# Step 3. Create Auto pool - Working Solution

- task: AzureCLI@2

displayName: 'Create Azure Batch Job with Auto Pool'

inputs:

azureSubscription: 'CCDC-DEV-UAT-SPN'

scriptType: 'batch' # Use batch script like Classic does internally

scriptLocation: 'inlineScript'

failOnStandardError: false

inlineScript: |

@echo off

echo Logging into Batch account...

call az batch account login --name $(BatchAccount) --resource-group $(batchResourceGroup)

if %ERRORLEVEL% NEQ 0 (

echo Failed to login to batch account

exit /b 1

)

echo Successfully logged in to batch account

REM Create job JSON file

echo Creating job configuration...

(

echo {

echo "id": "$(appId)-$(Build.BuildId)-%RANDOM%",

echo "poolInfo": {

echo "autoPoolSpecification": {

echo "autoPoolIdPrefix": "autopool",

echo "poolLifetimeOption": "job",

echo "keepAlive": false,

echo "pool": {

echo "vmSize": "$(vmSize)",

echo "virtualMachineConfiguration": {

echo "imageReference": {

echo "publisher": "canonical",

echo "offer": "0001-com-ubuntu-server-focal",

echo "sku": "20\_04-lts",

echo "version": "latest"

echo },

echo "nodeAgentSkuId": "batch.node.ubuntu 20.04"

echo },

echo "targetDedicatedNodes": 1

echo }

echo }

echo }

echo }

) > job.json

echo Creating batch job...

call az batch job create --json-file job.json

if %ERRORLEVEL% NEQ 0 (

echo Job creation failed

type job.json

del job.json

exit /b 1

)

del job.json

echo Job created successfully with auto-pool

Medho 3:

# Step 3. Create Auto pool using REST API

- task: AzureCLI@2

displayName: 'Create Azure Batch Job with Auto Pool'

inputs:

azureSubscription: 'CCDC-DEV-UAT-SPN'

scriptType: 'bash'

scriptLocation: 'inlineScript'

addSpnToEnvironment: true

inlineScript: |

set -e

echo "Getting access token for Batch service..."

ACCESS\_TOKEN=$(az account get-access-token --resource https://batch.core.windows.net/ --query accessToken -o tsv)

echo "Getting Batch account details..."

BATCH\_URL=$(az batch account show \

--name $(BatchAccount) \

--resource-group $(batchResourceGroup) \

--query accountEndpoint -o tsv)

# Remove the trailing slash if present

BATCH\_URL=${BATCH\_URL%/}

# Create job ID

JOB\_ID="$(appId)-$(Build.BuildId)-$RANDOM"

echo "Creating job with ID: $JOB\_ID"

# Create job JSON

cat > job.json <<EOF

{

"id": "$JOB\_ID",

"poolInfo": {

"autoPoolSpecification": {

"autoPoolIdPrefix": "autopool",

"poolLifetimeOption": "job",

"keepAlive": false,

"pool": {

"vmSize": "$(vmSize)",

"virtualMachineConfiguration": {

"imageReference": {

"publisher": "canonical",

"offer": "0001-com-ubuntu-server-focal",

"sku": "20\_04-lts",

"version": "latest"

},

"nodeAgentSkuId": "batch.node.ubuntu 20.04"

},

"targetDedicatedNodes": 1,

"taskSlotsPerNode": 1

}

}

},

"priority": 0,

"constraints": {

"maxWallClockTime": "PT72H",

"maxTaskRetryCount": 0

},

"onAllTasksComplete": "terminatejob"

}

EOF

echo "Job JSON content:"

cat job.json

# Make REST API call to create job

echo "Calling Batch REST API..."

RESPONSE=$(curl -s -w "\n%{http\_code}" -X POST \

-H "Authorization: Bearer $ACCESS\_TOKEN" \

-H "Content-Type: application/json; odata=minimalmetadata" \

-H "Accept: application/json" \

-d @job.json \

"https://${BATCH\_URL}/jobs?api-version=2024-02-01.19.0")

# Extract status code

HTTP\_CODE=$(echo "$RESPONSE" | tail -n1)

BODY=$(echo "$RESPONSE" | head -n-1)

echo "HTTP Status Code: $HTTP\_CODE"

echo "Response Body: $BODY"

# Clean up

rm -f job.json

# Check if successful (201 Created)

if [ "$HTTP\_CODE" = "201" ]; then

echo "Job $JOB\_ID created successfully with auto-pool!"

else

echo "Failed to create job. Status: $HTTP\_CODE"

echo "Error details: $BODY"

exit 1

fi