INTERNAL

Deploying Large-Scale Custom AI Models in SAP AI Core by Integrating with Docker Hub

Version: 1.0

Date: <2025-04-14>

Owner: Sriram Rokkam (s.rokkam@sap.com)

Table of contents

1. Background

As AI models grow in complexity and size, deploying them on cloud platforms often encounters limitations—especially size restrictions. SAP AI Core offers a robust platform for operationalizing AI workloads, but direct deployment of custom AI model's may encounter the size limitation. This white paper presents a proven solution that integrates Docker image deployment into SAP AI Core, enabling seamless integration and execution of large models like BERT Topic (≈6.5 GB) as example without hitting capacity bottlenecks.

2. Introduction

This white paper describes a step-by-step approach for deploying a large AI model encapsulated within a Docker image to SAP AI Core. The process provides a scalable, modular, and version-controlled method for running AI workloads.

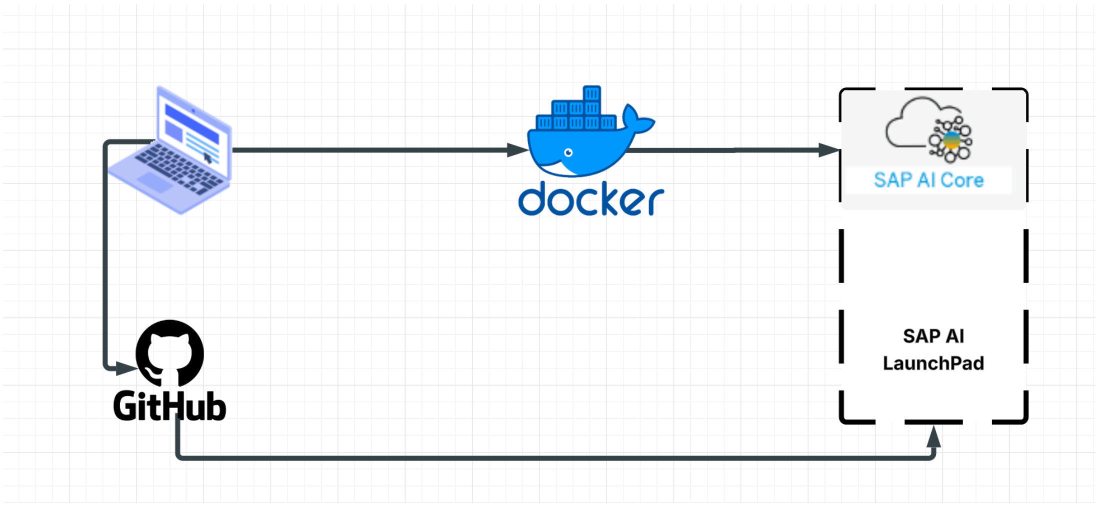
2.1. Key Highlights:

- Eliminate model size restrictions (tested with a 6.5 GB image).
- Enable reproducibility using Docker containers.
- Seamlessly integrate Docker Hub and GitHub into SAP AI Core workflows.

3. Architecture Overview

Description of components below:

• Local Dev Machine:	Code and Dockerfile creation.
• Docker Hub:	Hosting large AI model Docker images.
• GitHub:	Hosting configuration and YAML files.
• SAP AI Core:	Manages workflows and execution.
• SAP AI Launchpad:	UI layer for interaction and monitoring.



4. Deployment Steps

4.1. Pre-Requisites

4.1.1. Docker Hub Account + Access Token

- Create a Docker Hub account and generate a personal access token to allow secure pushing of images from your local machine to Docker Hub.

4.1.2. Docker Desktop Installed in local desktop

- Install Docker Desktop on your system to build and run Docker containers locally before pushing them to the cloud.

4.1.3. GitHub Account + Access Token

- Create a GitHub account and generate a personal access token (PAT). This token is used by SAP AI Core to authenticate and access your private repositories containing pipeline YAML files and related source code

4.1.4. SAP AI Core Access with Admin Role

- Ensure you have access to SAP AI Core with administrator privileges. This is necessary to:
  - Register Git repositories
  - Create Docker registry secrets
  - Manage applications, scenarios, and executions
- Admin access ensures full control over pipeline deployment and configuration in the AI Core landscape.

