**Documentation CI/CD Pipeline**

# **CI Documentation:**

trigger:

- none

resources:

repositories:

- repository: Item\_Master\_Microservice

type: git

name: Item\_Master\_Microservice

ref: main

variables:

- name: org

value: 'https://dev.azure.com/EA-APAC-ORG-NP-PK'

- name: project

value: ea-apac-proj-pd-orion

- name: organization

value: 'EA-APAC-ORG-NP-PK'

- name: feed

value: 'EA-APAC-ORG-NP-PK'

- name: modules

value: '"parent","utils","base","common","swagger","region","filemanage","auth"'

pool:

name: MRP-PK-NP-AGENTPOOL01

steps:

- script: echo 'CI process to build the commons jars and publish them into Artifacts !'

- checkout: Item\_Master\_Microservice

path: Item\_Master\_Microservice

displayName: 'checkout source'

- task: CopyFiles@2

displayName: 'Taks: copy Dockerfile'

inputs:

SourceFolder: '$(Pipeline.Workspace)/Item\_Master\_Microservice/'

Contents: '\*\*'

TargetFolder: '$(Pipeline.Workspace)/Item\_Master\_Microservice/target'

- task: Docker@1

displayName: 'build docker image'

inputs:

containerregistrytype: 'Azure Container Registry'

azureSubscriptionEndpoint: 'Azure PK MRP NON PROD'

azureContainerRegistry: aenapacacrnpmrp.azurecr.io

command: 'build an image'

dockerFile: '$(Pipeline.Workspace)/Item\_Master\_Microservice/Dockerfile'

imageName: 'aenapacacrnpmrp.azurecr.io/Item\_Master\_Microservice\_1:$(Build.BuildNumber)'

useDefaultContext: false

buildContext: '$(Pipeline.Workspace)/Item\_Master\_Microservice/target'

- task: Docker@1

displayName: 'push image to ACR'

inputs:

azureSubscriptionEndpoint: 'Azure PK MRP NON PROD'

azureContainerRegistry: aenapacacrnpmrp.azurecr.io

command: 'push an image'

imageName: 'aenapacacrnpmrp.azurecr.io/Item\_Master\_Microservice\_1:$(Build.BuildNumber)'

- task: CopyFiles@2

inputs:

Contents: |

item-master-deployment.yml

item-master-service.yml

TargetFolder: '$(build.artifactstagingdirectory)'

- task: PublishBuildArtifacts@1

inputs:

PathtoPublish: '$(Build.ArtifactStagingDirectory)'

ArtifactName: 'drop'

publishLocation: 'Container'

trigger:

- none

It will trigger the pipeline if any change occur in the repos. If we set the trigger to “**main”**  then if any change occurs in the main branch of the repo it will start the pipeline

resources:

repositories:

- repository: Item\_Master\_Microservice

type: git

name: Item\_Master\_Microservice

ref: main

It is the part in which we define what resources are we using like in this case we are using the repository Item microservice what is its type and what branch are we referring to.

pool:

name: MRP-PK-NP-AGENTPOOL01

This is the agent on which the pipeline will run. It is the linux virtual machine that loreal provided in which agent is installed that have access to the ACR and AKS. The pipeline will run on it.

checkout: Item\_Master\_Microservice

path: Item\_Master\_Microservice

displayName: 'checkout source'

- task: CopyFiles@2

displayName: 'Taks: copy Dockerfile'

inputs:

SourceFolder: '$(Pipeline.Workspace)/Item\_Master\_Microservice/'

Contents: '\*\*'

TargetFolder: '$(Pipeline.Workspace)/Item\_Master\_Microservice/target'

This part checkout the the repo and copy all the files in the the repos to the target folder

- task: Docker@1

displayName: 'build docker image'

inputs:

containerregistrytype: 'Azure Container Registry'

azureSubscriptionEndpoint: 'Azure PK MRP NON PROD'

azureContainerRegistry: aenapacacrnpmrp.azurecr.io

command: 'build an image'

dockerFile: '$(Pipeline.Workspace)/Item\_Master\_Microservice/Dockerfile'

imageName: 'aenapacacrnpmrp.azurecr.io/Item\_Master\_Microservice\_1:$(Build.BuildNumber)'

useDefaultContext: false

buildContext: '$(Pipeline.Workspace)/Item\_Master\_Microservice/target'

This is the part which builds the docker file “**azure subscription end point”** is that in which subscription we have created the ACR **“command”** command is to build an image **“docker file”** this is the path to docker file **“image name”** providing the image name. **“use default”** it is made false because we want to provide the **“build context”.** Build context is that part which is containing all our files which we will ne needing to create the docker image

- task: Docker@1

displayName: 'push image to ACR'

inputs:

azureSubscriptionEndpoint: 'Azure PK MRP NON PROD'

azureContainerRegistry: aenapacacrnpmrp.azurecr.io

command: 'push an image'

imageName: 'aenapacacrnpmrp.azurecr.io/Item\_Master\_Microservice\_1:$(Build.BuildNumber)'

This is the part that will push the docker image to the ACR

- task: CopyFiles@2

inputs:

Contents: |

item-master-deployment.yml

item-master-service.yml

TargetFolder: '$(build.artifactstagingdirectory)'

- task: PublishBuildArtifacts@1

inputs:

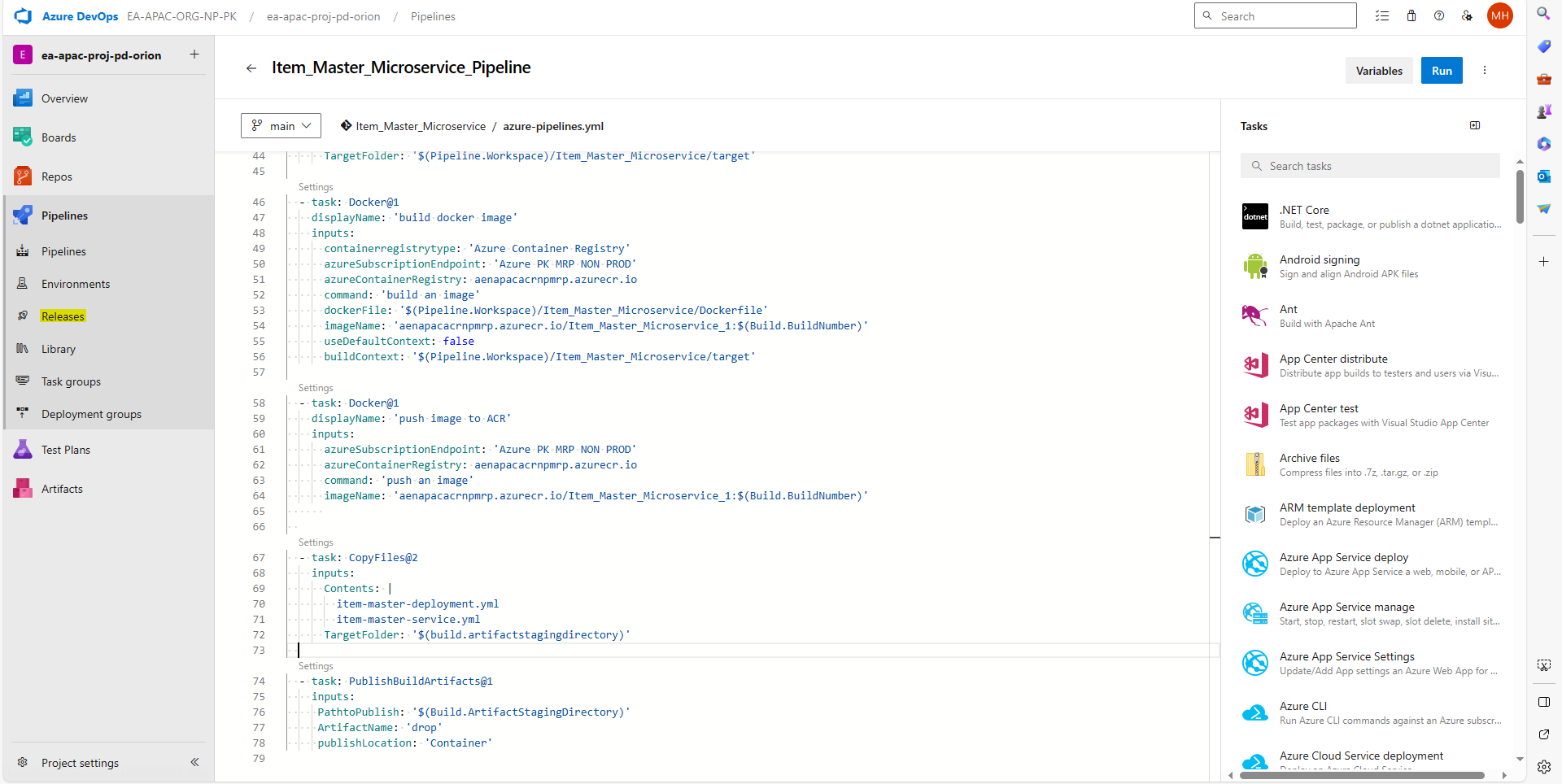
PathtoPublish: '$(Build.ArtifactStagingDirectory)'

ArtifactName: 'drop'

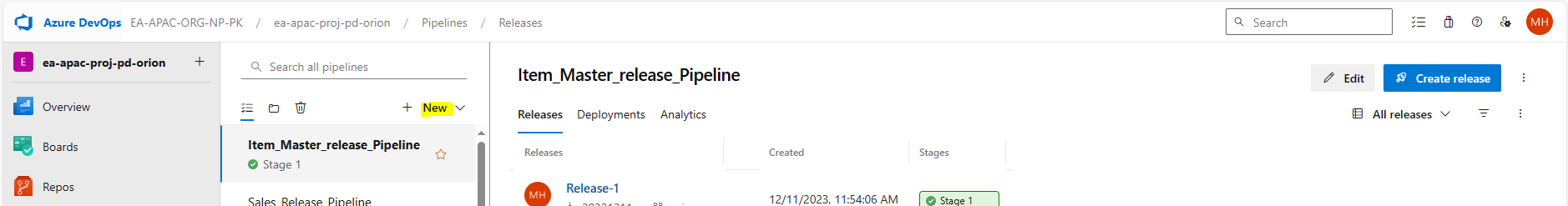
publishLocation: 'Container'

These two tasks we have created for the **“CD”** part it. The first part will copy the manifest files from the repos and then make an artifact that we will be using to in the CD part.

