Task-3

(Data Backup and Restore with Azure Blob Storage)

Aim: Implement a basic data backup and restore system using Azure Blob Storage. Write a Python or Node.js script to upload files from your local machine to Azure Blob Storage. Then, create a mechanism to download and restore these files back to your local machine. This project will give you hands-on experience with cloud-based storage services and data management.

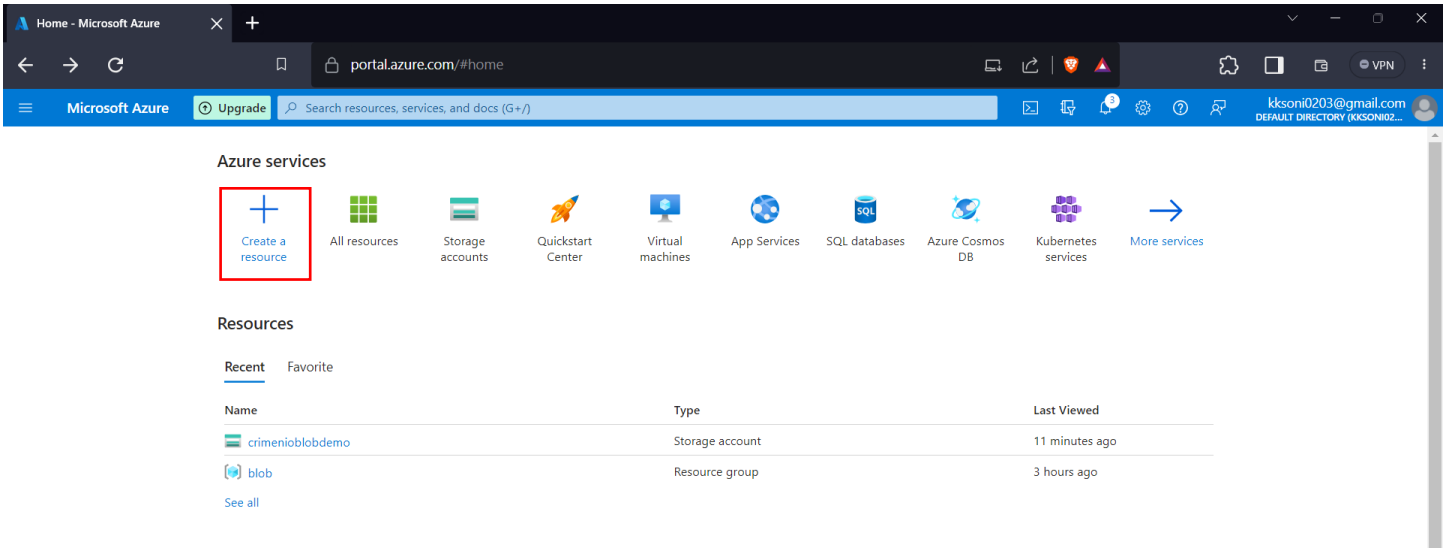
Step 1: Sign in to Azure Portal

1) Open a web browser and go to the Azure Portal: https://portal.azure.com/
2) Sign in with your Azure account.

