Azure Storage Account Workshop - 45 Minutes

Workshop Overview

Duration: 45 minutes
Skill Level: Beginner
Cost: Minimal (less than $1 with free tier)

Prerequisites

- Active Azure subscription
- Web browser
- Basic understanding of cloud storage concepts

Lab Architecture

Part 1: Create Storage Account (5 minutes)

Step 1.1: Navigate to Storage Accounts

1. Sign in to **Azure Portal** (portal.azure.com)
2. Click **Create a resource** (+ icon)
3. Search for **Storage account**
4. Click **Create**

Step 1.2: Configure Basics

1. **Subscription:** Select your subscription
2. **Resource Group:** Click **Create new**
   - Name: rg-storage-lab
   - Click **OK**
3. **Storage account name:** stglab + your initials + random numbers
   - Example: stglabjs12345
   - Must be 3-24 characters, lowercase letters and numbers only
   - Must be globally unique
4. **Region:** Select **East US** (or closest to you)
5. **Performance: Standard**
6. **Redundancy: -redundant storage (LRS)**

Step 1.3: Advanced Settings

1. Click **Next: Advanced**
2. **Security:**

    - Allow enabling public access on containers: **Enabled** (checked)
    - Enable storage account key access: **Enabled**
3. **Blob storage:**
    - Allow blob anonymous access: **Enabled**
4. **Hierarchical namespace:** Disabled
5. Click **Next: Networking**

Step 1.4: Networking

1. **Network access:** Enable public access from all networks
2. Click **Next: Data protection**
3. Keep defaults, click **Next: Encryption**
4. Keep defaults, click **Next: Tags**
5. Skip tags, click **Review + create**

Step 1.5: Create

1. Review all settings
2. Click **Create**
3. Wait 30-60 seconds for deployment
4. Click **Go to resource**

Part 2: Work with Blob Storage (15 minutes)

Step 2.1: Create Containers

Container 1 - Public Images:

1. In left menu, click **Containers** (under Data storage)
2. Click **+ Container**
3. **Name:** images
4. **Public access level: Blob (anonymous read access for blobs only)**
5. Click **Create**

Container 2 - Private Backups: 6. Click **+ Container** again 7. **Name:** backups 8. **Public access level: Private (no anonymous access)** 9. Click **Create**

Container 3 - Archive Storage: 10. Click **+ Container** 11. **Name:** archives 12. **Public access level: Private** 13. Click **Create**

Step 2.2: Upload Files to Images Container

Create a sample HTML file:

1. Open Notepad or any text editor on your computer
2. Paste this content:

```
<!DOCTYPE html>

<html>

<head><title>Test Page</title></head>

<body>

<h1>Hello from Azure Blob Storage!</h1>

<p>This file is publicly accessible.</p>

</body>

</html>
```

3. Save as index.html on your desktop

Upload to Azure: 4. In Azure Portal, click the **images** container 5. Click **Upload** 6. Click **Browse for files** 7. Select the index.html file you created 8. **Advanced settings:**

- Blob type: **Block blob**
- Block size: Keep default
- Access tier: **Hot**

9. Click **Upload**
10. Wait for upload to complete

Step 2.3: Access Public Blob

1. Click on the uploaded **index.html** file
2. In the properties, find and copy the **URL**
    - Example: https://stglabjs12345.blob.core.windows.net/images/index.html
3. Open a new browser tab
4. Paste the URL and press Enter
5. You should see your HTML page displayed
6. **Success!** This demonstrates public blob access

Step 2.4: Upload to Private Container

Create a backup file:

1. Create a new text file called backup-data.txt
2. Add some content: "This is confidential backup data"
3. Save it

Upload: 4. Go back to Azure Portal 5. Click **Containers**, then click **backups** container 6. Click **Upload** 7. Select backup-data.txt 8. Click **Upload**

