Perfect! I understand. You want to create the output container in the "Create Job & Task" step and keep the "Upload Package" step unchanged. Here's the complete solution:

```powershell

# Create Job & Task step - Complete script

Write-Host ">>> Step 5: Job & Task"

# Variables

$job = "job-$(Build.BuildId)"

$task = "task-$(Build.BuildId)"

$endpoint = "https://$(BatchAccount).$(Region).batch.azure.com"

$outputContainer = "$(container)-output" # ccdcbatchjobs-output

# Step 1: Create output container

Write-Host "Creating output container..."

az storage container create `

--account-name $(storageAccount) `

--name $outputContainer `

--auth-mode login 2>$null

if ($LASTEXITCODE -eq 0) {

Write-Host "Output container '$outputContainer' created successfully"

} else {

Write-Host "Output container '$outputContainer' already exists (this is fine)"

}

# Step 2: Verify the job exists (from Create auto Pool step)

Write-Host "Verifying job exists..."

az batch job show --job-id $job --account-name $(BatchAccount) --account-endpoint $endpoint 2>$null

if ($LASTEXITCODE -ne 0) {

Write-Error "Job $job not found. Make sure Create auto Pool step succeeded."

exit 1

}

# Step 3: Create and submit the task

Write-Host "Creating task..."

$pkgRef = "[{`"applicationId`":`"$(ApplicationId)`",`"version`":`"$(AppVersionFormat)`"}]"

# Command line that runs Python and uploads output

$commandLine = @"

/bin/bash -c "python `$AZ\_BATCH\_APP\_PACKAGE\_$(ApplicationId)/main.py > output.txt 2>&1 && echo 'Task completed, uploading output...' && az storage blob upload --account-name $(storageAccount) --container-name $outputContainer --name $(Build.BuildId)/$task-output.txt --file ./output.txt --auth-mode login --overwrite"

"@

az batch task create `

--job-id $job `

--task-id $task `

--command-line $commandLine `

--application-package-references $pkgRef `

--account-name $(BatchAccount) `

--account-endpoint $endpoint

if ($LASTEXITCODE -eq 0) {

Write-Host "Task '$task' created successfully!"

Write-Host "Output will be uploaded to:"

Write-Host " Container: $outputContainer"

Write-Host " Blob path: $(Build.BuildId)/$task-output.txt"

Write-Host " Full URL: https://$(storageAccount).blob.core.windows.net/$outputContainer/$(Build.BuildId)/$task-output.txt"

} else {

Write-Error "Failed to create task"

exit 1

}

# Step 4: Monitor task (optional - uncomment if you want to wait for completion)

# Write-Host "Monitoring task execution..."

# $timeout = 300 # 5 minutes

# $elapsed = 0

# while ($elapsed -lt $timeout) {

# $taskState = az batch task show --job-id $job --task-id $task --account-name $(BatchAccount) --account-endpoint $endpoint --query "state" -o tsv

# Write-Host "Task state: $taskState"

# if ($taskState -eq "completed") {

# Write-Host "Task completed!"

# break

# }

# Start-Sleep -Seconds 10

# $elapsed += 10

# }

```

This script:

1. \*\*Creates the output container\*\* (`ccdcbatchjobs-output`) if it doesn't exist

2. \*\*Verifies the job exists\*\* from the previous step

3. \*\*Creates a task\*\* that:

- Runs your Python application

- Captures all output (stdout and stderr) to `output.txt`

- Uploads the output to the new container

4. \*\*Shows where the output will be stored\*\*

The output structure will be:

```

Storage Account: lvbatchdev

├── ccdcbatchjobs/ (Input - unchanged)

│ └── app.zip

└── ccdcbatchjobs-output/ (Output - new container)

└── [BuildId]/

└── task-[BuildId]-output.txt

```

Your "Upload Package" step remains completely unchanged and continues using the `ccdcbatchjobs` container.​​​​​​​​​​​​​​​​