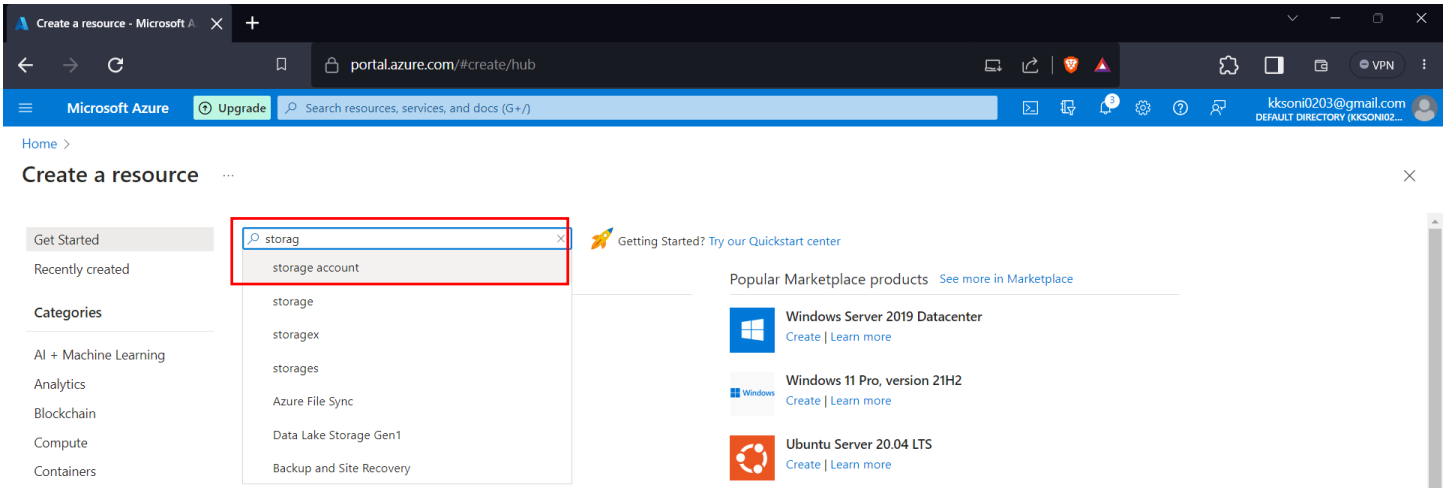
Step 2: Create a Resource Group

A resource group is a logical container for resources deployed in Azure.

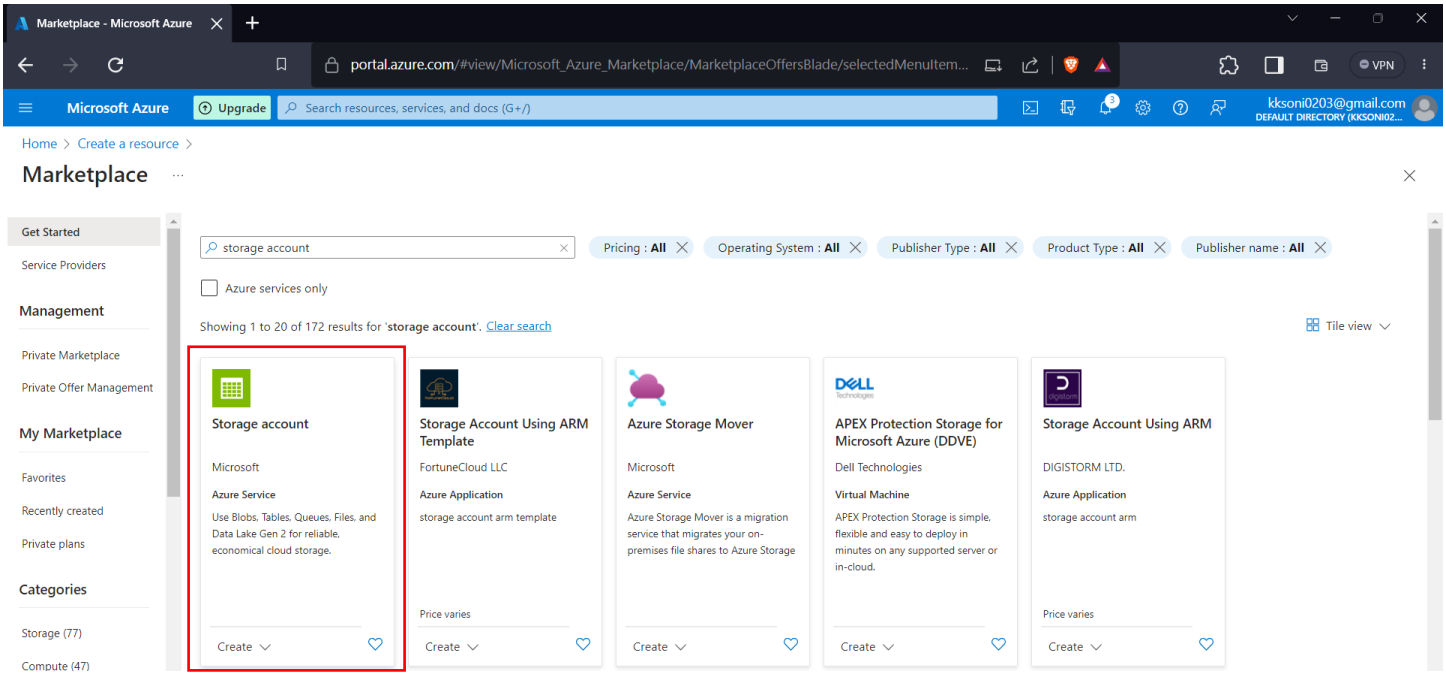
1) In the Azure Portal, click on the "Create a resource" button (+) on the left side.

