TEAMCENTER

Teamcenter AI Services – AWS Infrastructure Setup

December 2024

SIEMENS

About Siemens Digital Industries Software

Siemens Digital Industries Software is a leading global provider of product life cycle management (PLM) software and services with 7 million licensed seats and 71,000 customers worldwide. Headquartered in Plano, Texas, Siemens Digital Industries Software works collaboratively with companies to deliver open solutions that help them turn more ideas into successful products. For more information on Siemens Digital Industries Software products and services, visit www.siemens.com/plm.

Support Center: support.sw.siemens.com
Send Feedback on Documentation: support.sw.siemens.com/doc_feedback_form

Table of Contents

This document provides steps to set up these AWS services:

AWS Bedrock Setup

Provides managed API level access to various LLMs (Large Language Models), including embedding models. Please check for availability in your region.

AWS OpenSearch Serverless Setup

Serves as the vector database for Teamcenter AI Chat.

AWS Identity and Access Management Setup (IAM)

Manages access to AWS services and resources securely.

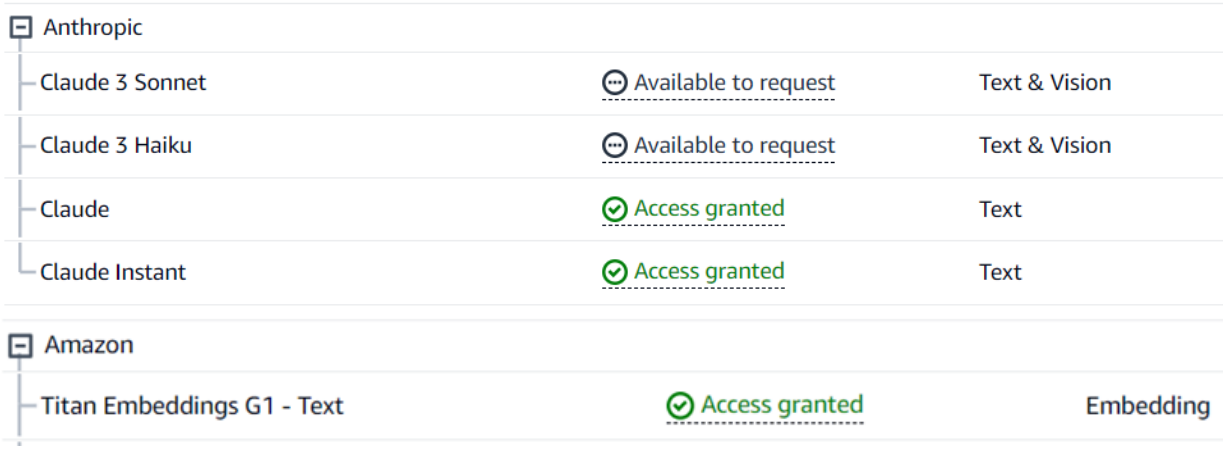
Section 1:
Enable AWS Bedrock

Enable AWS Bedrock

1. Log in to your AWS Management Console.
2. Navigate to the **Amazon Bedrock** service.
3. Follow the prompts to enable AWS Bedrock for your account.
4. Ensure you select the appropriate **Region** for your deployment.

Enable Access for Models

1. Navigate to the AWS Bedrock service dashboard.
2. Under **Model Access**, search for and then enable access to the following models.
   Note: We recommend that you use both models listed. These models have both been extensively tested.
   - **Anthropic: Claude Instant**
   - **Amazon: Titan Embeddings G1 – Text**
3. For each model, request model access.
4. Confirm that the models are successfully enabled and accessible. This step may take a few minutes.



Section 2:
Enable OpenSearch

Enable AWS OpenSearch

1. From the AWS Management Console, navigate to **Amazon OpenSearch Service**.
2. Follow the setup wizard to enable AWS OpenSearch for your account.
3. Choose the **Region** that matches your Bedrock deployment.

   Note: If you are signed into an IAM user in AWS console, then make sure to use that User. There is no need to create another IAM user. Skip creating the User and apply everything that is mentioned in this document to that specific user. Not using the designated IAM user role could results in issues with creating an index or other permission issues.

Create a New User and Allow Them Access to Assume Role

1. In IAM, click on **Users** > **Create User**.
2. Enter the user details and click **Next**.
3. Complete the setup and create the user.

Generate Remote Credentials

1. Navigate to the IAM Users dashboard and select the newly created user.
2. Under the **Security credentials** tab, click **Create access key**.
3. Select **Application running outside AWS** for your use case.
4. Provide a description and select **Create access key**. Record the **Access Key** and **Secret Access Key**.

