SOP-SP-001_SharePoint_ReadOnly_v1.0.0

SOP-SP-001_SharePoint_ReadOnly_v1.0.0

Standard Operating Procedure: SOP-SP-001

SOP-SP-001: SharePoint Read-Only Access Configuration

Document ID	Version	Effective Date	Author
SOP-SP-001	1.0	2025-12-02	Gemini Agent

1.0 Purpose

To establish a standardized and secure procedure for configuring a read-only Azure AD application. This application will grant the OberaConnect Second Brain system access to download and process documents from SharePoint. The procedure is designed to have zero impact on existing user permissions, data integrity, or Microsoft 365 settings.

2.0 Scope

This SOP applies to all Azure Administrators within OberaConnect who are tasked with configuring data access for internal systems. This procedure is strictly limited to the creation and configuration of the `SecondBrain-ReadOnly-SharePoint` application.

3.0 Responsibilities

- **Azure Administrator:** Responsible for executing all steps outlined in this SOP, ensuring permissions are correctly configured, and securely storing the generated credentials.

4.0 Prerequisites

- Administrative access to the OberaConnect Azure Portal (https://portal.azure.com).
- Access to the server environment where the Second Brain application is hosted to configure environment variables.

5.0 Procedure

This procedure is divided into three main parts: App Registration in Azure, local credential configuration, and connection testing.

5.1 Part A: Azure App Registration 5.1.1 Navigate to Azure App Registrations 1. Log in to the **Azure Portal**: https://portal.azure.com 2. From the left sidebar, select **Azure Active Directory**. 3. Select **App registrations**. 4. Click **+ New registration**.

5.1.2 Register the Application 1. Fill in the registration form with the following details: - **Name:** `SecondBrain-ReadOnly-SharePoint` - **Supported account types:** Select **Accounts in this organizational directory only (OberaConnect only - Single tenant)**. - **Redirect URI:** Leave this section blank. 2. Click the **Register** button to create the application.

5.1.3 Record Application Identifiers 1. After registration, you will be on the app's overview page. 2. Copy and securely store the following values for use in Step 5.2: - `Application (client) ID` - `Directory (tenant) ID`

5.1.4 Create a Client Secret 1. From the left sidebar, select **Certificates & secrets**. 2. Click **+ New client secret**. 3. Provide the following details: - **Description:** `SecondBrain Read Access` - **Expires:** `24 months` (or as per company policy). 4. Click **Add**. 5. **CRITICAL:** Immediately copy the **Value** of the client secret. This value is only displayed once. Securely store it for use in Step 5.2.

5.1.5 Grant Read-Only API Permissions 1. From the left sidebar, select **API permissions**. 2. Click **+ Add a permission**. 3. Select **Microsoft Graph**. 4. Select **Application permissions**. **Do not use Delegated permissions.** 5. In the "Select permissions" search box, find and check the following permissions: - `Sites.Read.All` - `Files.Read.All` 6. **VERIFY:** Ensure no "Write," "Manage," or "FullControl" permissions are added. This application must remain **READ-ONLY**. 7. Click **Add permissions**.

5.1.6 Grant Admin Consent 1. On the API permissions page, click the **Grant admin consent for OberaConnect** button. 2. A confirmation prompt will appear. Click **Yes**. 3. Verify that a green checkmark appears in the "Status" column for both permissions.

5.2 Part B: Credential Configuration

1. Connect to the WSL/Linux system hosting the application.

2. Navigate to the project directory: `bash cd /path/to/project`
3. Open the environment variable file for editing: `bash nano .env`
4. Add the following lines to the file, replacing the placeholder text with the values copied from steps 5.1.3 and 5.1.4: `bash # SharePoint Read-Only Access AZURE_TENANT_ID=your-tenant-id-here AZURE_CLIENT_ID=your-client-id-here AZURE_CLIENT_SECRET=your-secret-value-here`
5. Save the file and exit the editor (Ctrl+X, then Y, then Enter).

5.3 Part C: Connection Testing

1. Ensure you are in the project directory: `bash cd /path/to/project`
2. Execute the SharePoint importer script to test the connection: `bash ./venv/bin/python sharepoint_importer.py`
3. A successful test will output a list of SharePoint sites, confirming the credentials and permissions are correct.