Test private access: 9. Click on **backup-data.txt** 10. Copy the URL 11. Open in new browser tab 12. You should see: **"Public access is not permitted"** or authentication error 13. ✅ **Success!** This demonstrates private storage security

Step 2.5: Change Access Tier (Cost Optimization)

1. Go to **archives** container
2. Upload any file (reuse backup-data.txt)
3. After upload, click on the file
4. Click **Change tier** button at top
5. **Access tier:** Select **Cool**
6. Click **Save**
7. Notice: Cool tier costs less for storage, more for access

Understanding Tiers:

- **Hot:** Frequent access, higher storage cost, lower access cost
- **Cool:** Infrequent access (30+ days), lower storage cost
- **Archive:** Rare access (180+ days), lowest storage cost, hours to retrieve

---

Part 3: Configure File Share (10 minutes)

Step 3.1: Create File Share

1. In your storage account left menu, click **File shares**
2. Click **+ File share**
3. **Name:** shared-documents
4. **Tier: Transaction optimized**
5. **Provisioned capacity:** 5 GiB (minimum)
6. Click **Create**

Step 3.2: Upload Files to File Share

1. Click on **shared-documents** file share
2. Click **Upload**
3. Click **Browse for files**
4. Select one or more files from your computer
5. Click **Upload**

6. Files are now in shared storage

Step 3.3: Create Directory Structure

1. In the file share, click **+ Add directory**
2. **Name:** projects
3. Click **OK**
4. Click on **projects** folder
5. Click **+ Add directory**
6. **Name:** 2024
7. Click **OK**
8. You now have nested directories: /projects/2024/

Step 3.4: Get Connection Information

1. Go back to **shared-documents** overview
2. Click **Connect** button at top
3. You'll see connection scripts for:
   - **Windows:** PowerShell script using net use
   - **Linux:** Bash script using mount
   - **macOS:** Bash script
4. **Note:** These scripts allow mounting the share as a network drive

Example connection string format:

\\<storage-account>.file.core.windows.net\shared-documents

Part 4: Work with Table Storage (8 minutes)

Step 4.1: Access Tables

1. In storage account left menu, scroll down to **Tables**
2. Click **Tables**
3. Click **+ Table**
4. **Name:** products
5. Click **OK**

Step 4.2: Add Data Using Storage Browser

1. Click on **products** table
2. Click **Storage Browser** in left menu (under Data storage)

3. Navigate to **Tables → products**
4. Click **Add entity**

Entity 1 - Laptop: 5. **PartitionKey:** electronics 6. **RowKey:** 001 7. Click **Add property**

- Property name: ProductName
- Type: String
- Value: Laptop Pro 15

8. Click **Add property**
    - Property name: Price
    - Type: Double
    - Value: 1299.99
9. Click **Add property**
    - Property name: InStock
    - Type: Boolean
    - Value: true
10. Click **Insert**

Entity 2 - Mouse: 11. Click **Add entity** again 12. **PartitionKey:** electronics 13. **RowKey:** 002 14. Add properties:

- ProductName (String): Wireless Mouse
- Price (Double): 29.99
- InStock (Boolean): true

15. Click **Insert**

Entity 3 - Notebook: 16. Add one more entity: 17. **PartitionKey:** office-supplies 18. **RowKey:** 001 19. Add properties:

- ProductName (String): Notebook A4
- Price (Double): 5.99
- InStock (Boolean): false

20. Click **Insert**

Step 4.3: Query Table Data

1. In the table view, you'll see all three entities
2. Notice they're grouped by PartitionKey
3. **PartitionKey** = Category grouping (for performance)
4. **RowKey** = Unique identifier within partition

5. ✅ **Success!** You've created a NoSQL database

Part 5: Security and Access Control (5 minutes)

Step 5.1: Generate Shared Access Signature (SAS)