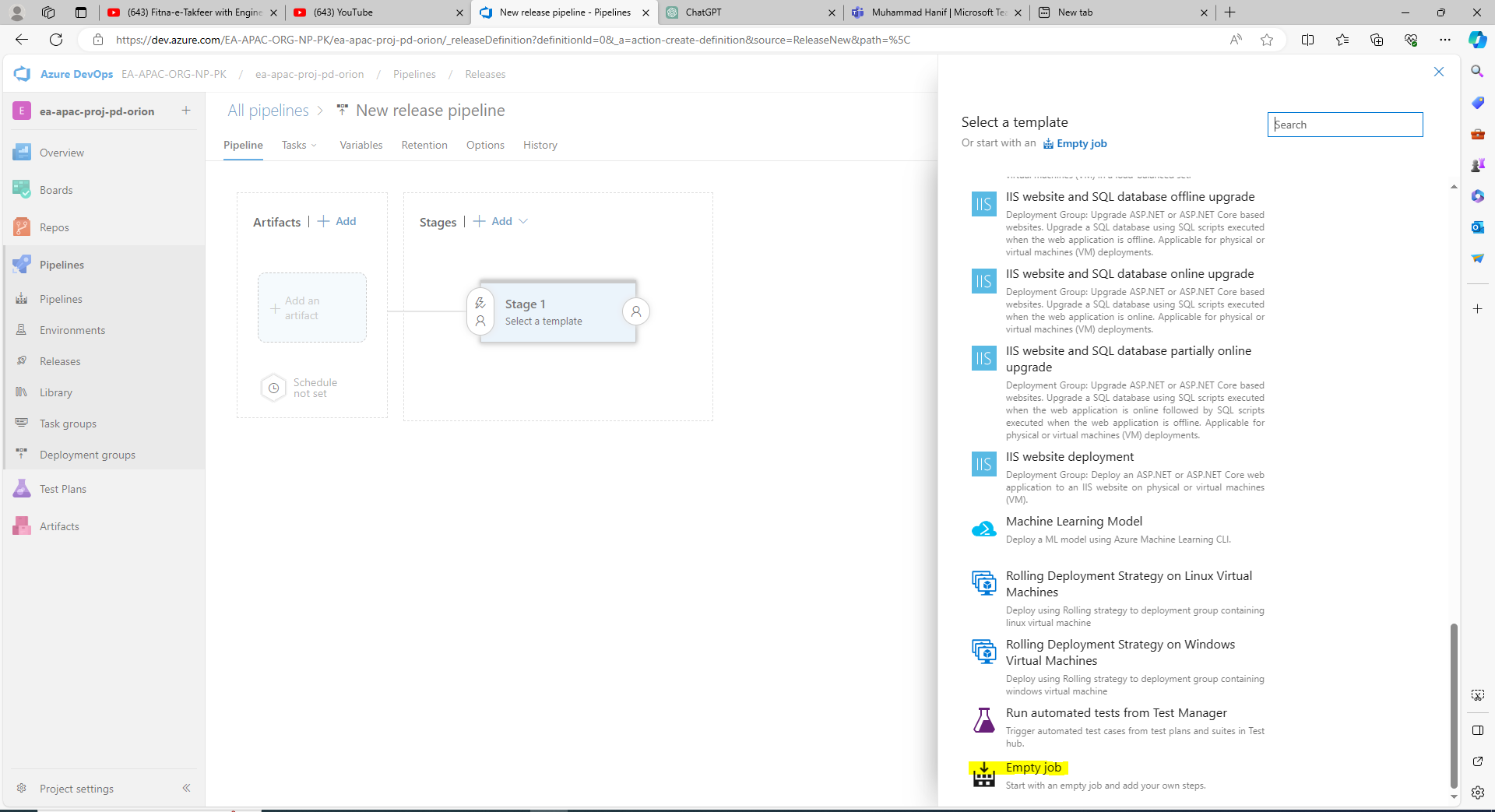
# **CD Documentation:**



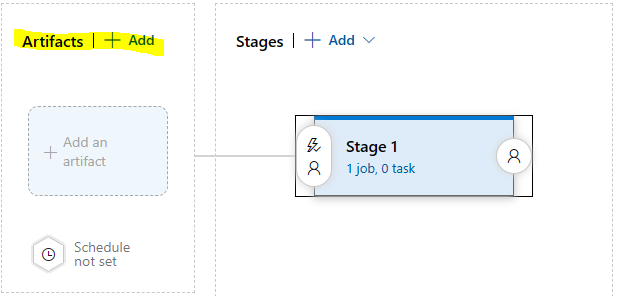
Click on the releases to create the release pipeline that will deploy the docker image to the **“AKS”.**



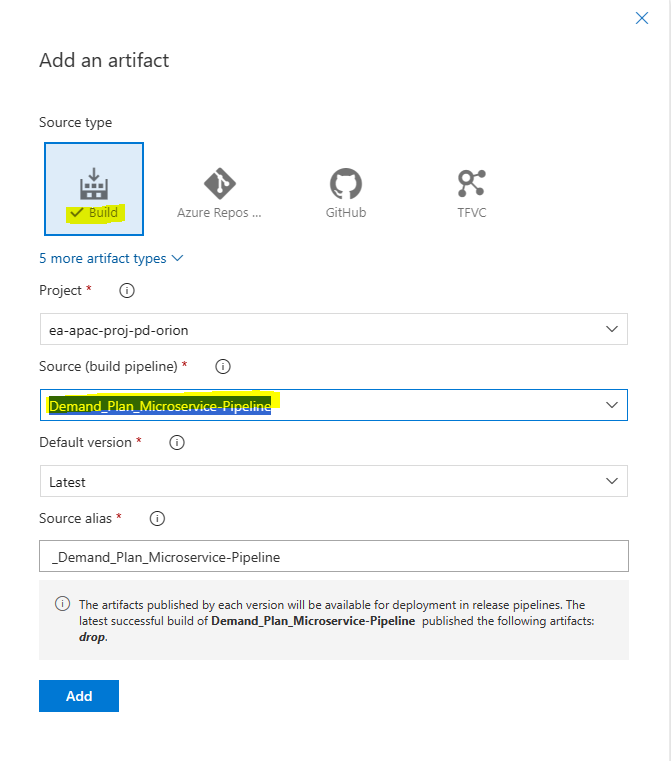
Click on new to create a new release pipeline