4.2. Create Python Program

4.2.1. main.py :

- Write your core AI logic (e.g., loading the BERT topic model) inside this Python file. It acts as the entry point for your container.

```
import bertopic as bt
import pandas as pd
print('Log: Please add "bertopic" in "requirements.txt"')
print("BERT Topic which is around 6.5 GB is installed Successfully")
print(f"BERTOPIC is installed")
```

4.2.2. requirements.txt :

- List all the Python dependencies your model or script needs (e.g., bertopic, pandas) for Docker to install during image build.

```
scikit-learn
bertopic
```

4.3. Dockerfile

- Create a file name Dockerfile (without extension) and copy the below given code
- Defines the instructions to build your Docker image — sets the base image, copies source code, installs dependencies, and sets permissions.

```
# Specify which base layers (default dependencies) to use
# You may find more base layers at https://hub.docker.com/
FROM python:3.12
#
# Creates directory within your Docker image
RUN mkdir -p /app/src/
#
# Copies file from your Local system TO path in Docker image
COPY main.py /app/src/
COPY requirements.txt /app/src/
#
# Installs dependencies within you Docker image
RUN pip3 install -r /app/src/requirements.txt
#
# Enable permission to execute anything inside the folder app
RUN chgrp -R 65534 /app && \
    chmod -R 777 /app
```

4.4. Build Docker Image

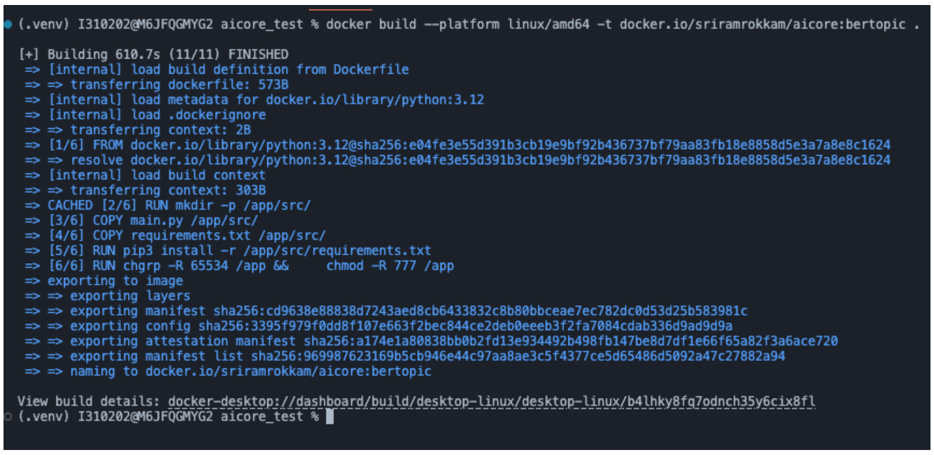
- Run a Docker command to package your app and dependencies into an image. This image can be used to run the model anywhere consistently.

# Windows

docker build -t docker.io/<username>/aicore:bertopic .

# macOS

docker build --platform linux/amd64 -t docker.io/<username>/aicore:bertopic .