1. Go to your storage account overview
2. In left menu, click **Shared access signature** (under Security + networking)
3. **Allowed services:** Check **Blob** only
4. **Allowed resource types:** Check all (Service, Container, Object)
5. **Allowed permissions:** Check **Read** and **List** only
6. **Start time:** Keep current date/time
7. **End time:** Set to 1 hour from now
8. **Allowed protocols:** HTTPS only
9. Click **Generate SAS and connection string**
10. Copy the **SAS token** (starts with ?sv=...)
11. Copy the **Blob service SAS URL**

Step 5.2: Test SAS Token Access

1. Take the URL from your **images/index.html** file
2. Add the SAS token to the end:
    - Original: https://stglabjs12345.blob.core.windows.net/images/index.html
    - With SAS: https://stglabjs12345.blob.core.windows.net/images/index.html?sv=... (paste full token)
3. This URL now provides temporary, limited access

Step 5.3: View Access Keys

1. In left menu, click **Access keys**
2. You'll see **key1** and **key2**
3. Click **Show** next to key1
4. These keys provide **full access** to the storage account
5. **Best practice:** Use SAS tokens for limited access instead of sharing keys

Step 5.4: Configure Firewall (Optional)

1. In left menu, click **Networking**
2. Under **Firewalls and virtual networks:**

3. You can restrict access to:
   - Specific IP addresses
   - Virtual networks
   - Azure services
4. **For this lab:** Keep "Public network access: Enabled from all networks"

Part 6: Monitoring and Management (2 minutes)

Step 6.1: View Metrics

1. In left menu, click **Metrics** (under Monitoring)
2. Click **Metric:** Select **Blob Capacity**
3. View storage usage over time
4. Click **Add metric**
5. Select **Transactions** to see activity

Step 6.2: Check Storage Usage

1. In left menu, click **Overview**
2. Under **Essentials**, note:
   - **Used capacity:** Shows how much storage you're using
   - **Performance tier:** Standard
   - **Replication:** LRS

Step 6.3: Access Activity Logs

1. In left menu, click **Activity log**
2. View all operations performed:
   - Container creation
   - File uploads
   - Configuration changes
3. Useful for auditing and troubleshooting

Part 7: Practical Exercise - Complete Scenario (Optional +5 min)

Scenario: Company File Management System

Task: Create a complete file organization system

1. **Create structure in your images container:**

    - Upload 3 different files (any files)
    - Organize them logically
2. **Create a backup:**
    - Download one file from images
    - Upload it to backups container
    - Verify it's private
3. **Set up archive:**
    - Upload an old document to archives
    - Change its tier to Cool
4. **Document your setup:**
    - Create a text file listing all URLs
    - Note which are public vs private

Verification Checklist

✅ **Completed Tasks**

Storage Account:

- [ ] Storage account created successfully
- [ ] Unique name assigned
- [ ] LRS redundancy configured

Blob Storage:

- [ ] Created 3 containers (images, backups, archives)
- [ ] Uploaded files to public and private containers
- [ ] Verified public access works
- [ ] Verified private access is blocked
- [ ] Changed blob access tier

File Share:

- [ ] Created file share
- [ ] Uploaded files
- [ ] Created directory structure
- [ ] Viewed connection information

Table Storage:

- [ ] Created products table
- [ ] Added 3 entities with properties
- [ ] Queried data successfully

Security:

- [ ] Generated SAS token
- [ ] Viewed access keys
- [ ] Understood access control concepts

Monitoring:

- [ ] Viewed storage metrics
- [ ] Checked storage usage
- [ ] Reviewed activity logs

Key Concepts Summary

Storage Services Comparison

Service	Use Case	Example
Blob Storage	Unstructured data, files, media	Images, videos, backups, logs
File Share	Shared file access, legacy apps	Shared documents, configuration files
Table Storage	NoSQL data, semi-structured	Product catalogs, user profiles
Queue Storage	Message queuing, async processing	Task queues, event processing

