**Create the Storage Account (Portal)**

1. **Go to Resource Group** → FetusApp-Development.
2. Click **Create** → search for **Storage account** (you want the plain Microsoft one) → **Create**.
3. **Basics tab**:
   * Subscription: your current one.
   * Resource group: FetusApp-Development.
   * Storage account name: unique, e.g. fetusappdocsdev.
   * Region: **UK South** (same as your App Service).
   * Performance: **Standard**.
   * Redundancy:
     + Cheapest: **LRS** (locally redundant, ~3 copies in one DC).
     + Safer: **GRS** (~doubles cost, geo-redundant).
4. **Advanced tab**:
   * Disable public access.
   * Enable soft delete for blobs (7–30 days, your choice).
   * Enable versioning (checkbox).
5. **Networking tab**: for now allow public endpoints with selected networks; later you can lock it down.
6. **Review + Create** → hit **Create**.

**Create the Container**

1. Once deployed, open the new storage account.
2. Go to **Containers** → **+ Container**.
3. Name: patient-docs.
4. Public access: **Private (no anonymous access)**.
5. Create.

**Assign Role to App Service**

1. In the storage account, go to **Access control (IAM)**.
2. **+ Add → Add role assignment**.
3. Role: **Storage Blob Data Contributor**.
4. Assign access to: **Managed identity**.
5. Pick your App Service’s Managed Identity (oidc-msi-96b9 in your screenshot).
6. Save.

**Set CORS on the storage account (Portal)**

1. Storage account → **Resource sharing (CORS)** → **Blob service**.
2. **+ Add** a rule:
   * **Allowed origins:** your web app origin (e.g. https://fetusapp-dev.azurewebsites.net)
   * **Allowed methods:** GET, PUT, HEAD, OPTIONS
   * **Allowed headers:** x-ms-\*,content-type,content-length
   * **Exposed headers:** ETag,x-ms-request-id,x-ms-version,Content-Length,Content-Type
   * **Max age (seconds):** 3600
3. **Save**.

(You don’t need Authorization headers—SAS is in the URL.)

**Folders (important)**

* Blob has **virtual folders** (just name prefixes). You “create a folder” by uploading to a path.
* Use a path like: patients/{patientId}/{docId}/{safeFilename}.
* **List files for a patient:** list blobs with prefix patients/{patientId}/.
* **Create folder on first upload:** just upload with that prefix—no extra step.
* **Delete folder when last file gone:** after deleting the final blob with that prefix, the “folder” disappears automatically.  
  (If you ever want to force-clean, delete all blobs with the prefix.)

Want the tiny Flask endpoint next to mint a **write SAS** for patients/{patientId}/{uuid}/{filename}?