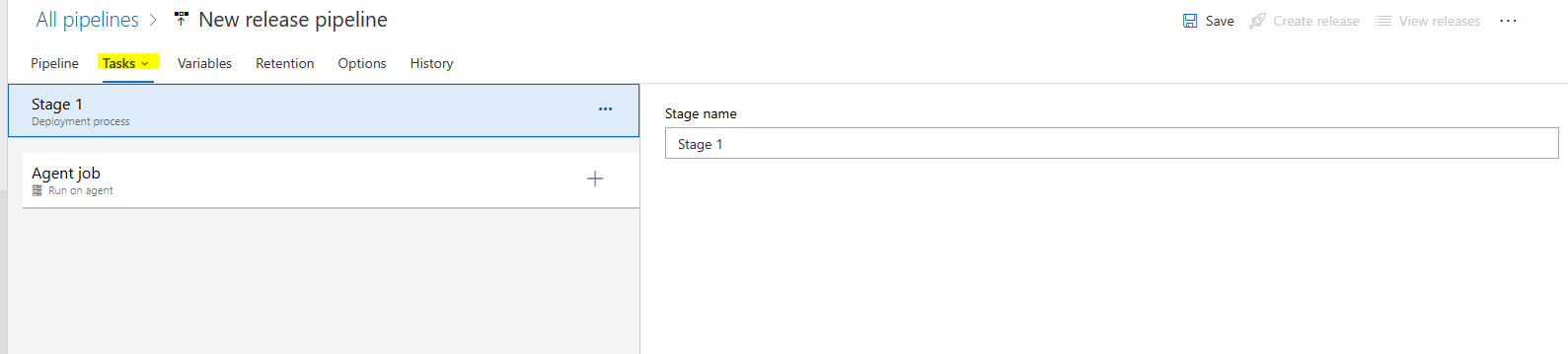
Create an empty job first it will be the last option that you will see.



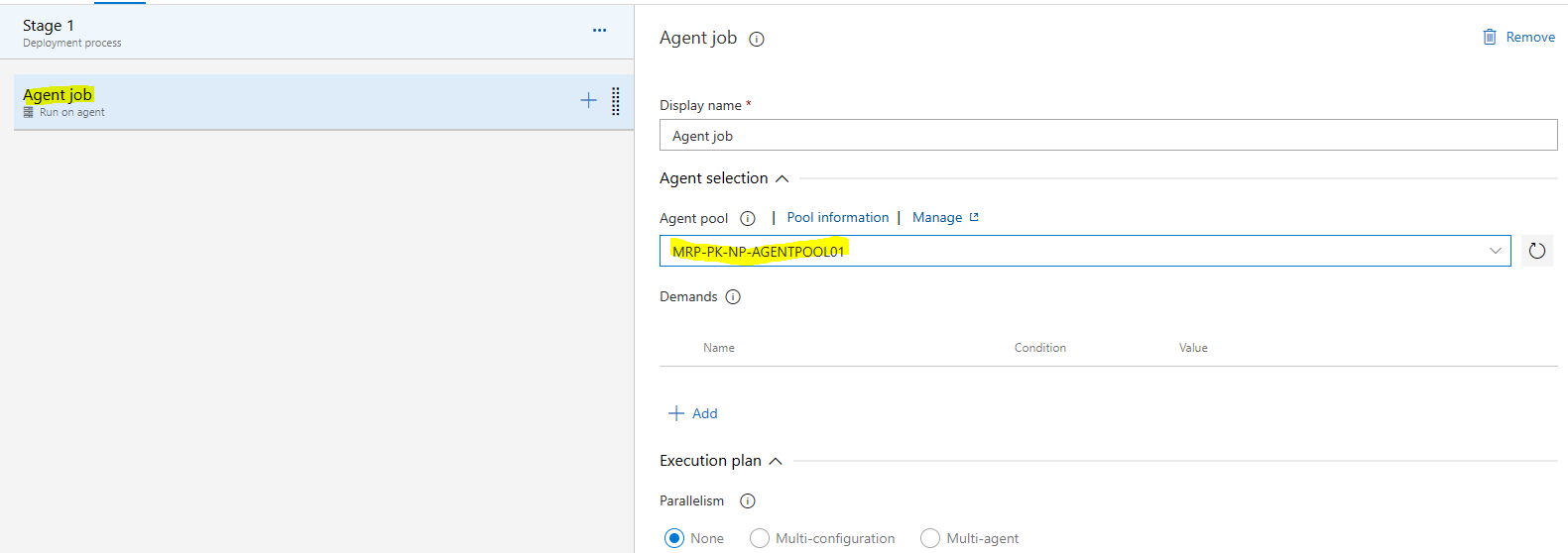
After this click on the artifact that to add the artifact.