Access Tiers Cost Comparison

Tier	Storage Cost	Access Cost	Use When
Hot	Highest	Lowest	Daily access
Cool	Lower	Higher	Monthly access (30+ days)
Archive	Lowest	Highest	Yearly access (180+ days)

Security Options

1. **Access Keys:** Full access (protect carefully)
2. **SAS Tokens:** Time-limited, permission-restricted access
3. **Public Access:** Anonymous access to specific containers
4. **Private Access:** Requires authentication
5. **Azure AD:** Role-based access control (enterprise)

Clean Up Resources

To avoid charges:

1. Go to **Resource Groups**
2. Click **rg-storage-lab**
3. Click **Delete resource group**
4. Type the resource group name: rg-storage-lab
5. Click **Delete**
6. All resources will be permanently deleted

Note: With minimal usage, this lab costs less than $0.10

Troubleshooting

Cannot access public blob URL

- Verify container public access level is set to "Blob"
- Check storage account allows blob anonymous access
- Ensure URL is exactly correct (case-sensitive)

File upload fails

- Check file size (max 4.75 TB for block blobs)
- Verify storage account has capacity
- Check network connection

Cannot create table entity

- Ensure PartitionKey and RowKey are provided
- Property names cannot contain spaces
- Check data type matches value

SAS token doesn't work

- Verify token hasn't expired
- Check permissions match your needs (read/write/list)
- Ensure full token is copied including "?" at start

Next Steps

Azure Storage Account Workshop - 45 Minutes

Workshop Overview

Duration: 45 minutes
Skill Level: Beginner
Cost: Minimal (less than $1 with free tier)

What You'll Learn

- Create and configure Azure Storage Accounts
- Work with Blob Storage (containers and files)
- Use File Shares for shared storage
- Implement Table Storage for NoSQL data
- Configure access control and security
- Use Azure Storage Explorer and Portal

Prerequisites

- Active Azure subscription
- Web browser
- Basic understanding of cloud storage concepts

Lab Architecture

```
Azure Storage Account
 ├── Blob Storage (Containers)
|    ├── Public container (images)
|    ├── Private container (backups)
|   └── Cool tier container (archives)
 ├── File Share
|   └── Shared files accessible via SMB
 ├── Table Storage
|   └── NoSQL data (product catalog)
└── Queue Storage
    └── Message queue (optional)
```

Part 1: Create Storage Account (5 minutes)

Step 1.1: Navigate to Storage Accounts

1. Sign in to **Azure Portal** (portal.azure.com)
2. Click **Create a resource** (+ icon)
3. Search for **Storage account**
4. Click **Create**

Step 1.2: Configure Basics

1. **Subscription:** Select your subscription
2. **Resource Group:** Click **Create new**
   - Name: rg-storage-lab
   - Click **OK**
3. **Storage account name:** stglab + your initials + random numbers
   - Example: stglabjs12345
   - Must be 3-24 characters, lowercase letters and numbers only
   - Must be globally unique
4. **Region:** Select **East US** (or closest to you)
5. **Performance: Standard**
6. **Redundancy: Locally-redundant storage (LRS)**

Step 1.3: Advanced Settings

1. Click **Next: Advanced**
2. **Security:**
   - Allow enabling public access on containers: **Enabled** (checked)
   - Enable storage account key access: **Enabled**
3. **Blob storage:**
   - Allow blob anonymous access: **Enabled**
4. **Hierarchical namespace:** Disabled
5. Click **Next: Networking**

Step 1.4: Networking

1. **Network access:** Enable public access from all networks
2. Click **Next: Data protection**
3. Keep defaults, click **Next: Encryption**
4. Keep defaults, click **Next: Tags**
5. Skip tags, click **Review + create**

Step 1.5: Create

1. Review all settings
2. Click **Create**
3. Wait 30-60 seconds for deployment
4. Click **Go to resource**

Part 2: Work with Blob Storage (15 minutes)

