

Hemos llegado hasta el pipeline del Jenkins pero este nos da error al intentar validar contra el AWS para subir la imagen creada. Hemos instalado el cliente de AWS en el bash del jenkins y entrado las credenciales y tambiénnn instalado el plugin del AWS Credentials:

Build log [x] AWS Creden DevOps\_PRA Shell Bash de configure aw set — AWS C

localhost:8080/job/pipeline6/11/pipeline-console/?selected-node=110 120%

**Jenkins** Search (CTRL+K) admin log out

Dashboard > pipeline6 > #11 > Pipeline Console

**Build #11** Rebuild Overview Configure

Failed 1 min 19 sec ago in 1 min 12 sec

- Tool Install
- configure aws credentials
- Checkout
- Build
- Test
- Archive
- Build Docker Image
- Push to ECR

Use a tool from a predefined tool installation

Fetches the environment variables for a given tool in a list of 'FOO=bar' ... 0.1 sec

**aws ecr get-login-password --region eu-central-1 | docker login --userna...** 2.5 sec

Shell Script

3 An error occurred (InvalidSignatureException) when calling the GetAuthorizationToken operation: The request signature we calculated does not match the signature you provided. Check your AWS Secret Access Key and signing method. Consult the service documentation for details.

4 Error: Cannot perform an interactive login from a non TTY device

5 script returned exit code 1

Jenkins 2.479.2

```
pipeline {
    agent any

    environment {
        AWS_ACCOUNT_ID="AKIAWLRRXXWRFZ44RZ4K"
        AWS_DEFAULT_REGION="eu-central-1"
        IMAGE_REPO_NAME="spring-boot-app"
        IMAGE_TAG="${env.BUILD_ID}"
        REPOSITORY_URI = "${AWS_ACCOUNT_ID}.dkr.ecr.${AWS_DEFAULT_REGION}.amazonaws.com/${IMAGE_REPO_NAME}"
    }

    tools {
        maven 'M3'
        jdk 'JDK21'
    }

    stages {

        stage('configure aws credentials') {
            steps {
                withCredentials([
                    $class: 'AmazonWebServicesCredentialsBinding',

```

```

    accessKeyVariable: 'AWS_ACCESS_KEY_ID',
    secretKeyVariable: 'AWS_SECRET_ACCESS_KEY',
    credentialsId: 'credentials_aws'])
  {
    script
    {
      sh 'aws configure set aws_access_key_id $AWS_ACCESS_KEY_ID'
      sh 'aws configure set aws_secret_access_key $AWS_SECRET_ACCESS_KEY'
      sh 'aws configure set default.region eu-central-1'
      //sh 'aws s3 ls'
    }
  }
}
}
}

```

// Previous stages (build, test) here

```

stage('Checkout') {
  steps {
    git 'https://github.com/AlbertProfe/BooksPageable.git'
  }
}

```

```

stage('Build') {
  steps {
    sh 'mvn clean package'
  }
}

```

```

stage('Test') {
  steps {
    sh 'mvn test'
  }
}

```

```

stage('Archive') {
  steps {
    archiveArtifacts artifacts: '**/target/*.jar', fingerprint: true
  }
}

```

```

stage('Build Docker Image') {
  steps {
    script {
      // Build the Docker image and tag it with both BUILD_NUMBER and latest
      sh "docker build -t ${IMAGE_REPO_NAME}:${IMAGE_TAG} -t $
{IMAGE_REPO_NAME}:latest ."
    }
  }
}
}

```

```

stage('Push to ECR') {
    steps {
        script {
            sh "aws ecr get-login-password --region ${AWS_DEFAULT_REGION} | docker login
--username AWS --password-stdin ${AWS_ACCOUNT_ID}.dkr.ecr.$
${AWS_DEFAULT_REGION}.amazonaws.com"
            sh "docker tag ${IMAGE_REPO_NAME}:${IMAGE_TAG} ${REPOSITORY_URI}:
${IMAGE_TAG}"
            sh "docker push ${REPOSITORY_URI}:${IMAGE_TAG}"
        }
    }
}

stage('Deploy to ECS') {
    steps {
        script {
            sh "aws ecs update-service --cluster your-cluster-name --service your-service-name --
force-new-deployment"
        }
    }
}
}

```