2. Develop for Azure Storage → 2.1 Develop solutions that use Azure Blob Storage → 2.2.2 Perform operations by using the appropriate SDK

1. What SDKs are supported for Azure Blob Storage operations?
2. How do you create and upload a blob using the Azure SDK (C#, Python)?
3. How do you download a blob using the Azure SDK?
4. How do you list blobs inside a container?
5. How do you delete a blob using the Azure SDK?
6. How do you perform conditional operations (e.g., upload if not exists)?
7. How do you use a stream to upload/download blobs?
8. How do you handle large blobs efficiently (e.g., upload in blocks)?
9. How do you set retries and timeouts in SDK operations?
10. What are best practices for SDK usage in production?

---

1. What SDKs are supported for Azure Blob Storage operations?

- **Official Azure SDKs**:
  - .NET (Azure.Storage.Blobs)
  - Python (azure-storage-blob)
  - Java (azure-storage-blob)
  - JavaScript/TypeScript (e.g., @azure/storage-blob)

---

2. How do you create and upload a blob using the Azure SDK (C#, Python)?

- **C# Example:**

```
BlobClient blobClient = containerClient.GetBlobClient("myblob.txt");
await blobClient.UploadAsync("localfile.txt", overwrite: true);
```

---

3. How do you download a blob using the Azure SDK?

- **C# Example:**

```
BlobDownloadInfo download = await blobClient.DownloadAsync();
using (FileStream fs = File.OpenWrite("downloaded.txt"))
{
   await download.Content.CopyToAsync(fs);
}
```

---

4. How do you list blobs inside a container?

- **C# Example:**

```
await foreach (BlobItem blobItem in containerClient.GetBlobsAsync())
{
   Console.WriteLine(blobItem.Name);
}
```

---

5. How do you delete a blob using the Azure SDK?

- **C# Example:**

```
await blobClient.DeleteIfExistsAsync();
```

---

6. How do you perform conditional operations (e.g., upload if not exists)?

- Set the conditions parameter with an If-None-Match: * condition.
- **C# Example:**

```
var conditions = new BlobRequestConditions { IfNoneMatch = new ETag("*") };
await blobClient.UploadAsync("localfile.txt", conditions: conditions);
```

7. How do you use a stream to upload/download blobs?

- **Upload Stream (C#):**

```
using var stream = File.OpenRead("localfile.txt");
await blobClient.UploadAsync(stream, overwrite: true);
```

- **Download Stream (C#):**

```
BlobDownloadInfo download = await blobClient.DownloadAsync();
using var file = File.OpenWrite("output.txt");
await download.Content.CopyToAsync(file);
```

---

8. How do you handle large blobs efficiently (e.g., upload in blocks)?

- Use **UploadAsync** for files up to 256 MB (default).
- For larger files, use **UploadAsync** with automatic chunking or manually use **BlockBlobClient.StageBlockAsync** and **CommitBlockListAsync**.

---

9. How do you set retries and timeouts in SDK operations?

- **C# Example:**

```
BlobClientOptions options = new BlobClientOptions
{
   Retry =
   {
      MaxRetries = 5,
      Delay = TimeSpan.FromSeconds(2),
      MaxDelay = TimeSpan.FromSeconds(10),
      Mode = RetryMode.Exponential
   }
};
var blobServiceClient = new BlobServiceClient(connectionString, options);
```

---

10. What are best practices for SDK usage in production?

- Always set appropriate retry policies and timeouts.
- Prefer streams for large files.
- Handle exceptions explicitly (e.g., RequestFailedException in C#).
- Reuse BlobServiceClient, BlobContainerClient, and BlobClient instances (they are thread-safe).
- Secure secrets and connection strings (use Azure Managed Identity if possible).