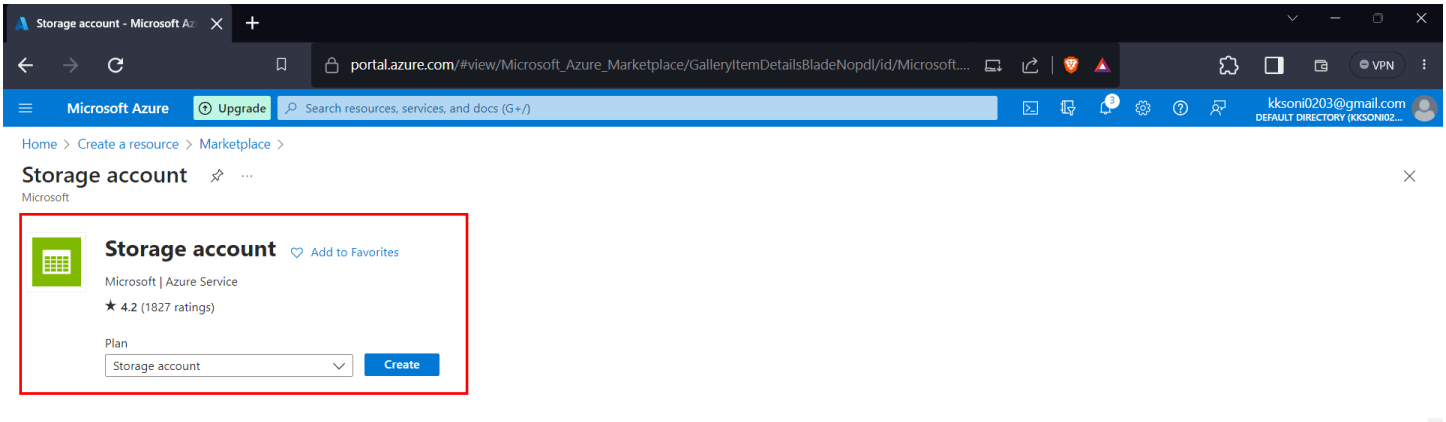


2) Search for "Storage account" and select "Storage account - blob, file, table, queue."

