. Open a new Powershell Terminal > New Terminal
5. Activate the virtual environment with `.venv\scripts\activate`
6. In the files navigation tree, open the file "populate_cosmosdb_mongodb.ipynb" it will open a jupyter notebook
7. Examine the jupyter notebook and the code in each cell
8. Execute each cell one by one or click the "Run all" button in the toolbar to run all cells

At the end of the process, there should be 295 products in the products collection and 1584 customers in the customers collection



```
lab3 > populate_cosmosdb_mongodb.ipynb > M> populate Cosmos DB for MongoDB from json file > # Insert customers
FEEFEE3B-6CB9-4A75-B896-5182531F661B
FDD4E68A-6284-4DC7-B48D-232F347CA827
There are 295 products in the collection
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

# Insert customers
with open('customer.json') as file:
    customers = json.load(file)

openai_client = AzureOpenAI(
    api_key = os.getenv("OPENAI_API_KEY"),
    api_version = os.getenv("OPENAI_API_VERSION"),
    azure_endpoint = os.getenv("OPENAI_API_BASE")
)

# Create a ThreadPoolExecutor
with ThreadPoolExecutor(max_workers=5) as executor:
    # For each customer in the customers list
    for customer in customers:
        # Use the executor to run add_doc in a separate thread
        executor.submit(add_doc, openai_client, customers_collection, customer)

# count customers
c = customers_collection.count_documents({})
print(f"There are {c} customers in the collection")

34E7A125-0F66-4673-A80B-20B4C6EAD3A
29C95F8A-9C52-48DB-A1C4-8A14C438FF86
23A65A9A-479C-44D2-9F6A-E6CDAB8B0E08
022B81FA-35E6-4CC5-9079-8EA61FE7FAAE
0E57A241-1B95-43A2-BCFB-637608B8AD1A
35052474-3D1A-433C-A310-10FA70F89508
45E422FD-0AE2-4C73-8883-61B1C3BB4431
44A6D5F6-AF44-4B34-8AB5-21C5DC50926E
3945DB3E-2632-466C-BCBE-0C252729C937
00500AA1-3E9D-4E83-9C21-07B0AF482B3F
00E4B453-A79A-4590-8CCB-28ED95003CB7
0148B088-5124-45FB-B815-22851683D8AA
01AFA50A-2009-4AC3-A008-B6F8E2003BB6
01D792C9-8C84-4120-B9FB-87C707D94B30
00CC8882-4B98-4273-BDD3-732CB8F5A2E0
4FEAA310-61D4-4A89-8E78-3CA6B34F7934
0288E4FA-1C89-411E-801E-5098B80CD044
029E366C-0023-40A8-9684-51F8BC734BCB
039896DA-698A-4C60-8A68-083D26A5C281
0379883C-C865-4DE7-BF65-AF7D267CE961
537E369C-C65B-4F23-B7C0-D07DFFAC08B
04B8E36F-E776-4B93-89A2-3FDD59B4F0B5
05223D68-4D04-4EF3-A4E2-BE7DEA6A8066
06794E40-1A3E-49B6-9914-94438A2D8813
06C02E4F-4B4B-4BAF-8C97-B952FF4E1313
...
FE1E7AAB-D1C3-4CE0-9941-8A016B0DC5D0
FEA9FEC-D4E4-4169-A3E3-600AD7D3E7D0
FFBC9F46-64CA-4976-BB08-1A456BF12011
There are 1584 customers in the collection
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

Check ingested data

To connect to the Azure Portal and view data from Azure Cosmos DB for Mongo vCore collection **products_team1**, follow these steps:

- Open a web browser and go to <https://portal.azure.com>
- Sign in with your Azure account credentials
- In the search box, type 'Azure Cosmos DB for MongoDB (vCore)' and select it from the results
- In the Azure Cosmos DB page, select your account name **cosmos-mongo-vcore-2024** from the list
- Click on 'Quick Start' in the menu
- Click on the 'Open MongoDB (vCore) shell' button

Enter Azure Cosmos DB for MongoDB vCore admin password: *****

