**Week 5: Automating the Social Media Sentiment Analysis System with Azure DevOps**

**Prerequisites:**

1. **Azure DevOps Account**: Ensure you have a project set up in Azure DevOps.
2. **Azure Databricks Workspace**: Access your Azure Databricks workspace where your notebooks and clusters are hosted.
3. **Service Principal or Personal Access Token (PAT) for Azure Databricks**: Generate a PAT in Databricks for authentication.
4. **Databricks CLI Installed and Configured**: Install the Databricks CLI on your local machine or CI agent for pipeline integration.

**Step 1: Set Up the Databricks CLI**

1. **Install Databricks CLI**:

pip install databricks-cli

1. **Configure the Databricks CLI**:

databricks configure --token

You will need the following:

* + **Databricks Host URL** (e.g., https://<databricks-instance>.azuredatabricks.net)
  + **Token**: A Personal Access Token (PAT) generated from Databricks for authentication.

**Step 2: Create an Azure DevOps Pipeline**

1. **Create a YAML Pipeline**:
   * Go to **Pipelines** → **Create Pipeline** in Azure DevOps.
   * Select the repository where your project code is stored.
   * Choose to configure your pipeline using a YAML file.
2. **Add Variables**: In Azure DevOps, navigate to **Pipelines** > **Library** and add the following variables for Databricks configuration:
   * **DATABRICKS\_HOST**: The URL of your Azure Databricks workspace.
   * **DATABRICKS\_TOKEN**: The Personal Access Token.

**Step 3: Azure DevOps YAML Pipeline Example**

Here’s an example of the azure-pipelines.yml file for automating the deployment and execution of the social media sentiment analysis pipeline:

trigger:

branches:

include:

- main

pool:

vmImage: 'ubuntu-latest'

variables:

DATABRICKS\_HOST: 'https://<databricks-instance>.azuredatabricks.net'

DATABRICKS\_TOKEN: $(databricksToken)

steps:

# Step 1: Install Python and Databricks CLI

- task: UsePythonVersion@0

inputs:

versionSpec: '3.x'

addToPath: true

- script: |

pip install databricks-cli

displayName: 'Install Databricks CLI'

# Step 2: Configure Databricks CLI

- script: |

databricks configure --host $(DATABRICKS\_HOST) --token $(DATABRICKS\_TOKEN)

displayName: 'Configure Databricks CLI'

env:

DATABRICKS\_HOST: $(DATABRICKS\_HOST)

DATABRICKS\_TOKEN: $(DATABRICKS\_TOKEN)

# Step 3: Upload Notebook to Databricks Workspace

- script: |

databricks workspace import ./notebooks/SentimentAnalysis\_notebook.py /Shared/SentimentAnalysis\_notebook -l PYTHON

displayName: 'Upload Notebook to Databricks Workspace'

# Step 4: Run Databricks Notebook

- script: |

JOB\_ID=$(databricks runs submit --json-file run\_config.json | jq -r '.run\_id')

echo "Job ID: $JOB\_ID"

databricks runs wait --run-id $JOB\_ID

displayName: 'Run Databricks Notebook'

**Explanation of the Pipeline:**

1. **Trigger**: The pipeline automatically triggers when changes are pushed to the main branch.
2. **Pool**: The pipeline uses the latest Ubuntu image for the build environment.
3. **Install Python and Databricks CLI**: Python and the Databricks CLI are installed to interact with the Databricks workspace.
4. **Configure Databricks CLI**: The Databricks CLI is configured using the environment variables for the Databricks host and token.
5. **Upload Notebook**: The SentimentAnalysis\_notebook.py is uploaded to the Databricks workspace at /Shared/ directory.
6. **Run Notebook**: The pipeline submits the notebook for execution using the JSON configuration file (run\_config.json).

**Step 4: Running Databricks Notebook with a JSON Config File**

A JSON configuration file (run\_config.json) defines the notebook parameters and cluster settings for running the notebook. Below is an example configuration file:

json

{

"run\_name": "Sentiment Analysis Notebook Run",

"new\_cluster": {

"spark\_version": "10.4.x-scala2.12",

"node\_type\_id": "Standard\_DS3\_v2",

"num\_workers": 2

},

"notebook\_task": {

"notebook\_path": "/Shared/SentimentAnalysis\_notebook",

"base\_parameters": {

"param1": "value1",

"param2": "value2"

}

}

}

* **run\_name**: The name of the notebook run.
* **new\_cluster**: The configuration of the Databricks cluster (e.g., Spark version, node type, and number of workers).
* **notebook\_task**: The path to the notebook in the Databricks workspace and any parameters required for the notebook execution.

**Summary of the Pipeline:**

1. **Step 1**: The pipeline installs Python and the Databricks CLI for interacting with the Databricks environment.
2. **Step 2**: The Databricks CLI is configured using the host URL and PAT for authentication.
3. **Step 3**: The social media sentiment analysis notebook (SentimentAnalysis\_notebook.py) is uploaded to the Databricks workspace.
4. **Step 4**: The uploaded notebook is executed using the configuration specified in run\_config.json, which includes cluster settings and parameters.

**Key Points:**

* **Databricks CLI**: Used for uploading notebooks and running jobs in Databricks.
* **Azure DevOps Variables**: Store sensitive information like tokens in Azure DevOps variable groups or secrets.
* **Run Configuration**: The JSON file (run\_config.json) specifies the cluster configuration and notebook execution parameters.