Section 3:
Set up Access Roles and Policies

Set up Role and Policy to Allow Access to Bedrock

1. Navigate to **IAM** in the AWS Management Console.
2. Click on **Policies** > **Create Policy.**
3. Add the following JSON policy to allow necessary permissions for Bedrock.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "bedrock:*"
            ],
            "Resource": "*"
        }
    ]
}
```

4. Name the policy **BedrockPolicy.**
5. Create the policy.

Set up Role and Policy to Allow Access to OpenSearch

1. Navigate to **IAM** in the AWS Management Console.
2. Click on **Policies** > **Create Policy.**
3. Add the following JSON policy to allow necessary permissions for OpenSearch.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "aoss:*"
            ],
            "Resource": "*"
        }
    ]
}
```

4. Name the policy **OpenSearchPolicy.**
5. Create the policy.
6. In IAM, click **Roles** > **Create Role**.
7. Select **Custom Trust Policy**.
8. Add an inline policy to the role to allow necessary permissions for Bedrock and OpenSearch. Make sure to fill in the user information with the user that you just created.

```
 {

     "Version": "2012-10-17",

     "Statement": [

         {

             "Effect": "Allow",

             "Principal": {

                 "Federated": "arn:aws:iam::ACCOUNT_ID:saml-
 provider/WebkeyProvider"
```

```
            },
            "Action": "sts:AssumeRoleWithSAML",
            "Condition": {
                "StringEquals": {
                    "SAML:aud": "https://signin.aws.amazon.com/saml"
                 }
            }
        },
        {
            "Sid": "Statement1",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::ACCOUNT_ID:user/[USER_ID]"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

9. Name the role.
10. Set the maximum session duration to **12 hours.**
    NOTE: If this is not set to exactly **12 hours**, then your credentials will be labeled as invalid.
11. Attach the previously created **BedrockPolicy** and **OpenSearchPolicy** to the **Role**.
12. Complete the role creation.
13. Record the full ARN. Example: **arn:aws:iam::ACCOUNT_ID:role/*[ROLE_NAME]***
    Note: Make sure to have only one role to which your policies are attached. For now, our services cannot handle juggling separate roles for OpenSearch and Bedrock, so combine the policies into one role that can then be assumed by the IAM user.

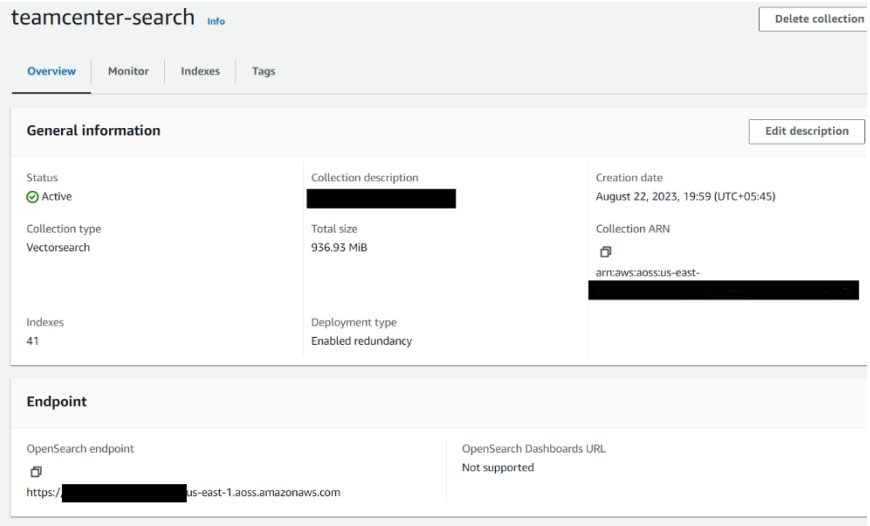
15. Navigate to the **Users** page in IAM, choose the user that you created**,** and scroll down to **permissions policies**. Click **Add permissions** and attach the following policy to allow the user to assume roles.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": [
                "arn:aws:iam::ACCOUNT_ID:role/[ROLE_NAME]"
            ]
        }
    ]
}
```

Create a New Collection of Type Vector on OpenSearch Serverless

1. In the OpenSearch Service dashboard, click **Collections** > **Create collection**.
2. Name the collection.
3. For **Collection type**, select **Vector search.**
4. For **Security**, choose **Standard Create.**
5. Under **Encryption**, select **Use AWS owned key**. This is the AWS KMS key that OpenSearch Serverless uses to encrypt your data.
6. Under **Network**, configure network settings for the collection.
    a) In **access type**, select **Public**.
    b) In **resource type**, choose both **Enable access to OpenSearch endpoints** and **Enable access to OpenSearch Dashboards.**
7. Complete the setup and record the **OpenSearch Endpoint**.