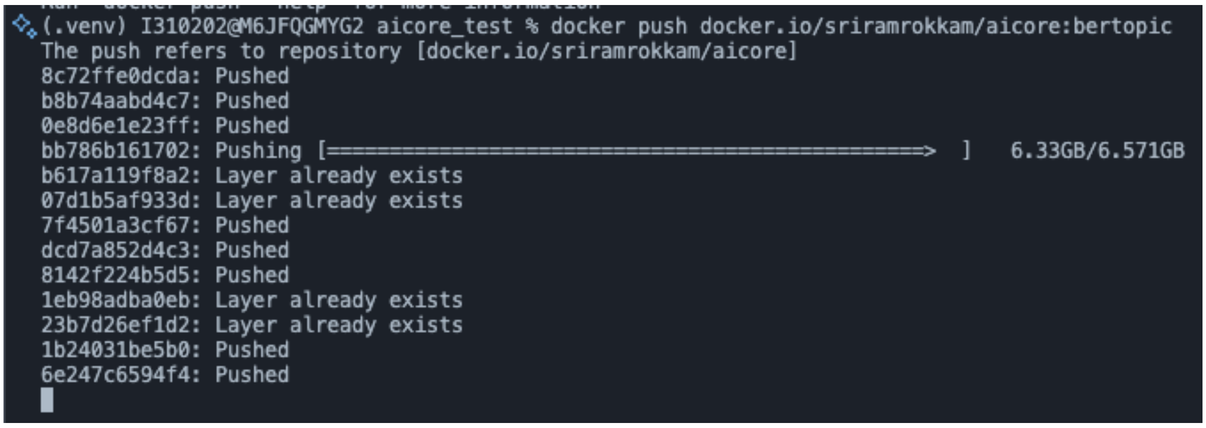
```
(.venv) I310202@M6JFQGMYG2 aicore_test % docker build --platform linux/amd64 -t docker.io/sriramrokkam/aicore:bertopic .

[+] Building 610.7s (11/11) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 573B
 => [internal] load metadata for docker.io/library/python:3.12
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [1/6] FROM docker.io/library/python:3.12@sha256:e04fe3e55d391b3cb19e9bf92b436737bf79aa83fb18e8858d5e3a7a8e8c1624
 => => resolve docker.io/library/python:3.12@sha256:e04fe3e55d391b3cb19e9bf92b436737bf79aa83fb18e8858d5e3a7a8e8c1624
 => [internal] load build context
 => => transferring context: 303B
 => CACHED [2/6] RUN mkdir -p /app/src/
 => [3/6] COPY main.py /app/src/
 => [4/6] COPY requirements.txt /app/src/
 => [5/6] RUN pip3 install -r /app/src/requirements.txt
 => [6/6] RUN chgrp -R 65534 /app &&     chmod -R 777 /app
 => exporting to image
 => => exporting layers
 => => exporting manifest sha256:cd9638e88838d7243aed8cb6433832c8b80bbceae7ec782dc0d53d25b583981c
 => => exporting config sha256:3395f979f0dd8f107e663f2bec844ce2deb0eeeb3f2fa7084cdab336d9ad9d9a
 => => exporting attestation manifest sha256:a174e1a80838bb0b2fd13e934492b498fb147be8d7df1e66f65a82f3a6ace720
 => => exporting manifest list sha256:969987623169b5cb946e44c97aa8ae3c5f4377ce5d65486d5092a47c27882a94
 => => naming to docker.io/sriramrokkam/aicore:bertopic

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/b4lhky8fq7odnch35y6cix8fl
(.venv) I310202@M6JFQGMYG2 aicore_test %
```

Figure 1: Build Docker Image

4.5. Push to Docker Hub

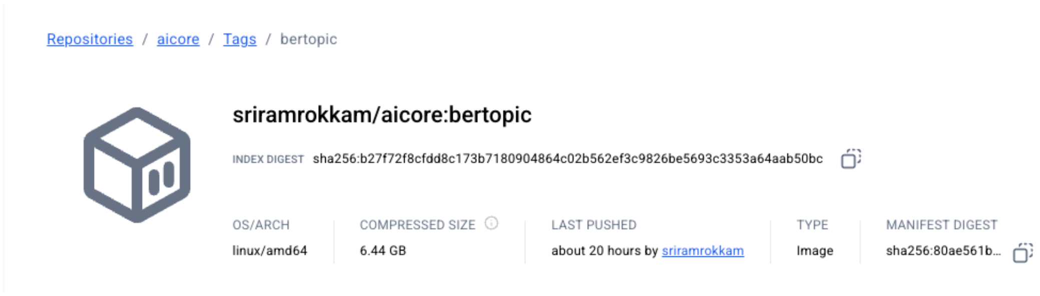
- Upload the built image to your Docker Hub repository so that SAP AI Core can pull and run it during execution.