Step 2.1: Create Containers

Container 1 - Public Images:

1. In left menu, click **Containers** (under Data storage)
2. Click **+ Container**
3. **Name:** images
4. **Public access level: Blob (anonymous read access for blobs only)**
5. Click **Create**

Container 2 - Private Backups: 6. Click **+ Container** again 7. **Name:** backups 8. **Public access level: Private (no anonymous access)** 9. Click **Create**

Container 3 - Archive Storage: 10. Click **+ Container** 11. **Name:** archives 12. **Public access level: Private** 13. Click **Create**

Step 2.2: Upload Files to Images Container

Create a sample HTML file:

1. Open Notepad or any text editor on your computer
2. Paste this content:

```
<!DOCTYPE html>
<html>
<head><title>Test Page</title></head>
<body>
<h1>Hello from Azure Blob Storage!</h1>
<p>This file is publicly accessible.</p>
</body>
</html>
```

3. Save as index.html on your desktop

Upload to Azure: 4. In Azure Portal, click the **images** container 5. Click **Upload** 6. Click **Browse for files** 7. Select the index.html file you created 8. **Advanced settings:**

- Blob type: **Block blob**
- Block size: Keep default
- Access tier: **Hot**

9. Click **Upload**
10. Wait for upload to complete

Step 2.3: Access Public Blob

1. Click on the uploaded **index.html** file
2. In the properties, find and copy the **URL**
   - Example: https://stglabjs12345.blob.core.windows.net/images/index.html
3. Open a new browser tab
4. Paste the URL and press Enter
5. You should see your HTML page displayed
6. ✅ **Success!** This demonstrates public blob access

Step 2.4: Upload to Private Container

Create a backup file:

1. Create a new text file called backup-data.txt
2. Add some content: "This is confidential backup data"
3. Save it

Upload: 4. Go back to Azure Portal 5. Click **Containers**, then click **backups** container 6. Click **Upload** 7. Select backup-data.txt 8. Click **Upload**

Test private access: 9. Click on **backup-data.txt** 10. Copy the URL 11. Open in new browser tab 12. You should see: **"Public access is not permitted"** or authentication error 13. ✅ **Success!** This demonstrates private storage security

Step 2.5: Change Access Tier (Cost Optimization)

1. Go to **archives** container
2. Upload any file (reuse backup-data.txt)
3. After upload, click on the file
4. Click **Change tier** button at top
5. **Access tier:** Select **Cool**
6. Click **Save**

7. Notice: Cool tier costs less for storage, more for access

Understanding Tiers:

- **Hot:** Frequent access, higher storage cost, lower access cost
- **Cool:** Infrequent access (30+ days), lower storage cost
- **Archive:** Rare access (180+ days), lowest storage cost, hours to retrieve

---

Part 3: Configure File Share (10 minutes)

Step 3.1: Create File Share

1. In your storage account left menu, click **File shares**
2. Click **+ File share**
3. **Name:** shared-documents
4. **Tier: Transaction optimized**
5. **Provisioned capacity:** 5 GiB (minimum)
6. Click **Create**

Step 3.2: Upload Files to File Share

1. Click on **shared-documents** file share
2. Click **Upload**
3. Click **Browse for files**
4. Select one or more files from your computer
5. Click **Upload**
6. Files are now in shared storage

Step 3.3: Create Directory Structure

1. In the file share, click **+ Add directory**
2. **Name:** projects
3. Click **OK**
4. Click on **projects** folder
5. Click **+ Add directory**
6. **Name:** 2024
7. Click **OK**
8. You now have nested directories: /projects/2024/

Step 3.4: Get Connection Information

1. Go back to **shared-documents** overview
2. Click **Connect** button at top
3. You'll see connection scripts for:
    - **Windows:** PowerShell script using net use
    - **Linux:** Bash script using mount
    - **macOS:** Bash script
4. **Note:** These scripts allow mounting the share as a network drive