6.0 Security & Compliance

- **Capabilities:** The application is strictly limited to reading SharePoint sites/libraries and downloading copies of files. It can view file metadata.
- **Restrictions:** The application **CANNOT** modify, delete, or create files in SharePoint. It cannot alter permissions, access user data (e.g., email), or change any Microsoft 365 settings.
- **Isolation:** This is a standalone app registration, completely isolated from other OberaConnect Azure applications. It uses its own credentials and permissions.

7.0 Revocation Procedure

To revoke access for this application, perform the following steps: 1. Navigate to **Azure Portal > Azure Active Directory > App registrations**. 2. Search for and select `SecondBrain-ReadOnly-SharePoint`. 3. On the application overview page, click **Delete**. 4. Confirm the deletion. Access will be immediately and permanently revoked.

8.0 Verification Checklist

Step	Verification Point	Completed ( )
1	App is named `SecondBrain-ReadOnly-SharePoint`.	
2	Permissions are **Application permissions**, not Delegated.	

Step	Verification Point	Completed ( )
3	Only `Sites.Read.All` and `Files.Read.All` are granted.	
4	No write, manage, or full control permissions are present.	
5	Admin consent has been granted (green checkmarks visible).	
6	Client ID, Tenant ID, and Client Secret are copied correctly.	
7	Credentials have been securely added to the `.env` file.	
8	The connection test script (`sharepoint_importer.py`) runs successfully.	

8.1 Troubleshooting

Issue	Cause	Resolution
"Insufficient privileges to complete the operation"	Admin consent not granted or not propagated	Go to API Permissions, click "Grant admin consent". Wait 5-10 minutes for propagation.
"Failed to get access token"	Incorrect credentials in `.env`	Verify AZURE_TENANT_ID, AZURE_CLIENT_ID, and AZURE_CLIENT_SECRET are correct. Check for extra spaces/characters.
"AADSTS700016: Application not found"	Wrong tenant ID or app deleted	Verify tenant ID matches your Azure AD. Check app registration still exists.
"The specified client secret has expired"	Client secret expired	Create new client secret in Azure Portal > Certificates & secrets. Update `.env`.

Issue	Cause	Resolution
"Access denied" to specific site	Site-level permissions restrict app access	App has tenant-wide read, but site admins can block. Check site permissions or use delegated flow.
Connection timeout	Network/firewall blocking Azure endpoints	Verify outbound HTTPS to `login.microsoftonline.com` and `graph.microsoft.com`.
"Resource does not exist"	Wrong site ID or site deleted	Verify site ID using Graph Explorer: `GET /sites?search=sitename`.
Empty file list returned	No files in library or wrong drive ID	Verify drive ID. Check library has files. Try listing root: `/drives/{drive-id}/root/children`.

Diagnostic Commands:

```
# Test authentication
curl -X POST "https://login.microsoftonline.com/{tenant_id}/oauth2/v2.0/token" \
  -d "client_id={client_id}&scope=https://graph.microsoft.com/.default&client_secret={secret

# Test Graph API access (with token)
curl -H "Authorization: Bearer {token}" "https://graph.microsoft.com/v1.0/sites"
```

Python Test Script:

```
from azure.identity import ClientSecretCredential
from msgraph.core import GraphClient

credential = ClientSecretCredential(
    tenant_id=os.getenv("AZURE_TENANT_ID"),
    client_id=os.getenv("AZURE_CLIENT_ID"),
    client_secret=os.getenv("AZURE_CLIENT_SECRET")
)
client = GraphClient(credential=credential)
result = client.get('/sites')
print(result.json())
```

9.0 Revision History

Version	Date	Author	Description
1.0	2025-12-02	Gemini Agent	Initial document creation.

Version	Date	Author	Description
1.1	2025-12-29	Jeremy Smith	SME Review: Added Section 8.1 (Troubleshooting) with common Azure AD app and Graph API issues. Added diagnostic commands.

10.0 Approval

Role	Name	Signature	Date
Document Owner			
Approved By			