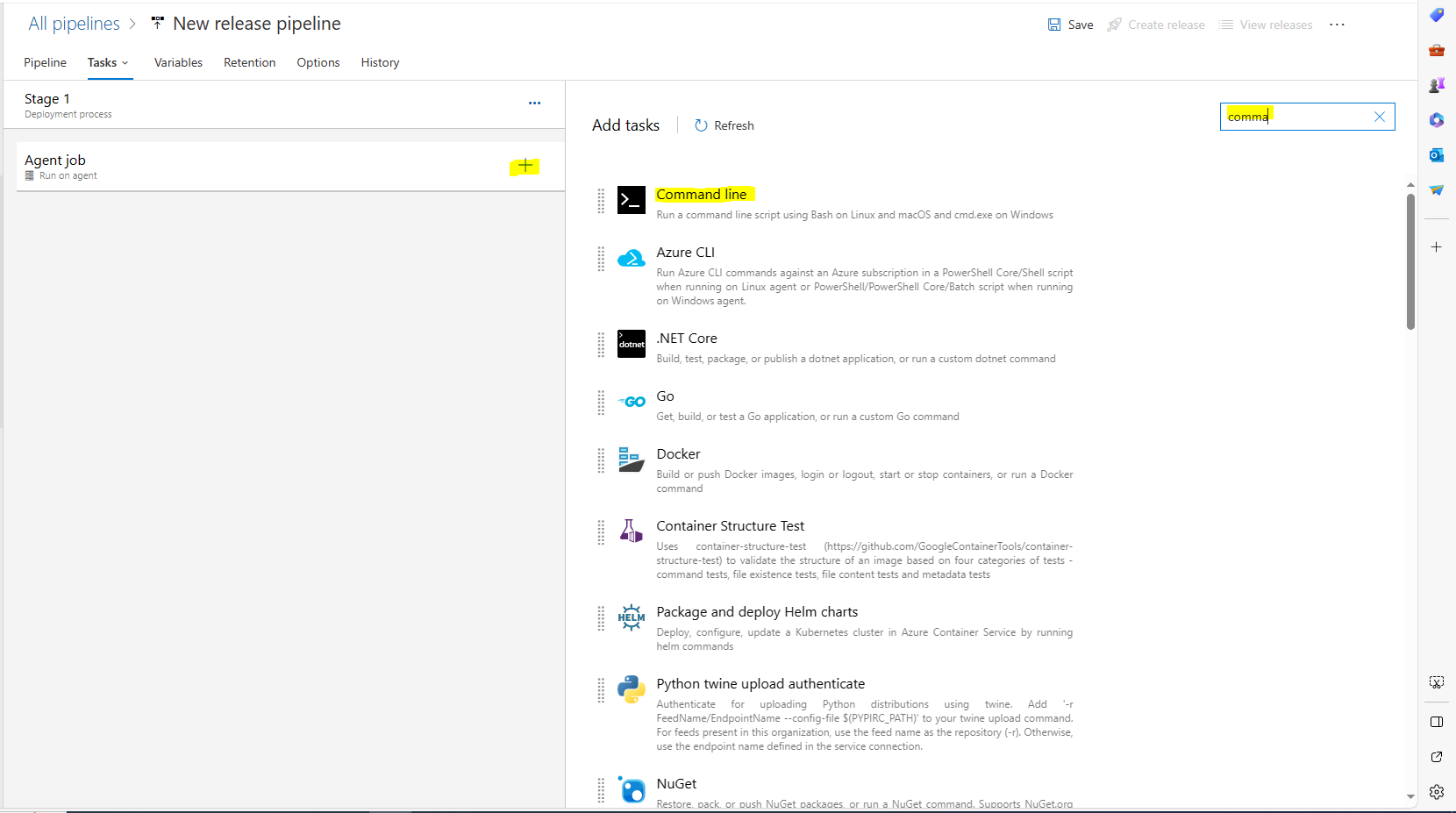
Choose build and in source select the pipeline for which you want to create the release pipeline. And then add it.



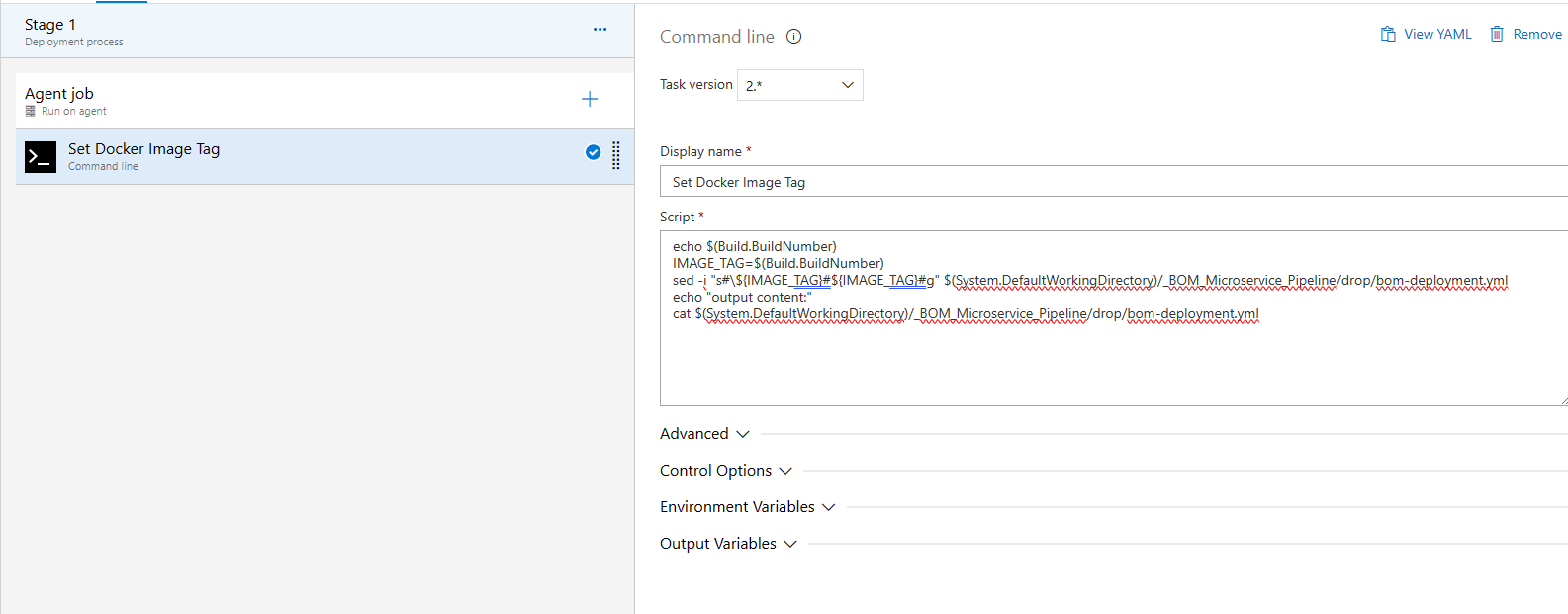
After this go on task



Click on the agent job and select the MRP-PK-NP-AGENTPOOL01 agent.