Example: **https://*[unique-identifier].[your-chosen-region]*.aoss.amazonaws.com**



Create a Data Access Policy for your Collection

1. Click on the collection that you have just created, scroll down, and click on **Manage Data Access**.

2. Click **Create Access Policy**. Name the policy and then click the **JSON** policy definition method and paste the following into the provided text box:

```
[
    {
        "Rules": [
            {
                "Resource": [
                    "collection/[COLLECTION_NAME]"
                ],
                "Permission": [
                    "aoss:CreateCollectionItems",
                    "aoss:DeleteCollectionItems",
                    "aoss:UpdateCollectionItems",
                    "aoss:DescribeCollectionItems"
                ],
                "ResourceType": "collection"
            },
            {
                "Resource": [
                    "index/[COLLECTION_NAME]/*"
                ],
                "Permission": [
                    "aoss:CreateIndex",
                    "aoss:DeleteIndex",
                    "aoss:UpdateIndex",
                    "aoss:DescribeIndex",
                    "aoss:ReadDocument",
                    "aoss:WriteDocument"
```

```
                ],
                "ResourceType": "index"
            }
        ],
        "Principal": [
            "arn:aws:iam::ACCOUNT_ID:user/[USER_ID]",
            "arn:aws:iam::ACCOUNT_ID:role/[ROLE_NAME]"
        ],
        "Description": "AI Chat Rule"
    }
]
```

3. Make sure to include your own values for the template provided above. This should grant your account access to be able to create an index from within that collection.

4. Click **Save**. The associated access policy should directly tie with your collection and all the resources that you have created so far.

Steps to Create a New Index Inside the Collection

Note: If you are using Teamcenter version 2406.0005 or later, this section is not required. If you are using previous versions, complete the following steps to create an index.

1. Navigate to your collection TC_AI_Chat in the OpenSearch Service dashboard and go to **Indexes**.

2. Click **Create Vector Index** and select JSON.

3. Name the index.

4. Enter the following in the JSON text area and click **Create.**

```
{
    "settings": {
        "index.knn": true
    },
    "mappings": {
        "properties": {
            "id": {
                "type": "text",
                "fields": {
                    "keyword": {
                        "type": "keyword", "ignore_above": 256
                    }
                }
            },
            "metadata": {
```

```
                "properties": {
                    "owning_file": {
                        "type": "text",
                        "fields": {
                            "keyword": {
                                "type": "keyword", "ignore_above": 256
                            }
                        }
                    },
                    "owning_object": {
                        "type": "text",
                        "fields": {
                            "keyword": {
                                "type": "keyword", "ignore_above": 256
                            }
                        }
                    },
                    "section_index": {"type": "long"},
                    "last_modified_date": {"type": "date"},
                    "knowledge_bases": {
                        "type": "text",
                        "fields": {
                            "keyword": {
                                "type": "keyword", "ignore_above": 256
                            }
                        }
                    },
                    "owning_user": {
                        "type": "text",
                        "fields": {
                            "keyword": {
                                "type": "keyword", "ignore_above": 256
                            }
                        }
                    },
                    "owning_group": {
                        "type": "text",
                        "fields": {
                            "keyword": {
                                "type": "keyword", "ignore_above": 256
                            }
                        }
                    }
                }
            },
            "text": {
                "type": "text",
                "fields": {
                    "keyword": {
                        "type": "keyword", "ignore_above": 256
                    }
                }
            },
            "vector_field": {
                "type": "knn_vector",
```

```
                    "dimension": 1536,
                    "method": {
                        "engine": "nmslib",
                        "space_type": "cosinesimil",
                        "name": "hnsw",
                        "parameters": {"ef_construction": 512, "m": 16}
                    }
                }
            }
        }
    }
```

Install and Configure Teamcenter AI Services

You must now install Teamcenter AI Microservices and configure the applicable AI features.

For detailed steps regarding the installation of Teamcenter AI Microservices in Deployment Center, please refer to *Teamcenter Installation Using Deployment Center* on Support Center.

For detailed steps regarding the configuration of Teamcenter AI Chat and Natural Language Search, please refer to *Data Indexing and Search Configuration* on Support Center.

References

https://github.com/aws-samples/serverless-rag-demo

https://docs.aws.amazon.com/opensearch-service/latest/developerguide/serverless-getting-started.html