staedean

STAEDEAN

Installation Guide for Life Sciences Solution Document Intelligence

Test Group Setup

Published: April 2026

staedean

TABLE OF CONTENTS

staedean

1. Introduction

LSS Document Intelligence: Test Group Setup is a STAEDEAN solution that extracts structured test specifications from PDF documents and performs intelligent semantic matching against a reference dataset of known tests. The solution uses Azure AI services (Document Intelligence and Azure OpenAI) deployed within your own Azure tenant.

This guide provides step-by-step instructions for provisioning the required Azure resources and deploying the application. It is intended for Azure administrators with Contributor-level access to an Azure subscription.

1.1 Architecture Overview

The deployment creates the following resources within a single resource group on your Azure tenant:

- **Azure App Service** (Linux, Python) — hosts the application API
- **Azure Document Intelligence** — extracts tables and text from PDF documents
- **Azure OpenAI** (GPT-5.1 deployment) — performs semantic similarity matching
- **Azure Storage Account** — stores input documents, reference data, and output logs
- **Azure Key Vault** — stores secrets consumed by the Dynamics 365 integration
- **Application Insights + Log Analytics** — monitoring and diagnostics

All inter-service authentication uses the App Service's system-assigned Managed Identity with Azure RBAC role assignments. No service keys are stored in application settings.

1.2 Security Model

- HTTPS-only with TLS 1.2 minimum
- Azure AD authentication (Easy Auth) protects all API endpoints
- Managed Identity eliminates credential storage for Azure service calls
- Blob storage public access is disabled
- Key Vault uses RBAC authorization with soft delete enabled (90-day retention)

2. PREREQUISITES

Before beginning, ensure the following:

2.1 Azure Subscription Requirements

- An active Azure subscription with sufficient quota
- **Contributor** role (or higher) on the subscription or target resource group
- **User Access Administrator** role (required for RBAC role assignments created by the ARM template)
- Permission to register applications in Azure Active Directory (Azure AD)

2.2 Azure OpenAI Access

The solution requires a GPT-5.1 model deployment. GPT-5.1 may require explicit access approval on your subscription. If you have not previously used Azure OpenAI:

1. Navigate to https://aka.ms/oai/access
2. Submit the access request form for your subscription
3. Wait for approval confirmation before proceeding

NOTE: Azure OpenAI is available in a limited set of regions. See Appendix A for the list of supported regions. You must select a supported region for the openAiLocation parameter during infrastructure deployment.

2.3 Tools

One of the following:

- Azure Portal (https://portal.azure.com)
- Azure CLI (az) version 2.50 or later
- Azure PowerShell (Az module) version 10.0 or later

The EasyAuthClientSPN.ps1 script in Phase 6 requires:

- PowerShell 7.x or later
- Microsoft.Graph PowerShell modules (the script will install them automatically if missing)

2.4 Artifacts Provided by STAEDEAN

You will receive the following files from STAEDEAN (included in the ExternalDeployment package):

- azuredeploy.json — ARM template for infrastructure provisioning
- codedeployment.json — ARM template for application code deployment (pre-configured with package URL)
- EasyAuthClientSPN.ps1 — PowerShell script for client service principal setup

3. Phase 1 - Create a Resource Group

Create a dedicated resource group to contain all solution resources.

3.1 Azure Portal

1. Sign in to the Azure Portal (https://portal.azure.com).
2. Navigate to **Resource groups > + Create**.
3. Configure:
   a. **Subscription**: Select your target subscription.
   b. **Resource group name**: Choose a descriptive name (e.g., rg-lss-docintel-prod).
   c. **Region**: Select a region that supports Azure OpenAI GPT-5.1 (see Appendix A).
4. Select **Review + create**, then **Create**.

3.2 Azure CLI

```
az group create \
  --name rg-lss-docintel-prod \
  --location <region>
```

Replace <region> with a supported Azure OpenAI region (e.g., eastus2, swedencentral).

staedean

4. Phase 2 - Register an Azure AD Application

The App Service uses Azure AD authentication (Easy Auth) to secure its API endpoints. You must create an App Registration before deploying the infrastructure.