After this click on plus and add the task the first task is to add the command line line.



echo $(Build.BuildNumber)

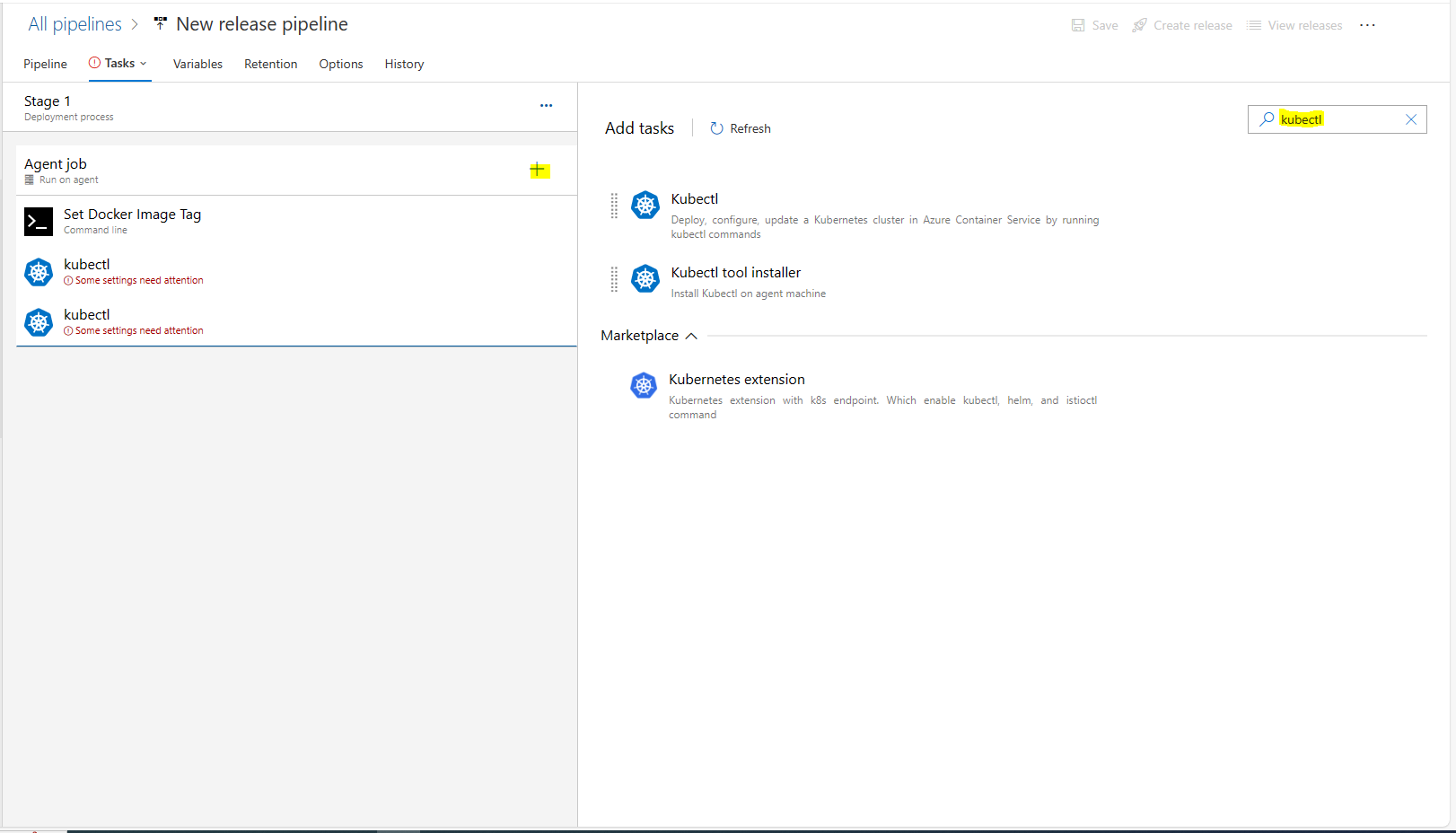
IMAGE\_TAG=$(Build.BuildNumber)

sed -i "s#\${IMAGE\_TAG}#${IMAGE\_TAG}#g" $(System.DefaultWorkingDirectory)/\_BOM\_Microservice\_Pipeline/drop/bom-deployment.yml

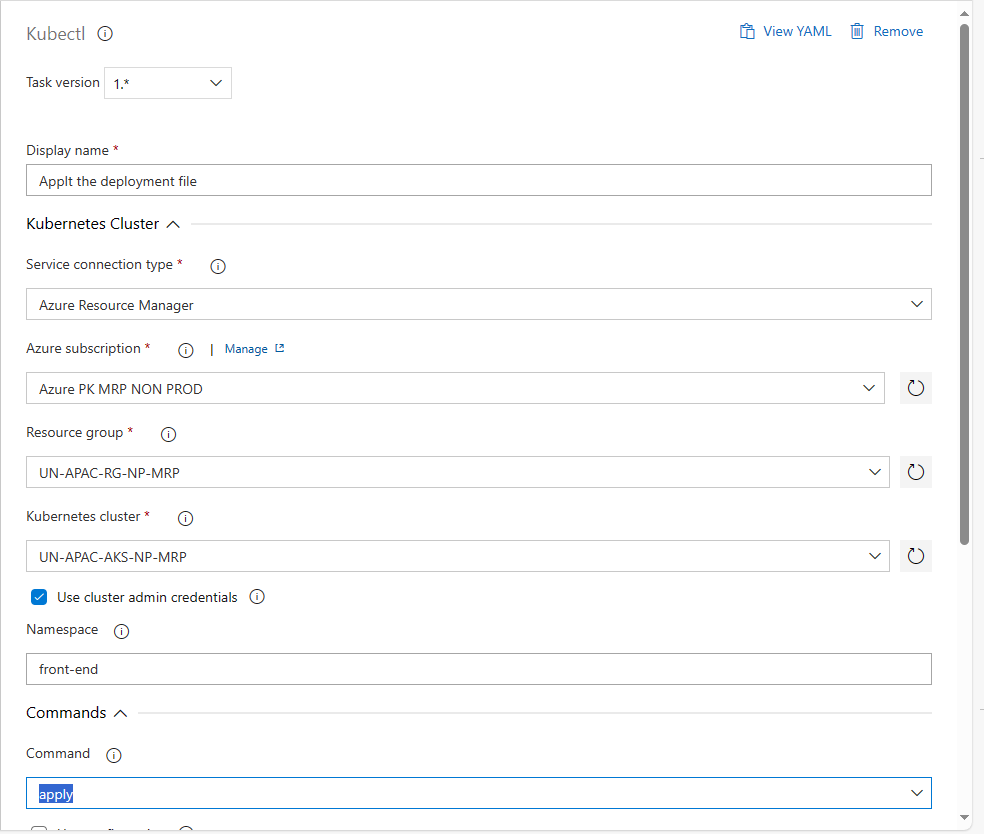
echo "output content:"

