We need to install plugins

- CloudBees AWS Credentials
- Amazon ECR
- Docker Pipeline

we need set up access key and secret key

[] login to j**enkins dashboard** and select **manage jenkins**

[] select system and scroll down in **Global properties** select **Environment variables**

[] add

[] in **name** AWS_ACCESS_KEY_ID (we can give whatever,we use this name in pipeline)

[] in **value** AKIASM6XNBZOR5U2YE7 (I gave access key id what I created in aws)

[] add

[] in **name** AWS_SECRET_ACCESS_KEY (we can give whatever,we use this name in pipeline)

[] in **value** vK6JhW+lToii27wLUiDgnBLJnJzxSIayFD97Ect4 (I gave secret access key id what I created in aws)

[] save

EXAMPLE: (I use like this in code)

[] AWS_ACCESS_KEY_ID="${env.AWS_ACCESS_KEY_ID}"

[] AWS_SECRET_ACCESS_KEY="${env.AWS_SECRET_ACCESS_KEY}"

---

1) Code for to push docker image to ECR

```
pipeline {
   agent any
environment {
    AWS_ACCESS_KEY_ID="${env.AWS_ACCESS_KEY_ID}"
      AWS_SECRET_ACCESS_KEY="${env.AWS_SECRET_ACCESS_KEY}"
      AWS_REGION=('us-east-1')
}
stages {
   stage('Cloning Git') {
steps {
    git credentialsId: 'PAT', url:
'https://github.com/kanchana08/Dockerfile_python.git'
```

```
}
}
   stage('Build Docker Image') {
      steps {
        sh "docker build -t lamda_ply:latest ."
         }
      }
      stage('Publish to ECR') {
         steps {
            script {
                  sh "aws ecr get-login-password --region us-east-1 | docker login --username AWS --password-stdin 165271113309.dkr.ecr.us-east-1.amazonaws.com"
                  sh "docker tag lamda_ply:latest 165271113309.dkr.ecr.us-east-1.amazonaws.com/lamda_ply:latest"
                  sh "docker push 165271113309.dkr.ecr.us-east-1.amazonaws.com/lamda_ply:latest"
                  }
               }
}
}
}
```

==========

2) <u>Code for to push docker image to ECR</u> (I put build function in stage('Publish to ECR'))

```
pipeline {
   agent any
environment {
    AWS_ACCESS_KEY_ID="${env.AWS_ACCESS_KEY_ID}"
      AWS_SECRET_ACCESS_KEY="${env.AWS_SECRET_ACCESS_KEY}"
      AWS_REGION=('us-east-1')
}
stages {
   stage('Cloning Git') {
steps {
    git credentialsId: 'PAT', url:
'https://github.com/kanchana08/Dockerfile_python.git'

}
}

      stage('Publish to ECR') {
         steps {
            script {
                  sh "aws ecr get-login-password --region us-east-1 | docker login --
username AWS --password-stdin 165271113309.dkr.ecr.us-east-1.amazonaws.com"
                  sh"docker build -t lamda_ply ."
                  sh "docker tag lamda_ply:latest 165271113309.dkr.ecr.us-east-
1.amazonaws.com/lamda_ply:latest"
                  sh "docker push 165271113309.dkr.ecr.us-east-
1.amazonaws.com/lamda_ply:latest"
```

```
            }

            }

        }
        }

}
```

==========

Code for s3 bucket :

```
pipeline{
   agent any
   environment {
      AWS_ACCESS_KEY_ID="${env.AWS_ACCESS_KEY_ID}"
      AWS_SECRET_ACCESS_KEY="${env.AWS_SECRET_ACCESS_KEY}"
```

```
    AWS_REGION=('us-east-1')
  }
  stages{
    stage('exm stage'){
      steps {
        script{

        sh"aws s3 ls"
        }
      }
    }
  }

}
```