```
/* switch to database_team01 */
```

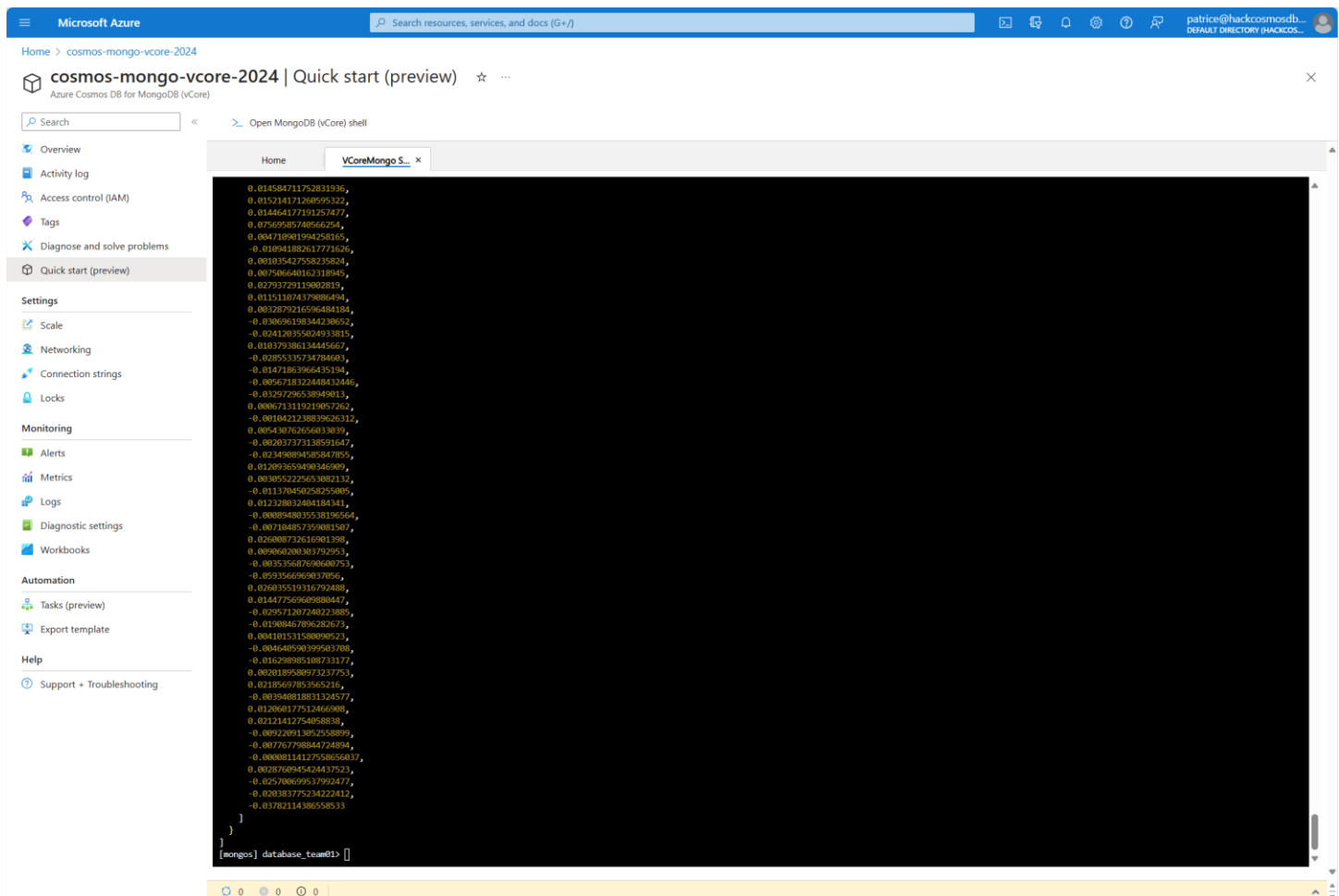
```
use database_user01
```

```
/* count documents in collection products_team1 */
```

```
db.products_user01.countDocuments()
```

```
/* display first document */
```

```
db.products_user01.find({}).limit(1)
```



The screenshot shows the Azure Portal interface for the 'cosmos-mongo-vcore-2024' account. The 'Quick start (preview)' tab is active, and the 'Open MongoDB (vCore) shell' button has been clicked. The shell interface is displayed, showing a list of documents from the 'products_team1' collection. The documents are displayed as a list of objects, each with a '_id' field containing a long hexadecimal string. The shell prompt is '[mongos] database_team01> '.

Troubleshooting

1. Consider that the cosmosuser used in the Mongo Shell may differ from the account that has been specified at cluster creation . For what follows you need to remember the credentials identified at cluster creation
2. If you encounter a “Error: Server record does not share hostname with parent URI” you should verify that:
 - a. The password has been escaped with special characters
 - b. The connection url is correct
 - c. The firewall settings allow the connection from the local machine
 - d. If point a-c are already verified you should check the DNS –
 - i. <https://learn.microsoft.com/en-us/azure/cosmos-db/mongodb/vcore/troubleshoot-common-issues#unable-to-connect-to-azure-cosmos-db-for-mongodb-vcore---timeout-error>
 - ii. run `npx resolve-mongodb-srv <url>` where <url> is the FQDN of the service
 - iii. run `nslookup -q=SRV _mongodb._tcp. <url>` where <url> is the FQDN of the service
 - e. If you still have issues then consider using a jumpbox machine deployed on Azure already using the Mongo stack like the Bitnami and connecto using the Azure Cosmos DB Mongo for vCore from there