cat $(System.DefaultWorkingDirectory)/\_BOM\_Microservice\_Pipeline/drop/bom-deployment.yml

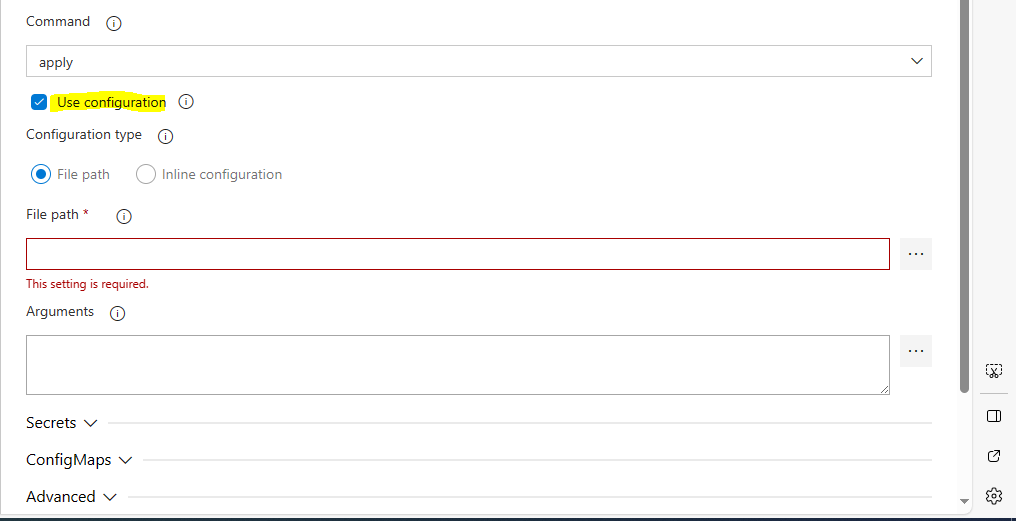
Add the above script in it to set the docker image tag in the manifest file



Again click on the plus sign and add the task **“kubectl”**. Add this task two times in it.