==========

Deleted previous latest tag images first and then push new images

```
pipeline {
   agent any
environment {
    AWS_ACCESS_KEY_ID="${env.AWS_ACCESS_KEY_ID}"
      AWS_SECRET_ACCESS_KEY="${env.AWS_SECRET_ACCESS_KEY}"
     // AWS_REGION=('us-east-1')
      AWS_REGION="${env.AWS_DEFAULT_REGION}"
}
stages {
   stage('Cloning Git') {
steps {
    git credentialsId: 'PAT', url:
'https://github.com/kanchana08/Dockerfile_python.git'


}
}
   stage('Build Docker Image') {
      steps {
     //   script {
  // dockerImage = docker.build "lamda_ply:latest"


       //   }
         sh "docker build -t lamda_ply:latest ."
      }
      }
      stage('Publish to ECR') {
         steps {
```

```
            script {
                sh "aws ecr get-login-password --region us-east-1 | docker login --
username AWS --password-stdin 165271113309.dkr.ecr.us-east-1.amazonaws.com"
                sh" aws ecr batch-delete-image --repository-name lamda_ply --image-
ids imageTag=latest"
                //sh" aws ecr batch-delete-image --repository-name lamda_ply --image-
ids imageTag=latest"
                 sh"docker tag lamda_ply:latest 165271113309.dkr.ecr.us-east-
1.amazonaws.com/lamda_ply:latest"
                sh "docker push 165271113309.dkr.ecr.us-east-
1.amazonaws.com/lamda_ply:latest"
                }

            }

        }
        }

}
```

==========

Update the ecr image to lambda function

[] select **create lambda** and **select container image** and **paste ecr uri**

[] `sh "aws lambda update-function-code --region ${AWS_DEFAULT_REGION} --function-name ${LAMBDA_FUNCTION_NAME} --image-uri`

```
${AWS_ACCOUNT_ID}.dkr.ecr.${AWS_DEFAULT_REGION}.amazonaws.com/${IMAGE_
REPO_NAME}:${IMAGE_TAG}"
```

[] We need to install plugins

2) AWS lambda plugin

3) Amazon EC2

4) CloudBees AWS Credentials Plugin

```
pipeline {
  agent any
  environment {
    AWS_ACCESS_KEY_ID="${env.AWS_ACCESS_KEY_ID}"
    AWS_SECRET_ACCESS_KEY="${env.AWS_SECRET_ACCESS_KEY}"
    AWS_REGION=('us-east-1')

  }
  stages {
    stage('Cloning Git') {
      steps {
        git credentialsId: 'PATH', url:
'https://github.com/kanchana08/Dockerfile_python.git'

      }

    }
```

```
    stage('Build Docker Image') {
        steps {
            sh "docker build -t playwright_123:latest ."

        }

    }
    stage('Publish to ECR') {
        steps {
            script {

                sh "aws ecr get-login-password --region us-east-1 | docker login --username AWS --password-stdin 909100690382.dkr.ecr.us-east-1.amazonaws.com"
                //sh" aws ecr batch-delete-image --repository-name playwright_123 --image-ids imageTag=latest"

                sh" aws ecr batch-delete-image --repository-name playwright_123 --image-ids imageTag=latest"
                sh "docker tag playwright_123:latest 909100690382.dkr.ecr.us-east-1.amazonaws.com/playwright_123:latest"
                sh "docker push 909100690382.dkr.ecr.us-east-1.amazonaws.com/playwright_123:latest"

            }

        }

    }
```

```
stage('lambda_function'){
   steps{
      script{
sh'aws lambda update-function-code --region us-east-1 --function-name
lambdafunction --image-uri 909100690382.dkr.ecr.us-east-
1.amazonaws.com/playwright_123:latest'


      }
   }
}
}


}
```