4.1 Create the App Registration

1. In the Azure Portal, navigate to **Azure Active Directory > App registrations > + New registration**.
2. Configure:
   a. **Name**: lss-docintel-api (or a name consistent with your naming convention)
   b. **Supported account types**: Accounts in this organizational directory only (Single tenant)
   c. **Redirect URI**: Leave blank for now (the ARM template configures this automatically)
3. Select **Register**.
4. On the **Overview** page, record the following values — you will need them during infrastructure deployment:
   a. **Application (client) ID** → this is the aadClientId parameter
   b. **Directory (tenant) ID** → this is the aadTenantId parameter

4.2 Create a Client Secret

1. Navigate to **Certificates & secrets > Client secrets > + New client secret**.
2. Configure:
   a. **Description**: lss-docintel-easyauth
   b. **Expires**: Select an appropriate expiry period per your organization's policy
3. Select **Add**.
4. Immediately copy the secret **Value** — this is the aadClientSecret parameter.

Important: The secret value is only displayed once. Store it securely (e.g., in a password manager) until you complete the infrastructure deployment.

4.3 Configure API Permissions

[PLACEHOLDER — Specific API permission requirements to be confirmed. Consult your STAEDEAN contact if additional permissions beyond the default Microsoft Graph User.Read are required.]

5. Phase 3 – Deploy Azure Infrastructure

The azuredeploy.json ARM template provisions all required Azure resources, configures Managed Identity with RBAC role assignments, and enables Azure AD authentication on the App Service.

5.1 Gather Parameters

Before starting the deployment, prepare the following parameter values:

Parameter	Required	Description
appName	Yes	Base name for all resources. Must be globally unique. Recommended format: lss-docintel-<company>-<env>. Example: lss-docintel-contoso-prod
aadClientId	Yes	Application (client) ID from Phase 2
aadTenantId	Yes	Directory (tenant) ID from Phase 2
aadClientSecret	Yes	Client secret value from Phase 2
location	No	Azure region for most resources. Defaults to the resource group's region.
openAiLocation	No	Azure region for the OpenAI resource. Defaults to the resource group's region. Must be a region that supports GPT-5.1 (see Appendix A).

NOTE on appName: The appName value is used as a prefix to derive names for all resources. For example, if appName is lss-docintel-contoso-prod:

- **Web App**: lss-docintel-contoso-prod
- **App Service Plan**: lss-docintel-contoso-prod-plan
- **OpenAI**: lss-docintel-contoso-prod-openai
- **Document Intelligence**: lss-docintel-contoso-prod-docintel
- **Storage Account**: stlssdocintelcontosoprod (alphanumeric, max 24 chars)
- **Key Vault**: kv-lssdocintelcontosoprod (alphanumeric, max 24 chars)

5.2 Deploy via Azure Portal

1. In the Azure Portal, navigate to the resource group created in Phase 1.
2. Select **Deployments > + Deploy from a custom template** (or navigate to **Custom deployment**).
3. Select **Build your own template in the editor**.
4. Select **Load file**, then choose the azuredeploy.json file.
5. Select **Save**.
6. Fill in the parameters:
    a. appName: Enter your chosen base name (e.g., lss-docintel-contoso-prod)
    b. aadClientId: Paste the Application (client) ID from Phase 2
    c. aadTenantId: Paste the Directory (tenant) ID from Phase 2
    d. aadClientSecret: Paste the client secret from Phase 2
    e. openAiLocation: Select a region from Appendix A (if different from the resource group region)
7. Select **Review + create**, then **Create**.
8. Wait for the deployment to complete. This typically takes 5–10 minutes.

5.3 Deploy via Azure CLI

```
az deployment group create \
  --resource-group rg-lss-docintel-prod \
  --template-file azuredeploy.json \
  --parameters \
    appName="lss-docintel-contoso-prod" \
    aadClientId="<client-id-from-phase-2>" \
    aadTenantId="<tenant-id-from-phase-2>" \
    aadClientSecret="<client-secret-from-phase-2>" \
    openAiLocation="swedencentral"
```

5.4 Record Deployment Outputs

After the deployment completes, navigate to **Deployments** in the resource group and select the completed deployment. Under **Outputs**, record the following values for reference:

- webAppUrl — the HTTPS endpoint of the deployed application
- storageAccountName — the storage account name
- keyVaultName — the Key Vault name
- keyVaultUri — the Key Vault URI
- docIntelligenceEndpoint — the Document Intelligence endpoint
- openAiEndpoint — the Azure OpenAI endpoint
- appInsightsInstrumentationKey — the Application Insights instrumentation key

staedean

6. Phase 4 - Deploy Application Code

After the infrastructure is provisioned, deploy the application code using the codedeployment.json ARM template.