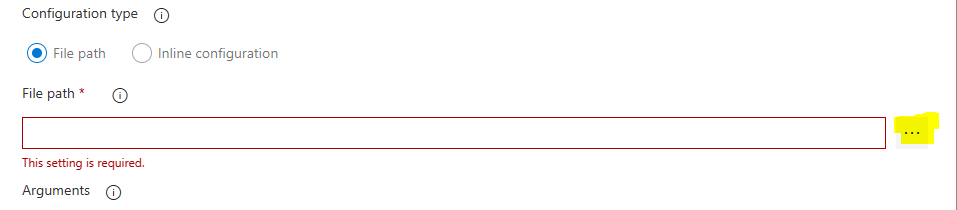


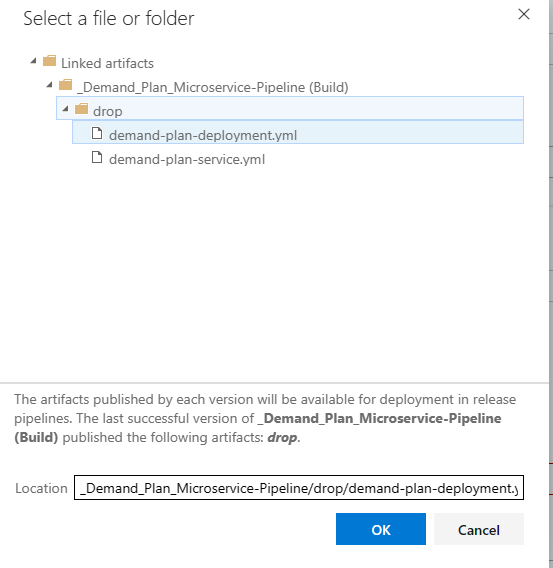
Fill the details



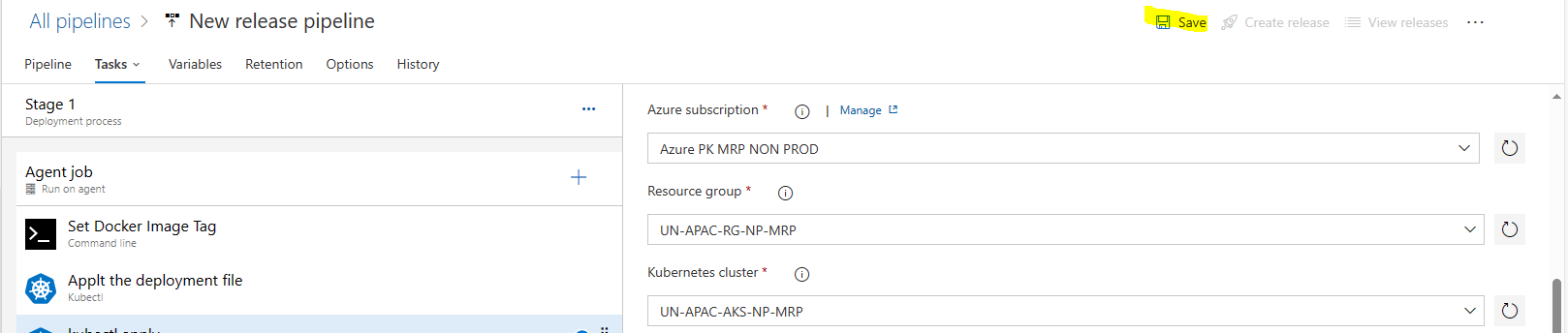
After this check the **”use configuration”**

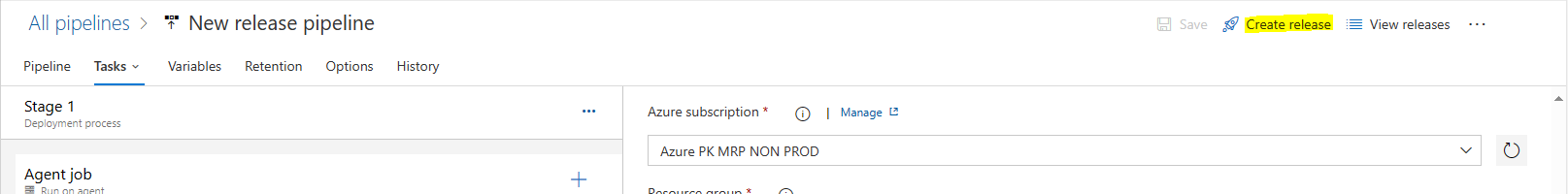
Add the file path of deployment file

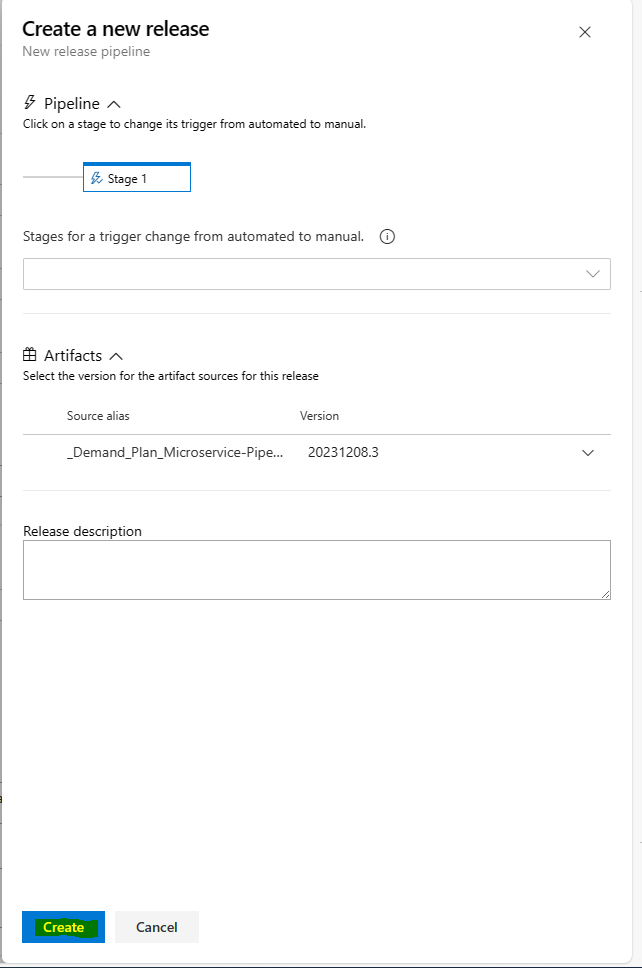


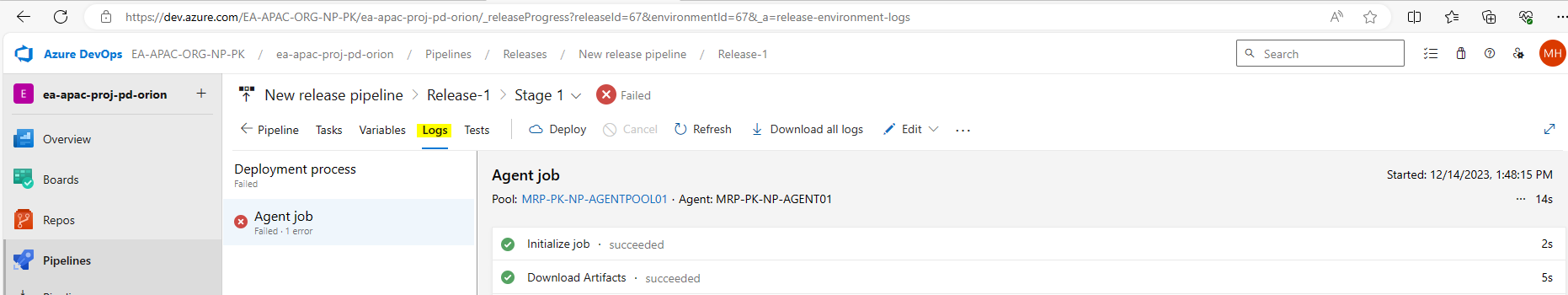


Now go in third task which is again kubectl but this time in the file path you have add the **“service file path”** this is the second file in the **“drop”**  you can see in the above picture.

  
  
After doing everything click on **“Save”**

  
  
After this create release





After this you can check the logs