==========

With testing lambda functions stage

```
pipeline {
   agent any
   environment {
      AWS_ACCESS_KEY_ID="${env.AWS_ACCESS_KEY_ID}"
      AWS_SECRET_ACCESS_KEY="${env.AWS_SECRET_ACCESS_KEY}"
      AWS_REGION=('us-east-1')


   }
   stages {
      stage('Cloning Git') {
         steps {
```

```
            git credentialsId: 'PATH', url:
'https://github.com/kanchana08/Dockerfile_python.git'


          }


      }
      stage('Build Docker Image') {
         steps {
            sh "docker build -t playwright_123:latest ."


         }


      }
      stage('Publish to ECR') {
         steps {
            script {


                  sh "aws ecr get-login-password --region us-east-1 | docker login --
username AWS --password-stdin 909100690382.dkr.ecr.us-east-1.amazonaws.com"
                  //sh" aws ecr batch-delete-image --repository-name playwright_123 --
image-ids imageTag=latest"


                  sh" aws ecr batch-delete-image --repository-name playwright_123 --
image-ids imageTag=latest"
                  sh "docker tag playwright_123:latest 909100690382.dkr.ecr.us-east-
1.amazonaws.com/playwright_123:latest"
                  sh "docker push 909100690382.dkr.ecr.us-east-
1.amazonaws.com/playwright_123:latest"
```

```
                }


            }


    }


stage('lambda_function'){
  steps{
    script{
sh'aws lambda update-function-code --region us-east-1 --function-name
lambdafunction --image-uri 909100690382.dkr.ecr.us-east-
1.amazonaws.com/playwright_123:latest'


    }
  }
}




stage('lambda_test_function'){
  steps{
    sh' aws lambda invoke --function-name lambdafunction out --log-type Tail'


  }
}
   }


}
```

==========

With testing lambda functions stage

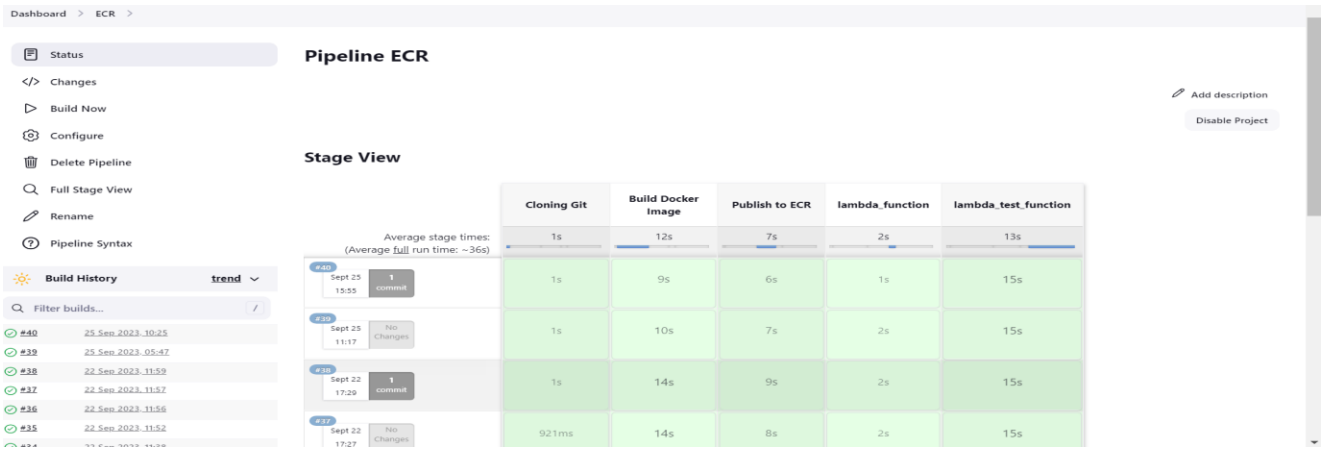
```
pipeline {
   agent any
   environment {
      AWS_ACCESS_KEY_ID="${env.AWS_ACCESS_KEY_ID}"
      AWS_SECRET_ACCESS_KEY="${env.AWS_SECRET_ACCESS_KEY}"
      AWS_REGION=('us-east-1')

   }
   stages {
      stage('Cloning Git') {
         steps {
            git credentialsId: 'PATH', url:
'https://github.com/kanchana08/Dockerfile_python.git'

         }

      }
      stage('Build Docker Image') {
         steps {
            sh "docker build -t playwright_123:latest ."
```

```
        }


    }
    stage('Publish to ECR') {
        steps {
            script {


                sh "aws ecr get-login-password --region us-east-1 | docker login --username AWS --password-stdin 909100690382.dkr.ecr.us-east-1.amazonaws.com"
                //sh" aws ecr batch-delete-image --repository-name playwright_123 --image-ids imageTag=latest"


                sh" aws ecr batch-delete-image --repository-name playwright_123 --image-ids imageTag=latest"
                sh "docker tag playwright_123:latest 909100690382.dkr.ecr.us-east-1.amazonaws.com/playwright_123:latest"
                sh "docker push 909100690382.dkr.ecr.us-east-1.amazonaws.com/playwright_123:latest"


                }


            }


    }


stage('lambda_function'){
  steps{
    script{
```