Example connection string format:

\\<storage-account>.file.core.windows.net\shared-documents

Part 4: Work with Table Storage (8 minutes)

Step 4.1: Access Tables

1. In storage account left menu, scroll down to **Tables**
2. Click **Tables**
3. Click **+ Table**
4. **Name:** products
5. Click **OK**

Step 4.2: Add Data Using Storage Browser

1. Click on **products** table
2. Click **Storage Browser** in left menu (under Data storage)
3. Navigate to **Tables** → **products**
4. Click **Add entity**

Entity 1 - Laptop: 5. **PartitionKey:** electronics 6. **RowKey:** 001 7. Click **Add property**

- Property name: ProductName
- Type: String
- Value: Laptop Pro 15

8. Click **Add property**
    - Property name: Price
    - Type: Double
    - Value: 1299.99
9. Click **Add property**

    - Property name: InStock
    - Type: Boolean
    - Value: true

10. Click **Insert**

Entity 2 - Mouse: 11. Click **Add entity** again 12. **PartitionKey:** electronics 13. **RowKey:** 002 14. Add properties:

- ProductName (String): Wireless Mouse
- Price (Double): 29.99
- InStock (Boolean): true

15. Click **Insert**

Entity 3 - Notebook: 16. Add one more entity: 17. **PartitionKey:** office-supplies 18. **RowKey:** 001 19. Add properties:

- ProductName (String): Notebook A4
- Price (Double): 5.99
- InStock (Boolean): false

20. Click **Insert**

Step 4.3: Query Table Data

1. In the table view, you'll see all three entities
2. Notice they're grouped by PartitionKey
3. **PartitionKey** = Category grouping (for performance)
4. **RowKey** = Unique identifier within partition
5. ✅ **Success!** You've created a NoSQL database

Part 5: Security and Access Control (5 minutes)

Step 5.1: Generate Shared Access Signature (SAS)

1. Go to your storage account overview
2. In left menu, click **Shared access signature** (under Security + networking)
3. **Allowed services:** Check **Blob** only
4. **Allowed resource types:** Check all (Service, Container, Object)
5. **Allowed permissions:** Check **Read** and **List** only
6. **Start time:** Keep current date/time
7. **End time:** Set to 1 hour from now

8. **Allowed protocols:** HTTPS only
9. Click **Generate SAS and connection string**
10. Copy the **SAS token** (starts with ?sv=...)
11. Copy the **Blob service SAS URL**

Step 5.2: Test SAS Token Access

1. Take the URL from your **images/index.html** file
2. Add the SAS token to the end:
   - Original: https://stglabjs12345.blob.core.windows.net/images/index.html
   - With SAS: https://stglabjs12345.blob.core.windows.net/images/index.html?sv=... (paste full token)
3. This URL now provides temporary, limited access

Step 5.3: View Access Keys

1. In left menu, click **Access keys**
2. You'll see **key1** and **key2**
3. Click **Show** next to key1
4. These keys provide **full access** to the storage account
5. **Best practice:** Use SAS tokens for limited access instead of sharing keys

Step 5.4: Configure Firewall (Optional)

1. In left menu, click **Networking**
2. Under **Firewalls and virtual networks:**
3. You can restrict access to:
   - Specific IP addresses
   - Virtual networks
   - Azure services
4. **For this lab:** Keep "Public network access: Enabled from all networks"

Part 6: Monitoring and Management (2 minutes)

Step 6.1: View Metrics

1. In left menu, click **Metrics** (under Monitoring)
2. Click **Metric:** Select **Blob Capacity**
3. View storage usage over time

4. Click **Add metric**
5. Select **Transactions** to see activity

Step 6.2: Check Storage Usage

1. In left menu, click **Overview**
2. Under **Essentials**, note:
    - **Used capacity:** Shows how much storage you're using
    - **Performance tier:** Standard
    - **Replication:** LRS