docker push docker.io/<username>/aicore:bertopic



```
(.venv) I310202@M6JFQGMYG2 aicore_test % docker push docker.io/sriramrokkam/aicore:bertopic
The push refers to repository [docker.io/sriramrokkam/aicore]
8c72ffe0dcda: Pushed
b8b74aabd4c7: Pushed
0e8d6e1e23ff: Pushed
bb786b161702: Pushing [=================================================>  ]   6.33GB/6.571GB
b617a119f8a2: Layer already exists
07d1b5af933d: Layer already exists
7f4501a3cf67: Pushed
dcd7a852d4c3: Pushed
8142f224b5d5: Pushed
1eb98adba0eb: Layer already exists
23b7d26ef1d2: Layer already exists
1b24031be5b0: Pushed
6e247c6594f4: Pushed
```

Figure 2: Push Docker Image


Repositories / aicore / Tags / bertopic

sriramrokkam/aicore:bertopic

INDEX DIGEST sha256:b27f72f8cfdd8c173b7180904864c02b562ef3c9826be5693c3353a64aab50bc

OS/ARCH	COMPRESSED SIZE	LAST PUSHED	TYPE	MANIFEST DIGEST
linux/amd64	6.44 GB	about 20 hours by sriramrokkam	Image	sha256:80ae561b...

Figure 3: Image in Docker Hub

4.6. Create Docker Registry Secret in AI Core

- Add Docker credentials (username and token) as a secret in SAP AI Core so it can authenticate and pull your private image from Docker Hub.

```
{
".dockerconfigjson":
"{\"auths\":{\"https://index.docker.io/v1/\":{\"username\":\"<DOCKER_USERNAME>\",\"password\":\"<ACCESS_TOKEN>\"}}}"
}
```

4.7. Push Code to GitHub

- Upload all relevant files (especially the AI Core YAML pipeline config) to a GitHub repo, which AI Core will use to trigger and manage executions.
    - Create repo

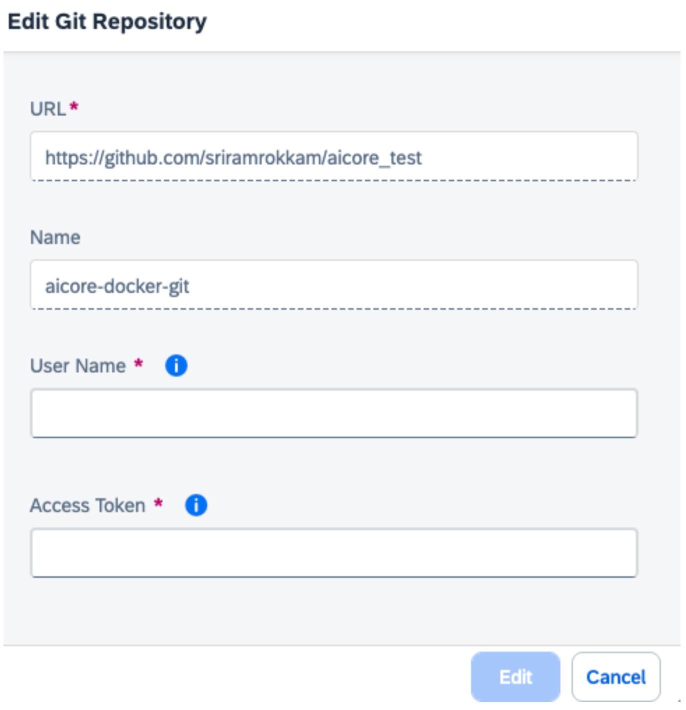
Add yaml/aicore-pipeline.yaml

```
apiVersion: argoproj.io/v1alpha1
kind: WorkflowTemplate
metadata:
  name: aicore-pipeline
  annotations:
    scenarios.ai.sap.com/description: "aicore docker"
    scenarios.ai.sap.com/name: "aicore-pipeline"
    executables.ai.sap.com/description: "aicore docker"
    executables.ai.sap.com/name: "aicore-pipeline"
  labels:
    scenarios.ai.sap.com/id: "aicore-pipeline"
    ai.sap.com/version: "1.0"
```

```yaml
spec:
  imagePullSecrets:
    - name: sriramrokkam
  entrypoint: aicore-pipeline
  templates:
    - name: aicore-pipeline
      steps:
        - - name: mypredictor
            template: mycodeblock1

    - name: mycodeblock1
      container:
        image: docker.io/sriramrokkam/aicore:bertopic
        command: ["/bin/sh", "-c"]
        args:
          - "python /app/src/main.py"
```