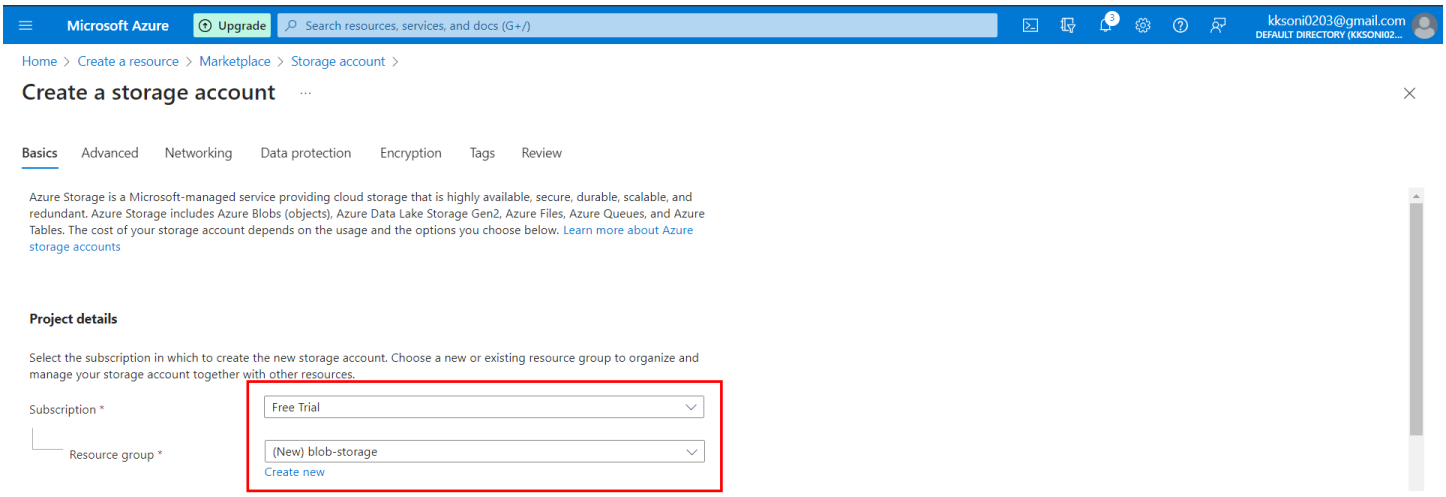


3) Select “Storge account” and click the "Create" button to create a new resource group.