6.1 Deploy via Azure Portal

1. In the Azure Portal, navigate to the same resource group.
2. Select **Deployments > + Deploy from a custom template**.
3. Select **Build your own template in the editor**.
4. Select **Load file**, then choose the codedeployment.json file.
5. Select **Save**.
6. Fill in the parameters:
   a. webAppName: Enter the same appName value used in Phase 3 (e.g., lss-docintel-contoso-prod)
7. Select **Review + create**, then **Create**.
8. Wait for the deployment to complete. This typically takes 3–5 minutes.

6.2 Deploy via Azure CLI

```
az deployment group create \
  --resource-group rg-lss-docintel-prod \
  --template-file codedeployment.json \
  --parameters \
    webAppName="lss-docintel-contoso-prod"
```

NOTE: The codedeployment.json template contains a pre-configured package URL provided by STAEDEAN. No modification of this URL is required.

7. Phase 5 - Verify the Deployment

After both deployments complete, verify that the application is running correctly.

7.1 Health Check

1. Open a web browser and navigate to:

   https://<appName>.azurewebsites.net/health

   Replace <appName> with the value used in Phase 3.
2. You will be redirected to the Azure AD login page. Sign in with a user account from your tenant.
3. After authentication, the health endpoint should return a JSON response indicating the service status.

7.2 API Documentation

The application provides interactive API documentation at:

- **Swagger UI**: https://<appName>.azurewebsites.net/docs
- **ReDoc**: https://<appName>.azurewebsites.net/redoc

7.3 Troubleshooting

If the application does not respond:

1. In the Azure Portal, navigate to the App Service resource.
2. Select **Log stream** (under **Monitoring**) to view real-time application logs.
3. Select **Diagnose and solve problems** for guided troubleshooting.
4. Verify that the Application Insights resource is receiving telemetry by navigating to the Application Insights resource and selecting **Live Metrics**.

Common issues:

- **403 Forbidden after login**: Verify the App Registration's Application ID URI is set to api://<aadClientId>. This is normally configured automatically but can be verified under **Azure AD > App registrations > <your app> > Expose an API**.
- **500 Internal Server Error on startup**: Check the App Service log stream. The most common cause is the Azure OpenAI deployment not being available in the selected region.

8. Phase 6 - Configure the Client Service Principal

This phase creates a dedicated client service principal that enables Dynamics 365 Finance & Operations (and other services) to authenticate to the API using the OAuth 2.0 client credentials flow.

8.1 Prerequisites

- PowerShell 7.x or later
- **Global Administrator** or **Application Administrator** role in Azure AD

The Application (client) ID from Phase 2 (the server App Registration)

8.2 Run the Script

Open a PowerShell terminal and execute:

```
.\EasyAuthClientSPN.ps1 `
  -ServerAppId   "<aadClientId-from-phase-2>" `
  -ClientAppName  "LSS DocIntel API Client - <company>"
```

The script will:

1. Connect to Microsoft Graph (you may be prompted to sign in and consent)
2. Set the Application ID URI on the server App Registration (api://<AppId>) if not already set
3. Add an access_as_application App Role to the server App Registration
4. Create a new client App Registration and Service Principal
5. Generate a client secret (valid for 1 year by default)
6. Grant admin consent for the client credentials flow

8.3 Record the Output

The script outputs the following values. Record them securely:

- Tenant ID
- Token Endpoint
- Client ID (of the new client SPN)
- Client Secret
- Scope

Important: The client secret is displayed only once. Store it immediately in Azure Key Vault or another secure location.

8.4 Store the Client Secret in Key Vault

Store the client secret in the Key Vault created during Phase 3:

```
az keyvault secret set \
  --vault-name <keyVaultName-from-phase-3-outputs> \
  --name "d365-client-secret" \
  --value "<client-secret-from-script-output>"
```

9. Phase 7 - Dynamics 365 Finance & Operations Configuration

This phase focuses on configuring the required components within Dynamics 365 Finance & Operations (F&O) to enable integration with the deployed LSS Document Intelligence API

9.1 Configure the Microsoft Entra ID application

To enable F&O to access the AI Agent through OData services, the API endpoint must be configured in Microsoft Entra ID applications.

Navigation: System Administration → Setup → Microsoft Entra ID applications

Configuration Details

- **Client ID** – Enter the aadClientId obtained during the application registration process.
- **Name** – Provide a meaningful name that aligns with the organization's naming conventions.
- **User ID** – Assign an administrator or equivalent user with the required access permissions.