4.8. Register Git in SAP AI Core

- Register the GitHub repository in SAP AI Core so it can track the YAML workflow and connect it with Docker execution logic.
    - Go to AI Core > Administration & Add GitHub repository under Repositories
    - Pass the Username and Access key created in 4.1.3

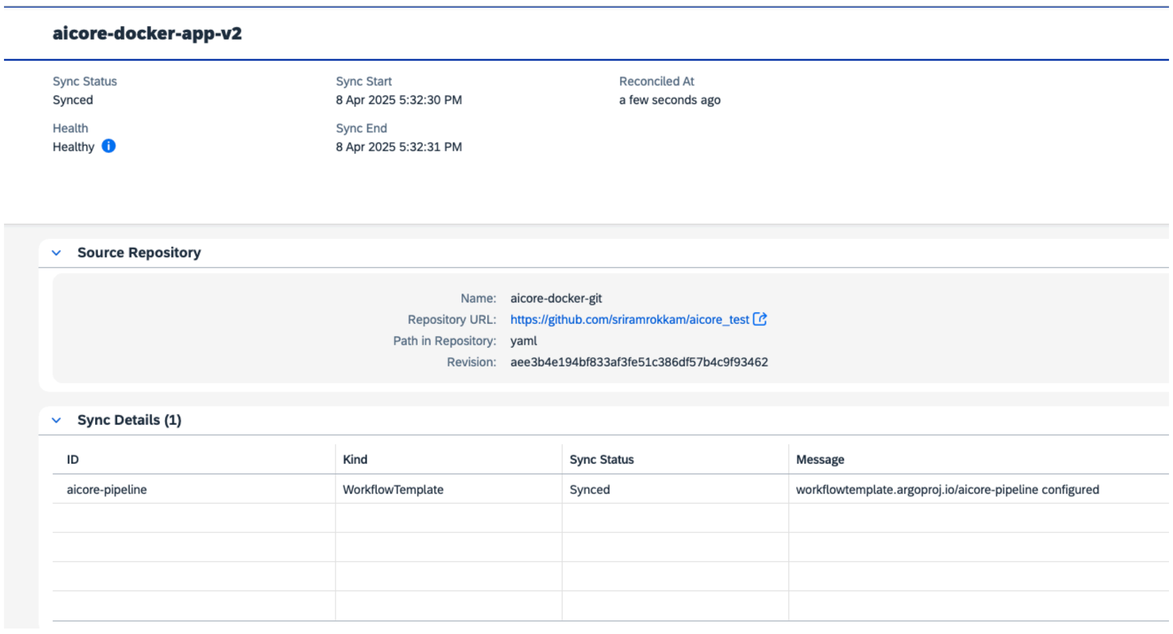


Edit Git Repository

URL*	https://github.com/sriramrokkam/aicore_test
Name	aicore-docker-git
User Name *	
Access Token *	

4.9. Create Application

- Create an AI Core application referencing your GitHub repo. This ties together the Docker image and YAML config for workflow orchestration.
  - Point to the Git repository
  - Validate sync and health status



aicore-docker-app-v2

Sync Status Synced	Sync Start 8 Apr 2025 5:32:30 PM	Reconciled At a few seconds ago
Health Healthy	Sync End 8 Apr 2025 5:32:31 PM	

Source Repository

Name:	aicore-docker-git
Repository URL:	https://github.com/sriramrokkam/aicore_test
Path in Repository:	yaml
Revision:	aee3b4e194bf833af3fe51c386df57b4c9f93462

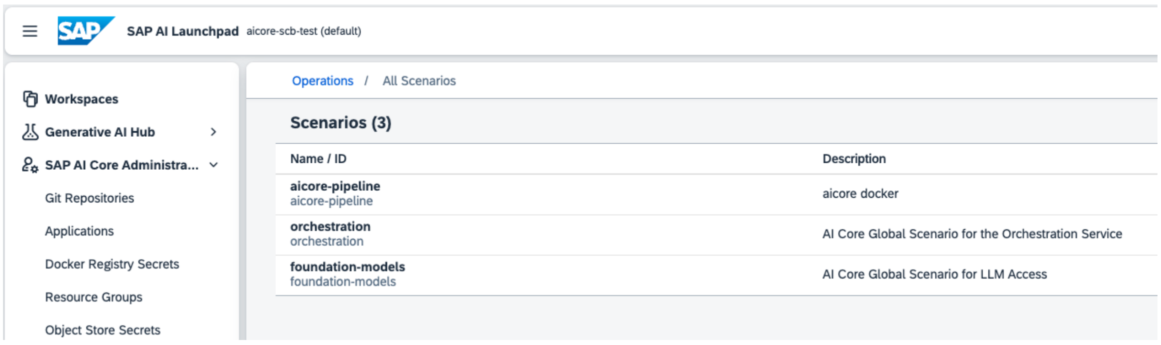
Sync Details (1)

