npm install -g anypoint-cli@latest # Installing Anypoint CLI 3.x

mkdir ~/.anypoint #New Directory

tmpPOMfile=$(mktemp /tmp/tempPOM.XXXXXX) # Create temporary file and storing the reference in a variable

tmpJsonInstancesfile=$(mktemp /tmp/tempInstances.XXXXXX) # Create temporary file and storing the reference in a variable

cp $AGENT\_TEMPDIRECTORY/credentials ~/.anypoint/ # Storing the "Credentials" file inside the .anypoint directory

export ANYPOINT\_ENV="$(ENV)" # Setting the Environment for the current application

unzip -q -c -a "$(Release.PrimaryArtifactSourceAlias)/$(ARTIFACT\_NAME)/target/\*.jar" \*\*/pom.xml > "$tmpPOMfile" # Unzip the artifact to store the POM file in a temporary file

assetId=$(xmllint --xpath '/\*[local-name()="project"]/\*[local-name()="dependencies"]/\*[local-name()="dependency"][\*[local-name()="classifier"]="raml"]/\*[local-name()="artifactId"]/text()' "$tmpPOMfile") # Extracting the RAML Asset ID from the Artifact POM using xmllint

echo "AssetID: ${assetId}"

assetVersion=$(xmllint --xpath '/\*[local-name()="project"]/\*[local-name()="dependencies"]/\*[local-name()="dependency"][\*[local-name()="classifier"]="raml"]/\*[local-name()="version"]/text()' "$tmpPOMfile") # Extracting the RAML Asset Version from the Artifact POM using xmllint

echo "AssetVersion: ${assetVersion}"

anypoint-cli api-mgr api list -o json > "$tmpJsonInstancesfile" # Retrieve the API instances for the specific environment in json and storing them in a temporary file

instanceId=$(jq --arg assetId "$assetId" '[.[] | select(."Asset ID"==$assetId)][0] | ."Instance ID"' "$tmpJsonInstancesfile") # Compare the AssetID from the Artifact and the API instances. If match is found the variable instanceId will store the matching API Manager Instance ID

echo "InstanceID: ${instanceId}"

export filename="$(Release.PrimaryArtifactSourceAlias)/$(ARTIFACT\_NAME)/target/\*.jar" # Storing the Artifact in variable

if [ "$instanceId" = "null" ] || [ "$instanceId" = "" ] # Check if instanceId is null or empty. It will be TRUE only if no matching API instance was found.

then

 echo "No instance avalible for the current Asset. Creating one"

 apiManageRes=$(anypoint-cli api-mgr api manage "$assetId" "$assetVersion" --type raml -m) # Create API Instance with the RAML AssetID and Version and store the response in a variable

 apiID=$(echo -n $apiManageRes | tail -c 8) # Cut Anypoint reponse to only the API ID and store the response in a variable

 echo "$apiID"

 anypoint-cli runtime-mgr cloudhub-application deploy --runtime "$(RUNTIME)" --workers "$(WORKERS)" --workerSize "$(WORKER\_SIZE)" --region "$(REGION)" --property "env:$(ENV)" --property "api.id:$apiID" --property "anypoint.platform.client\_id:$(CLIENT-ID)" --property "anypoint.platform.client\_secret:$(CLIENT-SECRET)" $(APP\_NAME) $filename # Deploy the Artifact on CloudHub with the specifications provided in the Release Variables + Link the API Instance.

else

 echo "The instance for specific application is present. updating the application"

 anypoint-cli api-mgr api change-specification "$instanceId" "$assetVersion" # Update API Instance version with the Artifact RAML version

 anypoint-cli runtime-mgr cloudhub-application modify --runtime "$(RUNTIME)" --workers "$(WORKERS)" --workerSize "$(WORKER\_SIZE)" --region "$(REGION)" $(APP\_NAME) $filename # Redeploy the Artifact on CloudHub with the specifications provided in the Release Variable

fi

rm "$tmpPOMfile" #remove temporary file

rm "$tmpJsonInstancesfile" #remove temporary file