Step 6.3: Access Activity Logs

1. In left menu, click **Activity log**
2. View all operations performed:
    - Container creation
    - File uploads
    - Configuration changes
3. Useful for auditing and troubleshooting

Clean Up Resources

To avoid charges:

1. Go to **Resource Groups**
2. Click **rg-storage-lab**
3. Click **Delete resource group**
4. Type the resource group name: rg-storage-lab
5. Click **Delete**
6. All resources will be permanently deleted

Note: With minimal usage, this lab costs less than $0.10

Troubleshooting

Cannot access public blob URL

- Verify container public access level is set to "Blob"
- Check storage account allows blob anonymous access

- Ensure URL is exactly correct (case-sensitive)

File upload fails

- Check file size (max 4.75 TB for block blobs)
- Verify storage account has capacity
- Check network connection

Cannot create table entity

- Ensure PartitionKey and RowKey are provided
- Property names cannot contain spaces
- Check data type matches value

SAS token doesn't work

- Verify token hasn't expired
- Check permissions match your needs (read/write/list)
- Ensure full token is copied including "?" at start

Bonus: Azure CLI for Storage Operations (15 minutes)

Setup Azure CLI

Option 1: Use Azure Cloud Shell (Recommended for this lab)

1. In Azure Portal, click the **Cloud Shell** icon (>_) at the top
2. Select **Bash** environment
3. Wait for shell to initialize
4. Azure CLI is pre-installed and authenticated

Option 2: Install Locally (Windows/Mac/Linux)

- Windows: Download from https://aka.ms/installazurecliwindows
- Mac: brew install azure-cli
- Linux: curl -sL https://aka.ms/InstallAzureCLIDeb | sudo bash

CLI Exercise 1: Create Storage Account

```
# Set variables

RESOURCE_GROUP="rg-storage-cli-lab"

LOCATION="eastus"

STORAGE_ACCOUNT="stgcli$(openssl rand -hex 4)"
```

```
# Create resource group
az group create \
  --name $RESOURCE_GROUP \
  --location $LOCATION

# Create storage account
az storage account create \
  --name $STORAGE_ACCOUNT \
  --resource-group $RESOURCE_GROUP \
  --location $LOCATION \
  --sku Standard_LRS \
  --kind StorageV2

# Get storage account key
STORAGE_KEY=$(az storage account keys list \
  --resource-group $RESOURCE_GROUP \
  --account-name $STORAGE_ACCOUNT \
  --query '[0].value' \
  --output tsv)

echo "Storage Account: $STORAGE_ACCOUNT"
echo "Key retrieved successfully"
```

CLI Exercise 2: Blob Operations

Create Container:

```
# Create public container
az storage container create \
  --name images \
  --account-name $STORAGE_ACCOUNT \
  --account-key $STORAGE_KEY \
  --public-access blob
```

```
# Create private container
az storage container create \
  --name documents \
  --account-name $STORAGE_ACCOUNT \
  --account-key $STORAGE_KEY \
  --public-access off

# List containers
az storage container list \
  --account-name $STORAGE_ACCOUNT \
  --account-key $STORAGE_KEY \
  --output table
```

Upload Files:

```
# Create a test file
echo "Hello from Azure CLI!" > test-file.txt

# Upload to blob storage
az storage blob upload \
  --container-name images \
  --name test-file.txt \
  --file test-file.txt \
  --account-name $STORAGE_ACCOUNT \
  --account-key $STORAGE_KEY

# Upload with metadata
az storage blob upload \
  --container-name images \
  --name document.txt \
  --file test-file.txt \
  --metadata author="AzureUser" department="IT" \
  --account-name $STORAGE_ACCOUNT \
  --account-key $STORAGE_KEY


# List blobs
az storage blob list \
  --container-name images \
  --account-name $STORAGE_ACCOUNT \
  --account-key $STORAGE_KEY \
  --output table
```