ID	Kind	Sync Status	Message
aicore-pipeline	WorkflowTemplate	Synced	workflowtemplate.argoproj.io/aicore-pipeline configured

Figure 4: Application Sync Status

4.10. Scenario and Executable Creation

- Once the application is set up, SAP AI Core automatically creates a scenario representing the execution blueprint for your pipeline.
  - Auto-created post Application setup
  - Create configuration referencing this scenario



SAP AI Launchpad aicore-scb-test (default)

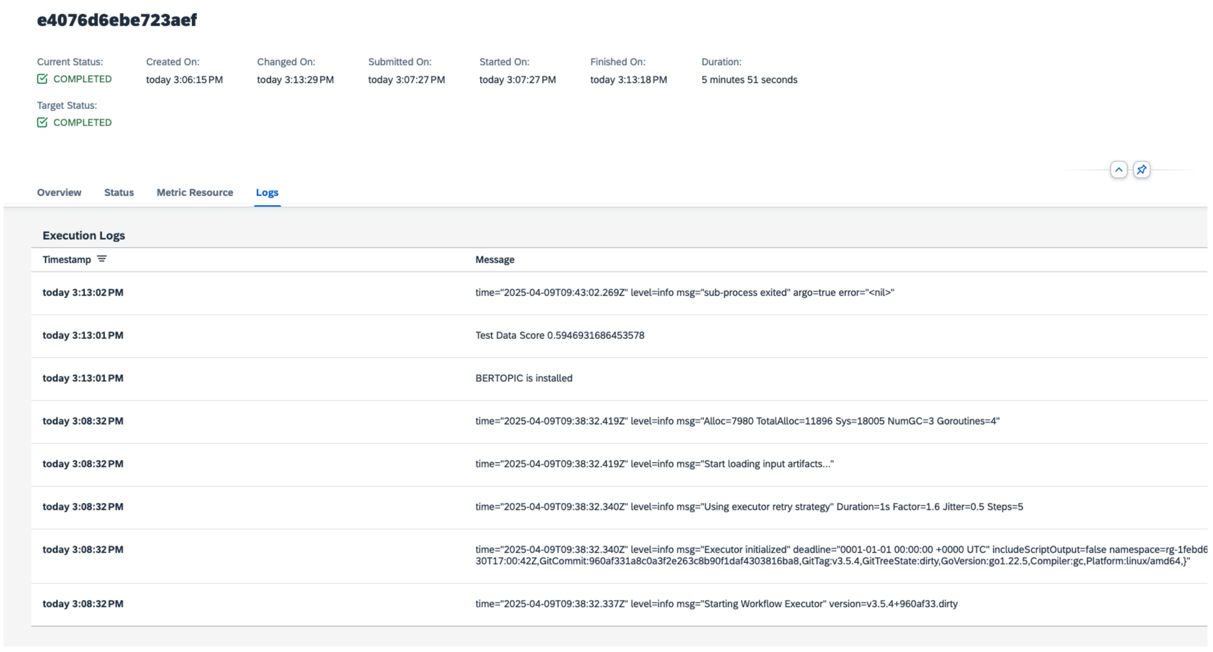
Scenarios (3)

Name / ID	Description
aicore-pipeline aicore-pipeline	aicore docker
orchestration orchestration	AI Core Global Scenario for the Orchestration Service
foundation-models foundation-models	AI Core Global Scenario for LLM Access