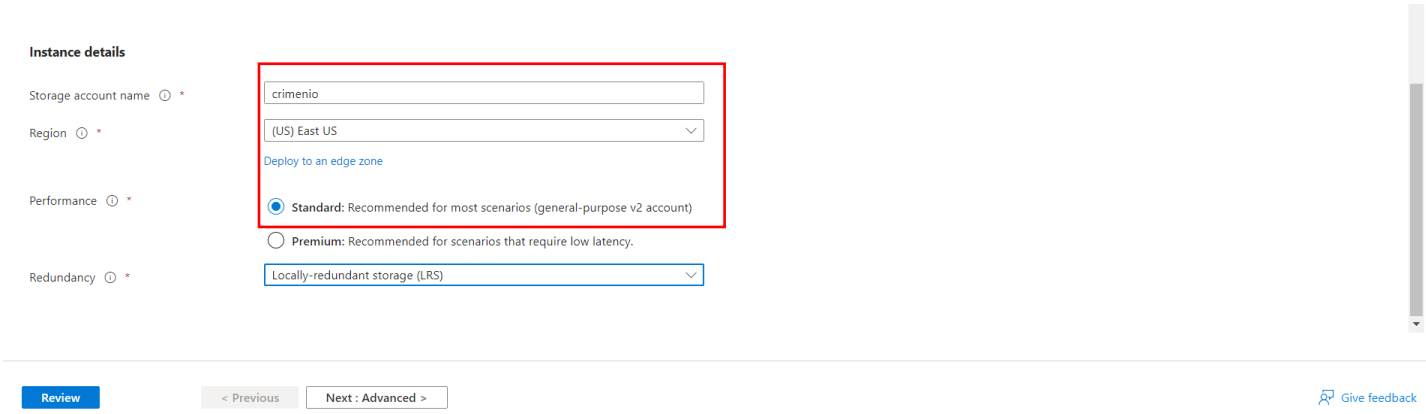




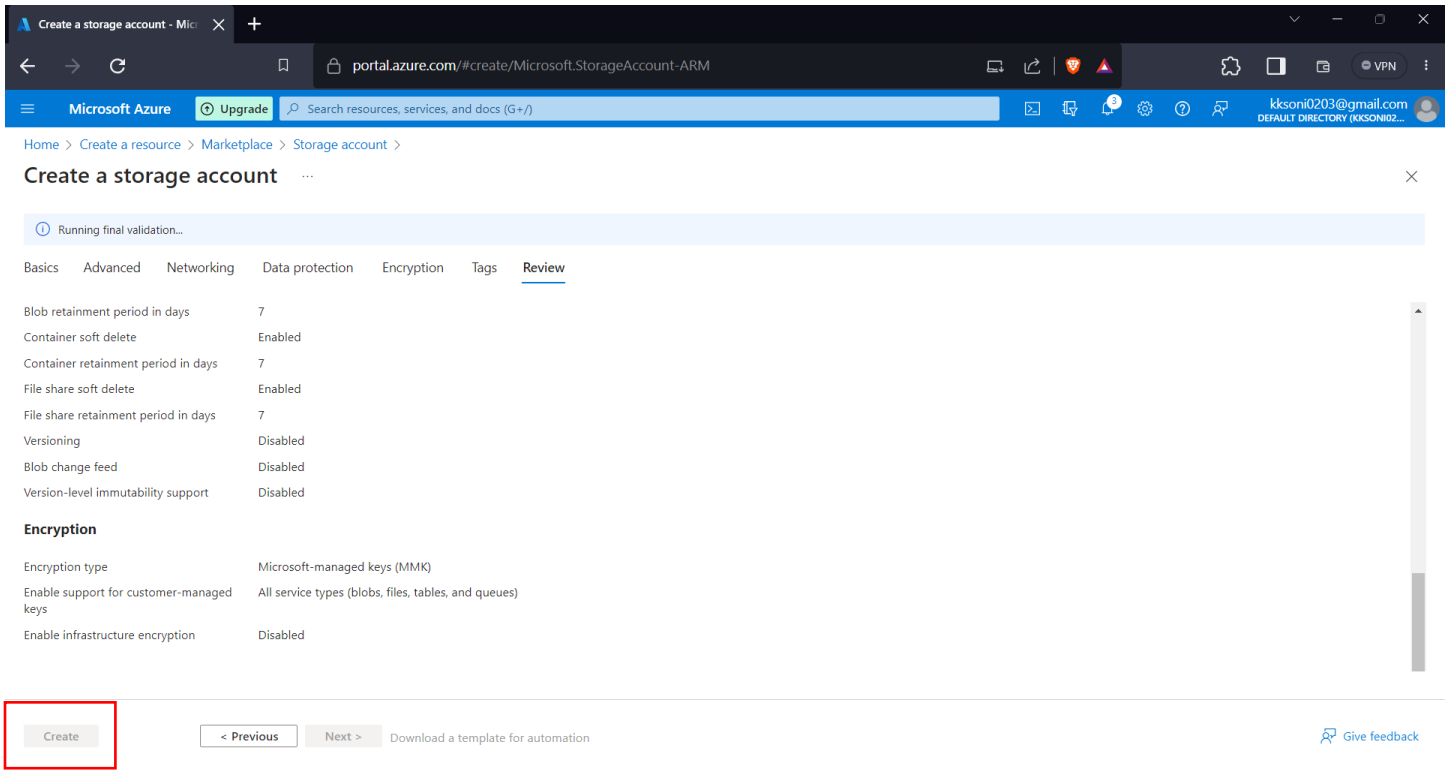
4) Provide a unique name for the resource group and choose a subscription.



5) Provide storage account name, select a region, select standard as performance and select LRS.