9.2 Configure Key Vault Parameters

Navigation: System Administration → Setup → Key Vault Parameters

Follow the steps below to configure the Key Vault parameters:

1. Click New to create a new record.
2. Enter an appropriate Name and Description.
3. Under the General tab:
    a. Paste the **Key Vault URL** copied from the Azure Portal into the Key Vault URL field.
4. Update the following fields:
    a. Key Vault Client
    b. Key Vault Secret Key
5. Set the Enabled for sealed bidding field to 'No'.
6. Navigate to the Secrets tab.
7. Click Add to create a new secret entry.
8. Provide the relevant:
    a. Name
    b. Description
9. Update the Secret Value with the value stored in the Azure Key Vault.
10. Set the Secret Type to Manual.
11. Click Validate and ensure that the validation is completed successfully.

9.3 License Retrieval

A dedicated license code named "**Life Sciences AI Hub**" is provided for this feature. Import the license into the system using **Staedean Solution Management**.

9.4 Enable the Configuration Key

Navigation: System Administration → Setup → License Configuration

A new configuration key named "**AI Hub**" is available in the system. This configuration key is disabled by default. To activate the feature, enable the "**AI Hub**" configuration key.

9.5 Configure Document Intelligence Parameters

Navigation: System Administration → Setup → Document Intelligence Parameters

Update the required parameters under the Test Group tab.

Parameter Details

- **Container Name** - Specify the Azure Blob Storage container name used for document storage.
- **AI Agent Endpoint URL** - Specify the endpoint URL of the AI Agent service.
- **Key Vault Certificate Name** - Select the Key Vault certificate created during the Key Vault setup process.
- **Minimum Confidence Score** - Specify the minimum confidence score required for a test to be considered a valid match with the document data. The accepted value range is between **0 and 1**.

staedean

10. Appendix A - Azure OpenAI Regional Availability

The GPT-5.1 model (GlobalStandard deployment) is available in the following Azure regions. Select one of these regions for the openAiLocation parameter during infrastructure deployment.

Region Name	Region Code
East US	eastus
East US 2	eastus2
North Central US	northcentralus
South Central US	southcentralus
West US	westus
West US 3	westus3
Sweden Central	swedencentral
UK South	uksouth
France Central	francecentral
Japan East	japaneast
Australia East	australiaeast

NOTE: Regional availability is subject to change. For the most current list, consult the Azure OpenAI models documentation: https://learn.microsoft.com/en-us/azure/ai-services/openai/concepts/models

If your preferred region is not listed, select the geographically closest supported region for the openAiLocation parameter. Other resources (App Service, Storage, etc.) can remain in your preferred region using the location parameter.

11. Appendix B - Deployed Resources Summary

The following Azure resources are created by the azuredeploy.json template:

Resource Type	Name Pattern	SKU / Tier
App Service Plan	<appName>-plan	Basic B1 (Linux)
App Service (Web App)	<appName>	—
Azure Document Intelligence	<appName>-docintel	S0
Azure OpenAI	<appName>-openai	S0
└ Model Deployment	gpt-5.1	GlobalStandard (cap. 50)
Storage Account	st<appName> (max 24)	Standard_LRS
└ Blob Container	pdf-documents	—
└ Blob Container	reference-data	—
└ Blob Container	output-logs	—
Key Vault	kv-<appName> (max 24)	Standard
└ Secret	storage-account-key	—
Log Analytics Workspace	<appName>-logs	PerGB2018 (30-day)
Application Insights	<appName>-insights	—

RBAC Role Assignments

The following roles are granted to the App Service Managed Identity:

- **Storage Blob Data Contributor** — on the Storage Account
- **Cognitive Services User** — on the Document Intelligence resource
- **Cognitive Services OpenAI User** — on the Azure OpenAI resource

12. Appendix C - ARM Template Parameters Reference

azuredeploy.json Parameters

Parameter	Type	Required	Default	Description
appName	string	Yes	—	Base name for all resources (must be globally unique)
location	string	No	Resource group location	Azure region for most resources
openAiLocation	string	No	Resource group location	Azure region for Azure OpenAI (see Appendix A)
pdfContainerName	string	No	pdf-documents	Blob container name for PDF input documents
refContainerName	string	No	reference-data	Blob container name for reference datasets
logsContainerName	string	No	output-logs	Blob container name for execution logs
aadClientId	string	Yes	—	Azure AD Application (client) ID from Phase 2
aadTenantId	string	Yes	—	Azure AD Directory (tenant) ID from Phase 2
aadClientSecret	securestring	Yes	—	Azure AD client secret from Phase 2

codedeployment.json Parameters

Parameter	Type	Required	Default	Description
webAppName	string	Yes	—	Name of the App Service created in Phase 3 (same as appName)