Figure 5: Scenario Created Automatically

4.11. Trigger Execution

- Manually trigger the execution of the scenario, which will pull the Docker image and run your model (e.g., BERT topic model) in the AI Core environment.
  - In Launchpad, trigger new execution
  - Monitor for status: completed
  - Expected log output: "BERT Topic... installed successfully"



e4076d6ebe723aef

Current Status:	Created On:	Changed On:	Submitted On:	Started On:	Finished On:	Duration:
✓ COMPLETED	today 3:06:15 PM	today 3:13:29 PM	today 3:07:27 PM	today 3:07:27 PM	today 3:13:18 PM	5 minutes 51 seconds

Target Status: ✓ COMPLETED

Execution Logs

Timestamp	Message
today 3:13:02 PM	time="2025-04-09T09:43:02.269Z" level=info msg="sub-process exited" argo=true error="<nil>"
today 3:13:01 PM	Test Data Score 0.5946931686453578
today 3:13:01 PM	BERTOPIC is installed
today 3:08:32 PM	time="2025-04-09T09:38:32.419Z" level=info msg="Alloc=7980 TotalAlloc=11896 Sys=18005 NumGC=3 Goroutines=4"
today 3:08:32 PM	time="2025-04-09T09:38:32.419Z" level=info msg="Start loading input artifacts..."
today 3:08:32 PM	time="2025-04-09T09:38:32.340Z" level=info msg="Using executor retry strategy" Duration=1s Factor=1.6 Jitter=0.5 Steps=5
today 3:08:32 PM	time="2025-04-09T09:38:32.340Z" level=info msg="Executor initialized" deadline="0001-01-01 00:00:00 +0000 UTC" includeScriptOutput=false namespace=rg-1febd6 30T17:00:42Z,GitCommit:960af331a8c0a3f2e263c8b90f1daf4303816ba8,GitTag:v3.5.4,GitTreeState:dirty,GoVersion:go1.22.5,Compiler:gc,Platform:linux/amd64,}"
today 3:08:32 PM	time="2025-04-09T09:38:32.337Z" level=info msg="Starting Workflow Executor" version=v3.5.4+960af33.dirty

Figure 6: Final Execution with status Completed

5. Troubleshooting

Issue	Root Cause	Resolution
ImagePullBackOff	Incorrect Docker credentials or secret name	Check registry secret format and values
Workflow not found	Git sync failure	Validate Git URL, YAML location, and branch
Model fails to load	Insufficient memory in AI Core container	Switch to higher resource class or optimize Dockerfile
Execution stuck in “Running”	Long install or network issue	Review logs for timeout, retry with minimal build
"main.py not found"	Wrong COPY path in Dockerfile	Check and align Docker paths for main.py

6. Conclusion

This white paper demonstrates a practical and scalable approach to deploying large AI models—such as BERT Topic (6.5 GB)—into SAP AI Core using Docker-based workflows. By leveraging Docker images and external repositories, we eliminate the traditional size limitations imposed by direct model uploads. This method ensures consistent, version-controlled deployments while aligning with enterprise-grade DevOps best practices.

The outlined architecture provides a solid foundation for operationalizing complex AI models in SAP AI Core, paving the way for future enhancements such as automated data pipelines, CI/CD integrations, and real-time inferencing. Whether for proof of concept or production-ready solutions, this approach empowers AI practitioners and SAP developers to deploy robust, containerized AI solutions at scale—securely and efficiently.

7. About us :

Sriram Rokkam is a Global Business AI Architect and a key member of the SAP Cloud ERP Business AI & Intelligent Enterprise Centre of Excellence (CoE). The team consists of a diverse group of techno- functional industry and AI / innovation experts, including solution architects, data scientists, business transformation specialists, and AI process experts.

The team works closely with customers across the globe to drive SAP Business AI adoption & shape intelligent business processes, drive innovation, and deliver scalable AI solutions that create real business value.

8. Connect with Us

We're always open to connecting with fellow professionals and customers exploring AI in the enterprise. Let's collaborate and accelerate the adoption of Business AI together.

👉 Sriram Rokkam - s.rokkam@sap.com
SAP Global Business AI Architect (LinkedIn)

👉 Shantanu Garg - shantanu.garg@sap.com
Global Head of Artificial Intelligence & IE CoE (LinkedIn)