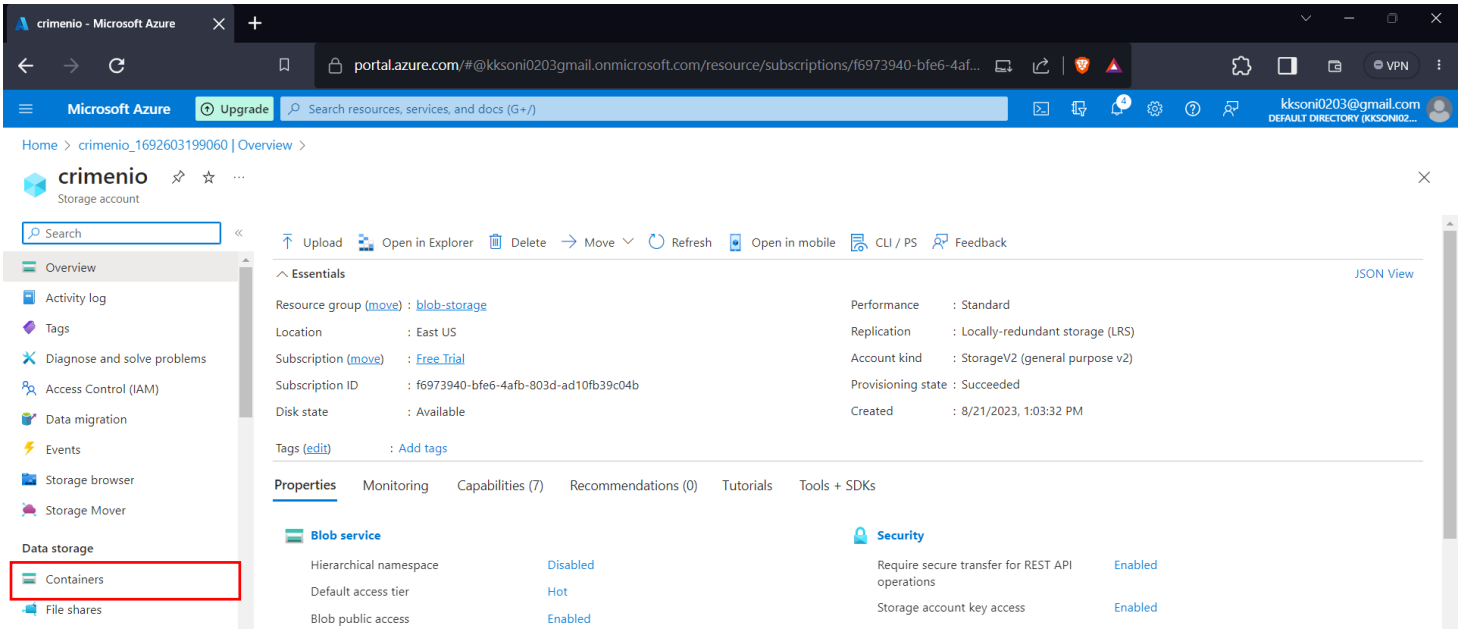


6) Review the settings and then click “Create”.

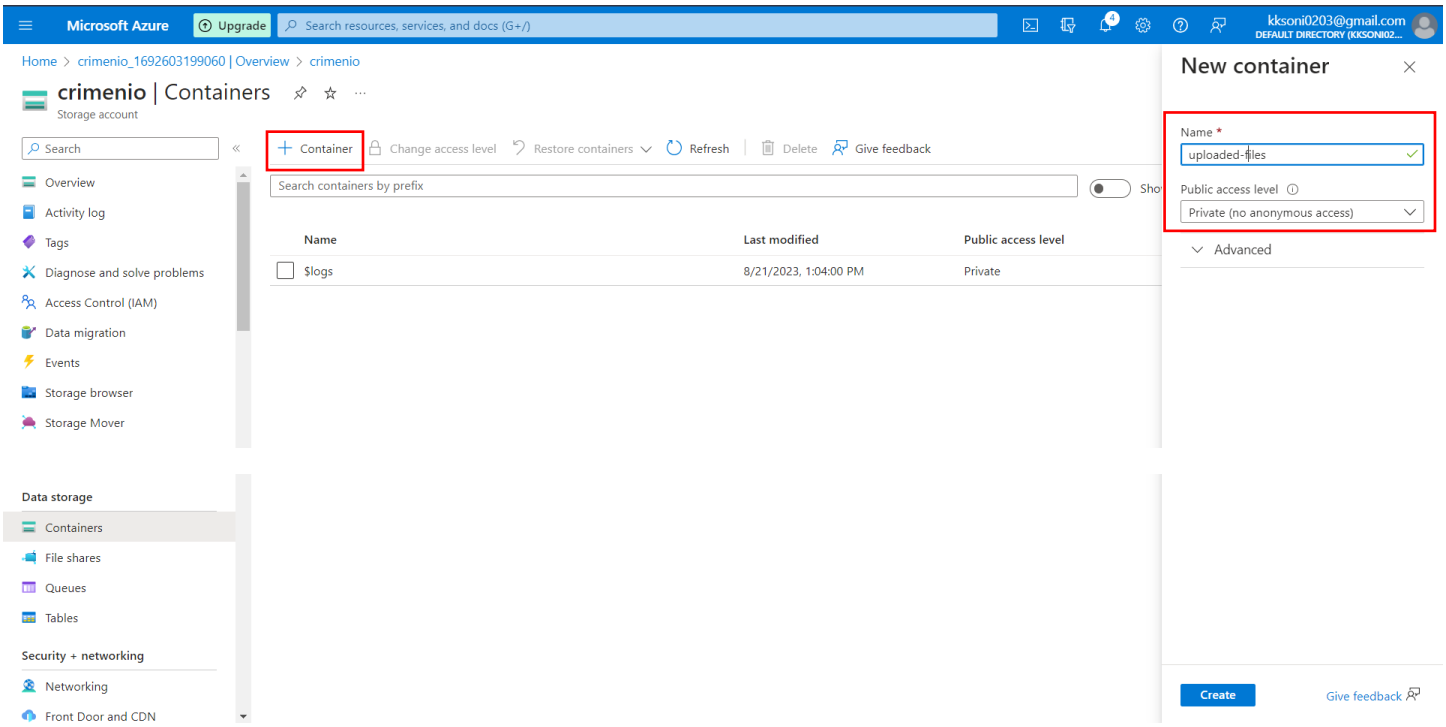


Step 3: Create a Container within the Storage Account

1) Once your storage account is created, go to the resource group containing your storage account.
2) In the storage account overview page, navigate to the "Containers" section in the left menu.