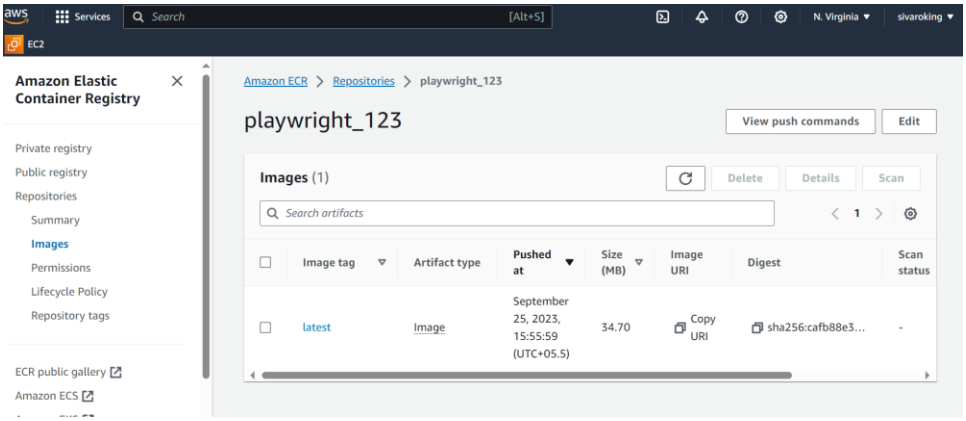
```
sh'aws lambda update-function-code --region us-east-1 --function-name
lambdafunction --image-uri 909100690382.dkr.ecr.us-east-
1.amazonaws.com/playwright_123:latest'


      }
   }
}


stage('lambda_test_function'){
   steps{
      sh' aws lambda invoke --function-name lambdafunction out --log-type Tail'


   }
}
    }


}
```

After testing the lambda functions, AWS CloudWatch log groups created automatically.