3) Click the "+ Container" button to create a new container and choose a public access level for the container. For private access, select "Private" (recommended). Click the "Create" button.



Step 2: Install Required Libraries

Ensure you have the azure-storage-blob package installed. You can install it using the following command:

```
# pip install azure-storage-blob
```

Step 3: Write the Upload Script

Create a Python script, for example, upload.py, and write the following code:

```
from azure.storage.blob import BlobServiceClient
import os
# import data

storage_account_key =
"tOcwQcQoth/eLzOnTNuxRdXzMRLSR+wVv2oDk3SXn6wMl6L/hRJ/3ooq6o33EOHwrH4sdPqtaZjt+ASt4tUwlg=="
storage_account_name = "crimenioblobdemo"
connection_string =
"DefaultEndpointsProtocol=https;AccountName=crimenioblobdemo;AccountKey=tOcwQcQoth/eLzOnTNuxRdXzMRLSR+wVv2oDk3SXn6wMl6L/hRJ/3ooq6o33EOHwrH4sdPqtaZjt+ASt4tUwlg==;EndpointSuffix=core.windows.net"
container_name = "backup"

def uploadToBlobStorage(file_path, file_name):
    blob_service_client = BlobServiceClient.from_connection_string(connection_string)
    blob_client =blob_service_client.get_blob_client(container=container_name,blob=file_name)

    with open(file_path,"rb") as data:
        blob_client.upload_blob(data)
        print("Upload "+file_name+" file")

uploadToBlobStorage('C:\\Users\\Krishnakant\\Desktop\\Task\\TAsk.jpg','Task')
```

Note: Replace "your_connection_string" with the actual connection string you obtained in Step 1. Also, set the local_directory to the path of the directory containing files you want to upload.

Step 4: Write the Download Script

Create another Python script, for example, download.py, and write the following code:

```
from azure.storage.blob import BlobServiceClient
import os

storage_account_key                                                                                          =
"tOcwQcQoth/eLzOnTNuxRdXzMRLSR+wVv2oDk3SXn6wMl6L/hRJ/3ooq6o33EOHwrH4sdPqtaZjt+ASt4tUwlg=="
storage_account_name = "crimenioblobdemo"
connection_string                                                                                            =
"DefaultEndpointsProtocol=https;AccountName=crimenioblobdemo;AccountKey=tOcwQcQoth/eLzOnTNuxR
dXzMRLSR+wVv2oDk3SXn6wMl6L/hRJ/3ooq6o33EOHwrH4sdPqtaZjt+ASt4tUwlg==;EndpointSuffix=core.windo
ws.net"
container_name = "backup"

download_directory = "C:\\Users\\Krishnakant\\Desktop\\Blob"

def download_files():
    blob_service_client = BlobServiceClient.from_connection_string(connection_string)
    container_client = blob_service_client.get_container_client(container_name)

    # Ensure the download directory exists
    os.makedirs(download_directory, exist_ok=True)

    # Download and save each blob to the local directory
    for blob in container_client.list_blobs():
        blob_client = container_client.get_blob_client(blob)
        blob_data = blob_client.download_blob()
        blob_file_path = os.path.join(download_directory, blob.name)

        with open(blob_file_path, "wb") as blob_file:
            blob_file.write(blob_data.readall())

        print(f"Downloaded: {blob.name}")

if __name__ == "__main__":
    download_files()
    print("Download process completed.")
```

Note: Replace "your_connection_string" with the actual connection string you obtained in Step 1. Also, set the download_directory to the path where you want to save the downloaded files.

Step 5: Run the Scripts

Open a terminal or command prompt and navigate to the directory where your scripts are located. Run the following commands:

For uploading:

python upload.py

For downloading:

python download.py

Note: These scripts will upload the files from your local directory to Azure Blob Storage and then download and restore them back to the specified local directory.

Remember to adapt the paths and filenames according to your preferences