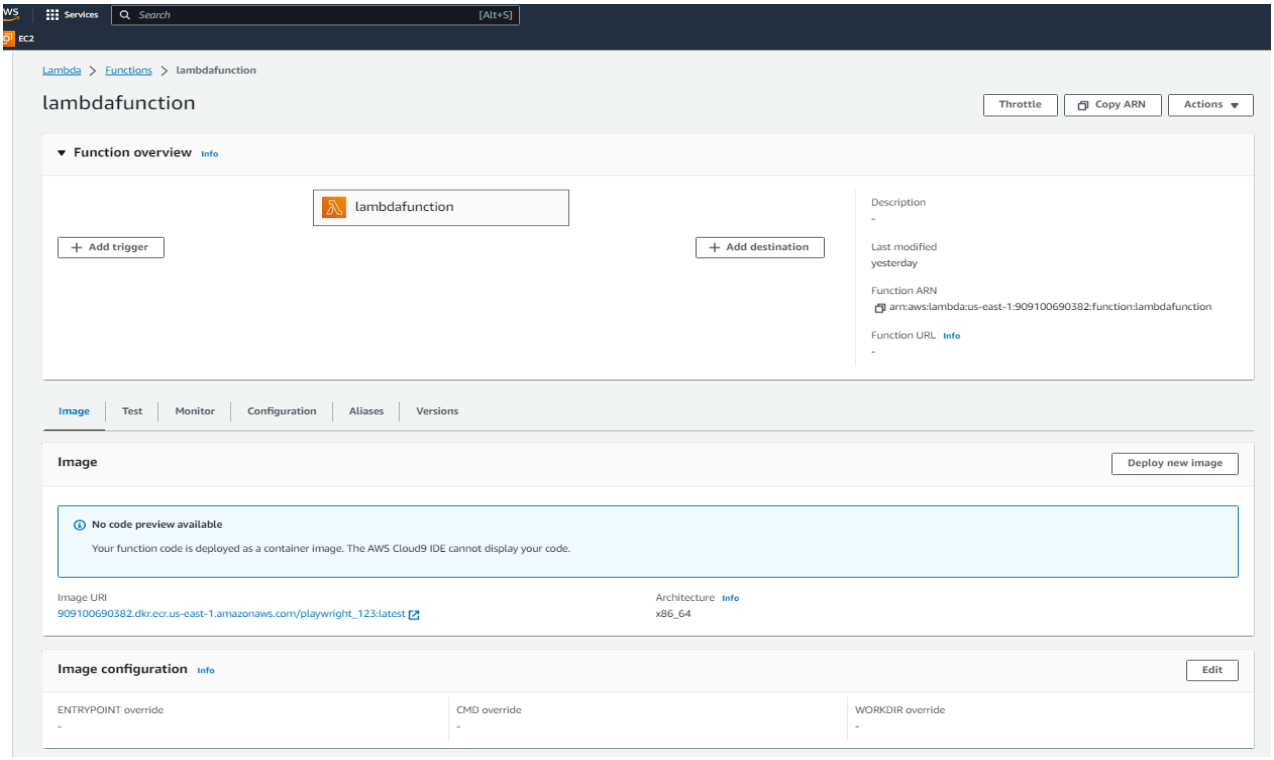
Jenkins output:


Dashboard > ECR >
Status
Changes
Build Now
Configure
Delete Pipeline
Full Stage View
Rename
Pipeline Syntax
Build History
trend
Filter builds...
#40 25 Sep 2023, 10:25
#39 25 Sep 2023, 05:47
#38 22 Sep 2023, 11:59
#37 22 Sep 2023, 11:57
#36 22 Sep 2023, 11:56
#35 22 Sep 2023, 11:52
Pipeline ECR
Add description
Disable Project
Stage View
Cloning Git
Build Docker Image
Publish to ECR
lambda_function
lambda_test_function
Average stage times:
(Average full run time: ~36s)
1s
12s
7s
2s
13s
#40 Sept 25 15:55 1 commit 1s 9s 6s 1s 15s
#39 Sept 25 11:17 No Changes 1s 10s 7s 2s 15s
#38 Sept 22 17:29 1 commit 1s 14s 9s 2s 15s
#37 Sept 22 17:27 No Changes 921ms 14s 8s 2s 15s

ECR image


aws
Services
Search
[Alt+S]
N. Virginia
sivaroking
EC2
Amazon Elastic Container Registry
Private registry
Public registry
Repositories
Summary
Images
Permissions
Lifecycle Policy
Repository tags
ECR public gallery
Amazon ECS
Amazon ECR > Repositories > playwright_123
playwright_123
View push commands
Edit
Images (1)
Delete
Details
Scan
Search artifacts
Image tag
Artifact type
Pushed at
Size (MB)
Image URI
Digest
Scan status
latest
Image
September 25, 2023, 15:55:59 (UTC+05.5)
34.70
Copy URI
sha256:cafb88e3...
-

Lambda functions



Cloud watch log: