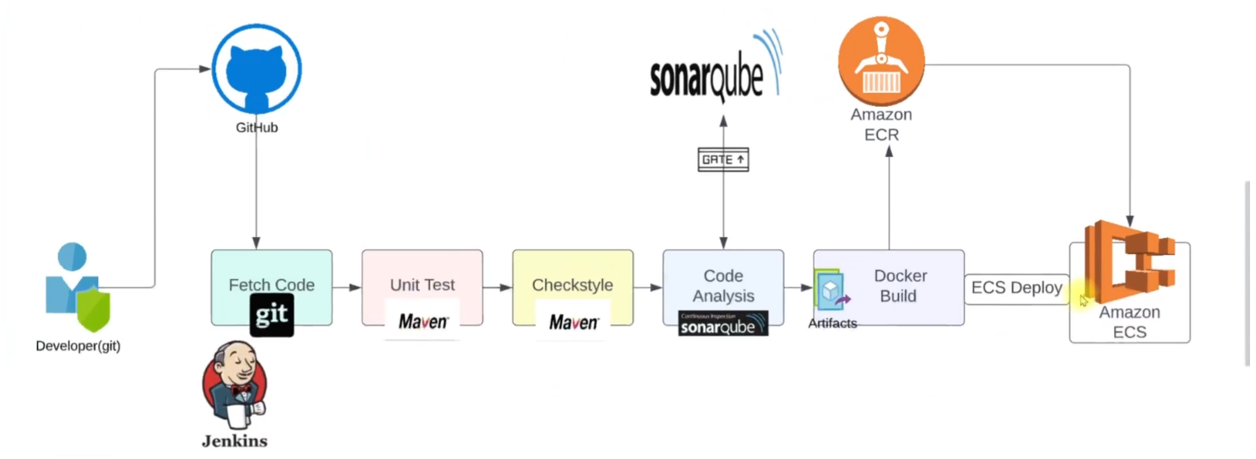
CICD jenkins & AWS ECS

project overview :

in this project i will be creating a jenkins CICD pipeline for web application deployment using Jenkins, Sonarqube, AWS ECR & ECS.

jenkins and sonarqube are installed each on an EC2 Instance.

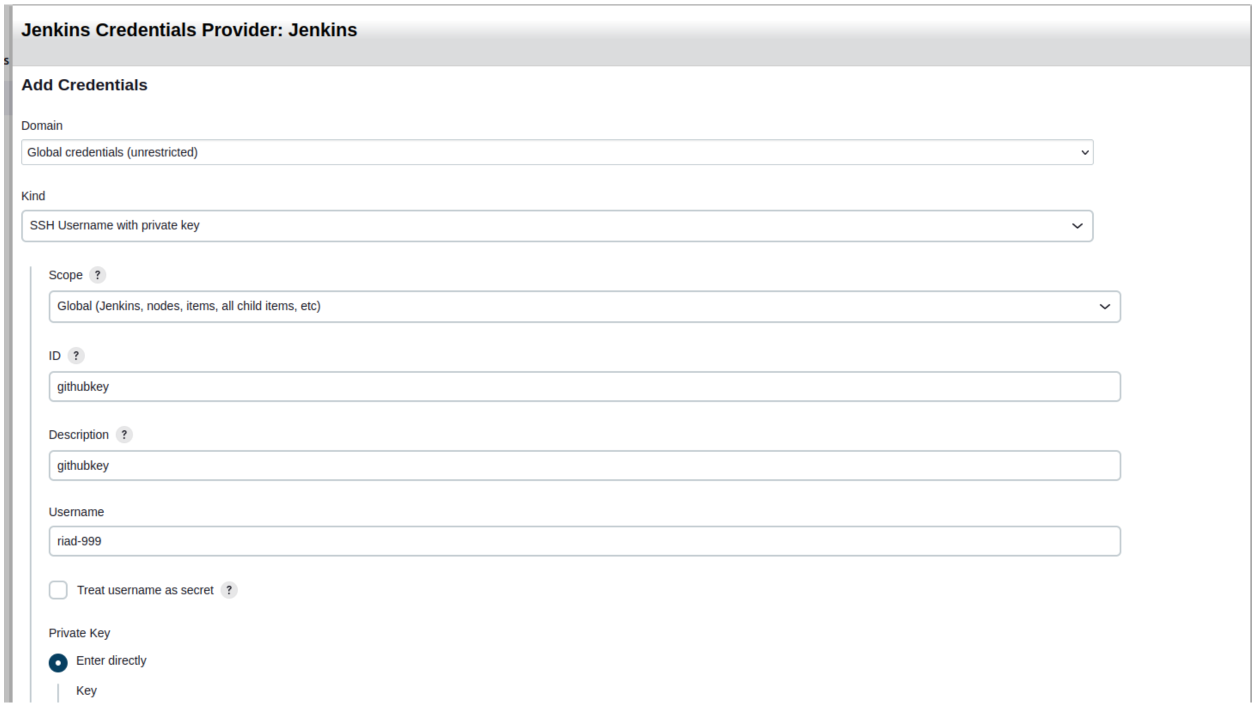


Configure Jenkins to Access Github

in this pipeline we need to give jenkins the permissions to access the github private repo, and this is done by adding the ssh private key of my github account as a jenkins credential.

first we need to make jenkins accept ssh connection : go to manage jenkins, security, host key verfication strategy ⟶ select "accept first connection".

know configure jenkins to access the github repo by adding the ssh private key to the credentials.


Jenkins Credentials Provider: Jenkins

Add Credentials

Domain: Global credentials (unrestricted)

Kind: SSH Username with private key

Scope: Global (Jenkins, nodes, items, all child items, etc)

ID: githubkey

Description: githubkey

Username: riad-999

[ ] Treat username as secret

Private Key

● Enter directly

Key

Install Prerequisites for Jenkins

Connect to jenkins instance with ssh and execute these commands to install aws cli and the docker engin :

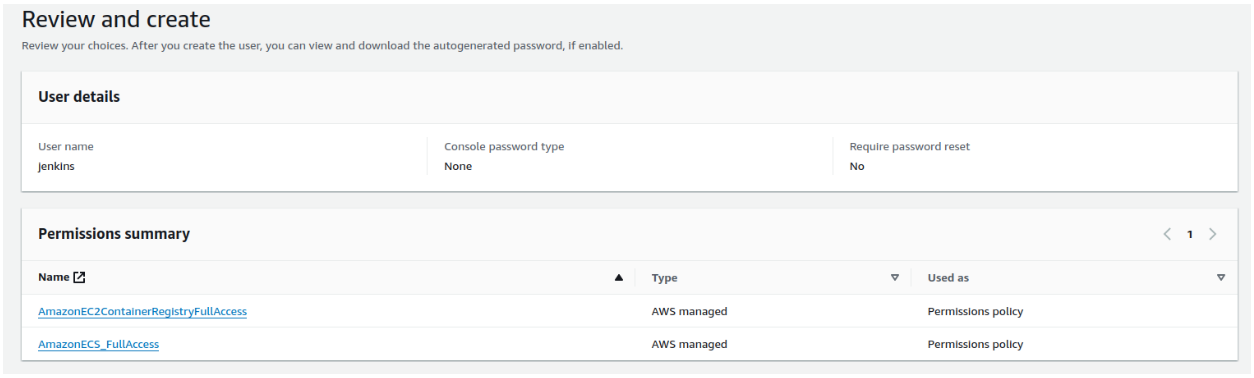
```
# install aws cli
Sudo apt update && sudo apt install awscli –y
# Add Docker's official GPG key:
sudo apt-get update
sudo apt-get install ca-certificates curl
sudo install -m 0755 -d /etc/apt/keyrings
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o
sudo chmod a+r /etc/apt/keyrings/docker.asc

# Add the repository to Apt sources:
echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/key
  $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update
sudo apt-get install docker-ce docker-ce-cli containerd.io docke

# add jenkins to the docker group to have permissions
usermod -a -G docker jenkins
```

Create jenkins IAM user

Go to IAM :

Create a user “jenkins”

Give him these two pollicies: amazonEC2containerRegistryFullAccess, amazoneEcsFullAccess



Review and create

Review your choices. After you create the user, you can view and download the autogenerated password, if enabled.

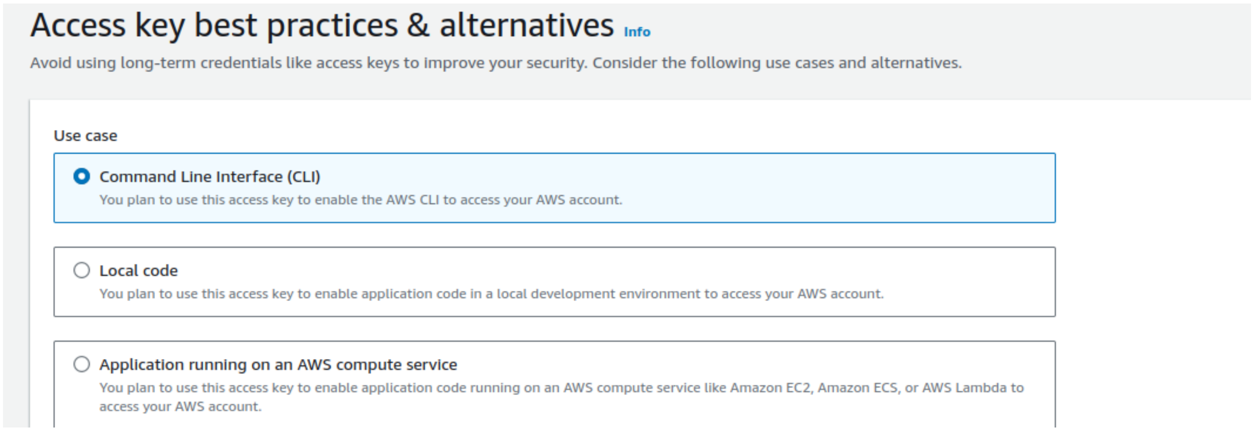
User details

User name	Console password type	Require password reset
jenkins	None	No

Permissions summary

Name	Type	Used as
AmazonEC2ContainerRegistryFullAccess	AWS managed	Permissions policy
AmazonECS_FullAccess	AWS managed	Permissions policy

Create a cli access key for the user, download it



Access key best practices & alternatives Info

Avoid using long-term credentials like access keys to improve your security. Consider the following use cases and alternatives.

Use case

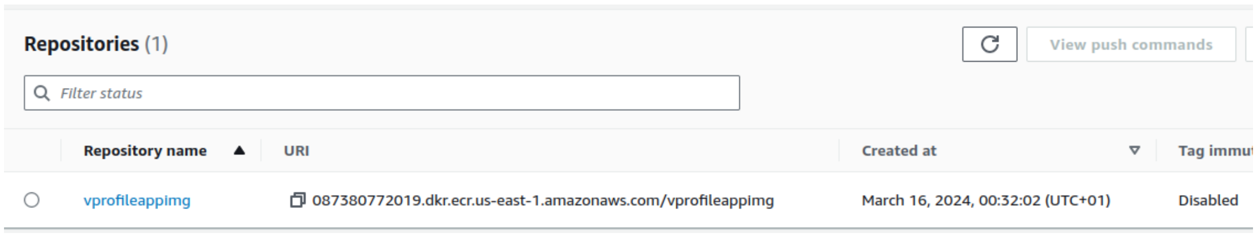
[✓ Command Line Interface (CLI)]
You plan to use this access key to enable the AWS CLI to access your AWS account.

Local code
You plan to use this access key to enable application code in a local development environment to access your AWS account.

Application running on an AWS compute service
You plan to use this access key to enable application code running on an AWS compute service like Amazon EC2, Amazon ECS, or AWS Lambda to access your AWS account.

Create an AWS ECR repository

Go to ECR service, go to repositories, create a repo, private, name: vprofileappimg

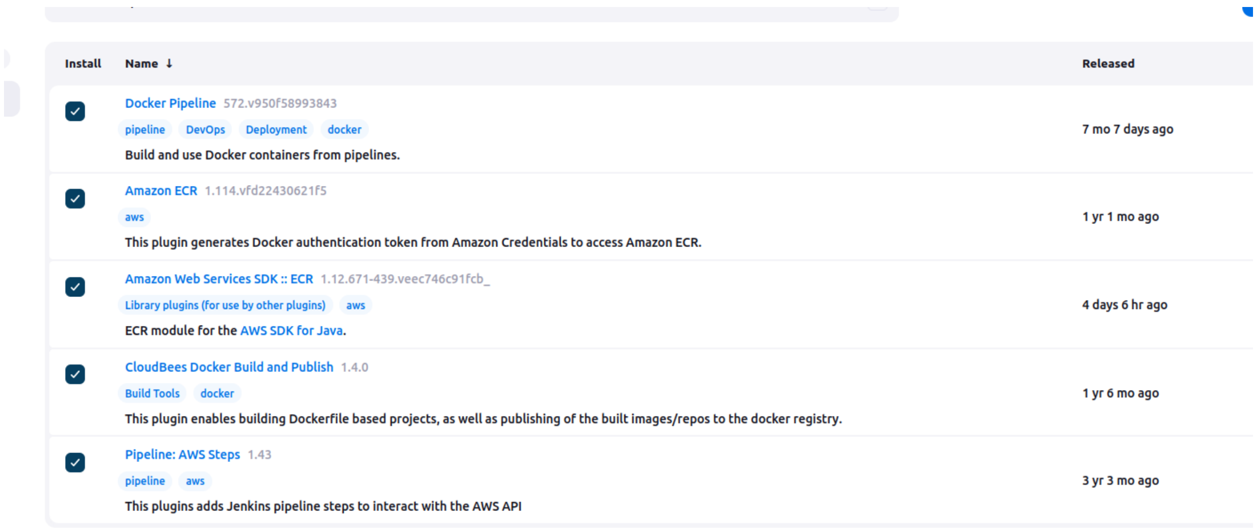


Repository name	URI	Created at	Tag immu
vprofileappimg	087380772019.dkr.ecr.us-east-1.amazonaws.com/vprofileappimg	March 16, 2024, 00:32:02 (UTC+01)	Disabled

configure jenkins for continous integration in aws ECR

Go to jenkins dashboard:

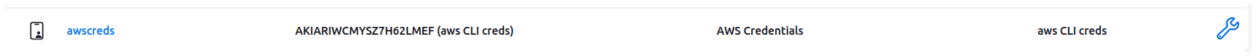
Install these plugins : docker pipeline, amazone ecr, amazone web server sdk, cloudbees   docker build and publish, pipeline aws steps



Install	Name	Released
✓	Docker Pipeline 572.v950f58993843 pipeline DevOps Deployment docker Build and use Docker containers from pipelines.	7 mo 7 days ago
✓	Amazon ECR 1.114.vfd22430621f5 aws This plugin generates Docker authentication token from Amazon Credentials to access Amazon ECR.	1 yr 1 mo ago
✓	Amazon Web Services SDK :: ECR 1.12.671-439.veec746c91fcb_ Library plugins (for use by other plugins) aws ECR module for the AWS SDK for Java.	4 days 6 hr ago
✓	CloudBees Docker Build and Publish 1.4.0 Build Tools docker This plugin enables building Dockerfile based projects, as well as publishing of the built images/repos to the docker registry.	1 yr 6 mo ago
✓	Pipeline: AWS Steps 1.43 pipeline aws This plugins adds Jenkins pipeline steps to interact with the AWS API	3 yr 3 mo ago

Go to manage global credentials, add a credential:

Select kind: aws credentials, id: awscreds, set the access and secret keys



awscreds	AKIARIWCMYSZ7H62LMEF (aws CLI creds)	AWS Credentials	aws CLI creds

Set the enviorment variables of the jenkins file

jenkins file for continous integration

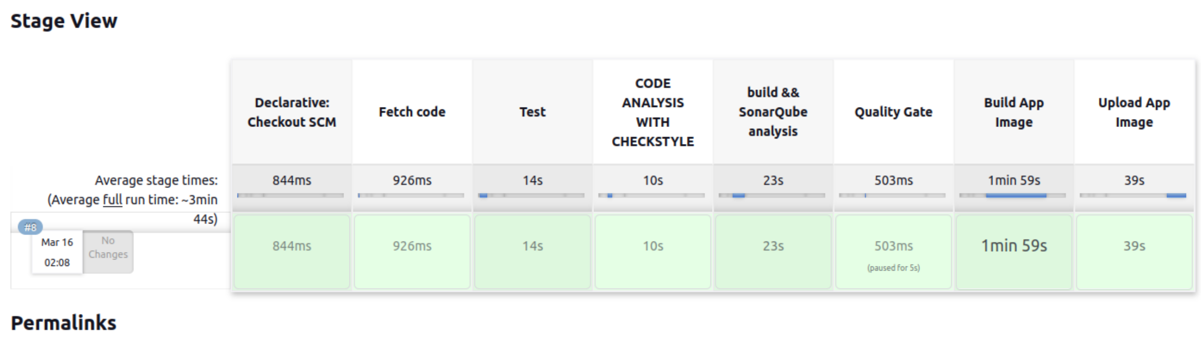
```
pipeline {
    agent any
    environment {
        registryCredential = 'ecr:us-east-1:awscreds'
        appRegistry = "087380772019.dkr.ecr.us-east-1.amazonaws
        vprofileRegistry = "https://087380772019.dkr.ecr.us-east
    }
  stages {
    stage('Fetch code'){
      steps {
        script {
            // Define the SSH key credentials ID configured in J
            def sshKeyCredentials = 'githubkey'

            // Checkout code from the private GitHub repository
            git credentialsId: sshKeyCredentials, branch: 'main
        }
    }
    }


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

    stage ('CODE ANALYSIS WITH CHECKSTYLE'){
            steps {
                sh 'mvn checkstyle:checkstyle'
            }
            post {
                success {
                    echo 'Generated Analysis Result'
                }
            }
        }

        stage('build && SonarQube analysis') {
            environment {
             scannerHome = tool 'sonar4.7'
          }
            steps {
                withSonarQubeEnv('sonar') {
                 sh '''${scannerHome}/bin/sonar-scanner -Dsonar
                   -Dsonar.projectName=vprofile-repo \
                   -Dsonar.projectVersion=1.0 \
                   -Dsonar.sources=src/ \
                   -Dsonar.java.binaries=target/test-classes/com
                   -Dsonar.junit.reportsPath=target/surefire-rep
                   -Dsonar.jacoco.reportsPath=target/jacoco.exec
                   -Dsonar.java.checkstyle.reportPaths=target/ch
                }
            }
        }

        stage("Quality Gate") {
            steps {
                timeout(time: 1, unit: 'HOURS') {
                    // Parameter indicates whether to set pipeli
                    // true = set pipeline to UNSTABLE, false =
                    waitForQualityGate abortPipeline: true
                }
            }
        }

    stage('Build App Image') {
       steps {

         script {
                dockerImage = docker.build( appRegistry + ":$BUI
             }

     }
```

```
    }

    stage('Upload App Image') {
          steps{
            script {
              docker.withRegistry( vprofileRegistry, registryCre
                dockerImage.push("$BUILD_NUMBER")
                dockerImage.push('latest')
              }
            }
          }
     }
  }
}
```

docker file

```
FROM openjdk:11 AS BUILD_IMAGE
RUN apt update && apt install maven -y
RUN git clone https://github.com/devopshydclub/vprofile-project
RUN cd vprofile-project && git checkout docker && mvn install

FROM tomcat:9-jre11

RUN rm -rf /usr/local/tomcat/webapps/*

COPY --from=BUILD_IMAGE vprofile-project/target/vprofile-v2.war

EXPOSE 8080
CMD ["catalina.sh", "run"]
```

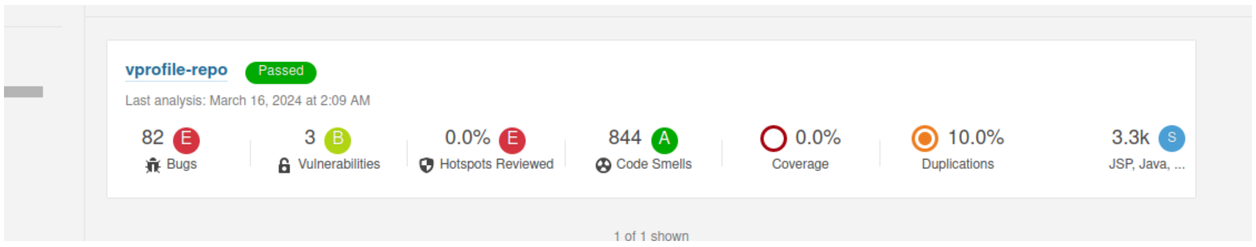
jenkins pipeline


Stage View

	Declarative: Checkout SCM	Fetch code	Test	CODE ANALYSIS WITH CHECKSTYLE	build && SonarQube analysis	Quality Gate	Build App Image	Upload App Image
Average stage times: (Average full run time: ~3min 44s)	844ms	926ms	14s	10s	23s	503ms	1min 59s	39s
#8 Mar 16 02:08 No Changes	844ms	926ms	14s	10s	23s	503ms (paused for 5s)	1min 59s	39s

Permalinks

Sonarqube

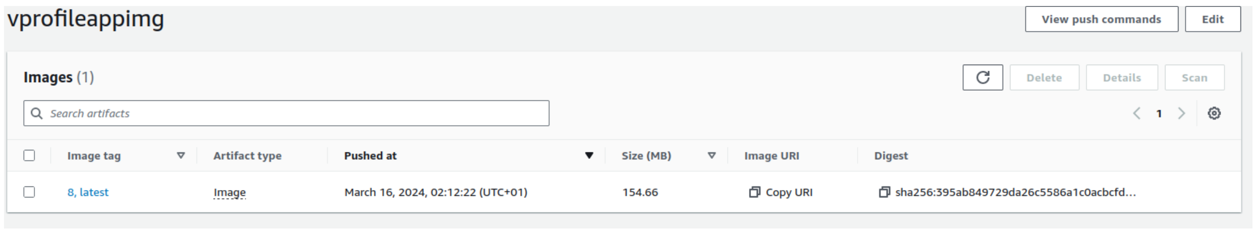

vprofile-repo Passed

Last analysis: March 16, 2024 at 2:09 AM

82 E	3 B	0.0% E	844 A	0.0%	10.0%	3.3k S
Bugs	Vulnerabilities	Hotspots Reviewed	Code Smells	Coverage	Duplications	JSP, Java, ...

1 of 1 shown

ECR new image uploaded

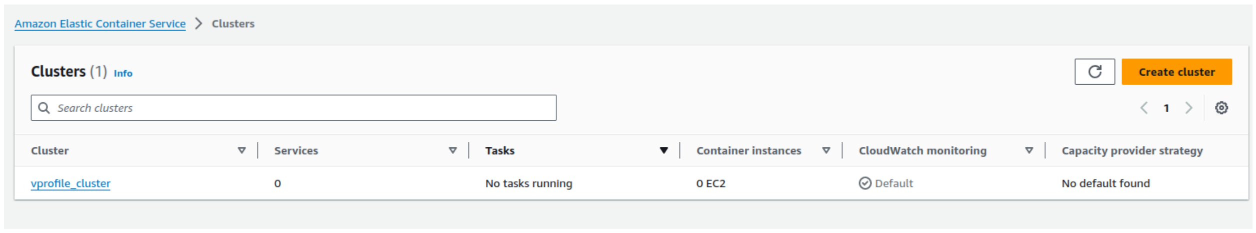

vprofileappimg

Images (1)

Image tag	Artifact type	Pushed at	Size (MB)	Image URI	Digest
8, latest	Image	March 16, 2024, 02:12:22 (UTC+01)	154.66	Copy URI	sha256:395ab849729da26c5586a1c0acbcfd...

Delivery

Go to ECS service, create a cluster


Amazon Elastic Container Service > Clusters

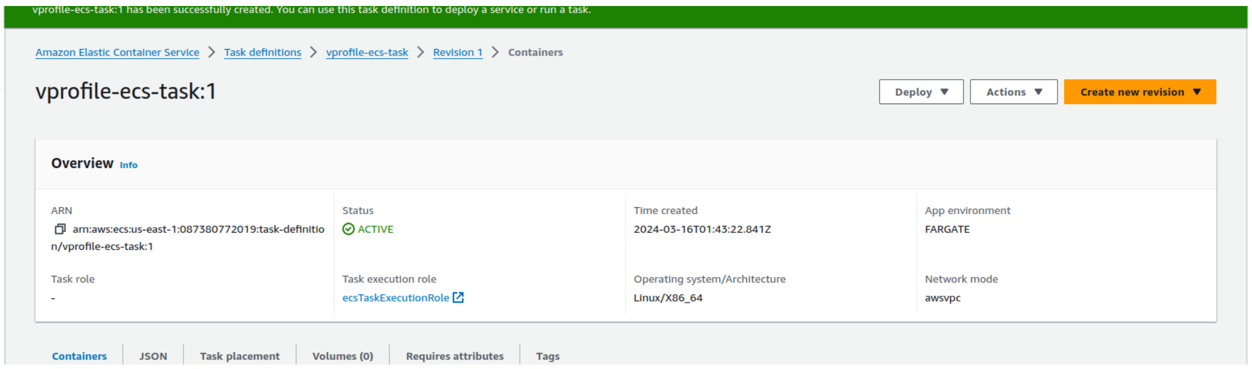
Clusters (1) Info

Cluster	Services	Tasks	Container instances	CloudWatch monitoring	Capacity provider strategy
vprofile_cluster	0	No tasks running	0 EC2	Default	No default found

Create a task definition (a task definition is similar to a template for EC2 instance)

Linux x86_64, 1 cpu, 2 GB RAM

Set the ECR URI, port 8080


vprofile-ecs-task:1 has been successfully created. You can use this task definition to deploy a service or run a task.

Amazon Elastic Container Service > Task definitions > vprofile-ecs-task > Revision 1 > Containers

vprofile-ecs-task:1

Overview Info

ARN	Status	Time created	App environment
arn:aws:ecs:us-east-1:087380772019:task-definition/vprofile-ecs-task:1	ACTIVE	2024-03-16T01:43:22.841Z	FARGATE
Task role	Task execution role	Operating system/Architecture	Network mode
-	ecsTaskExecutionRole	Linux/X86_64	awsvpc

Go to the task definition created, go to the role, add this policy to the role “CloudWatchLogsFullAccess”

Go to the cluster and create a service

Service

Familiy: “the task created” revision: latest

Service name: vprofileappsvc

Disable the deployment failure detection

Networking :

Create a new security group for the load balancer (and containers)

Name: vprofileappecselb-sg

allow HTTP on 80, custom on 8080 from anywhere

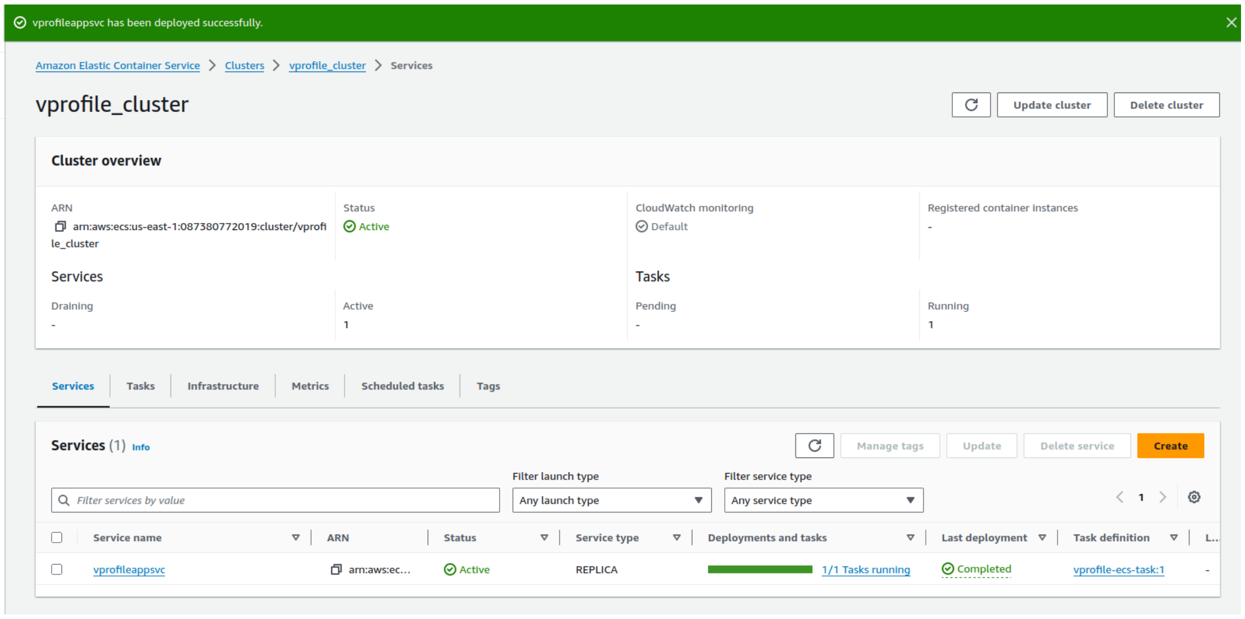
Load balancer options :

name : vproappelbecs

Container to load balancer : 8080:8080

Target group name : vproecs-tg

Healthcheck endpoint: /login



vprofileappsvc has been deployed successfully.

Amazon Elastic Container Service > Clusters > vprofile_cluster > Services

vprofile_cluster

Cluster overview

ARN	Status	CloudWatch monitoring	Registered container instances
arn:aws:ecs:us-east-1:087380772019:cluster/vprofile_cluster	Active	Default	-

Services

Draining	Active
-	1

Tasks

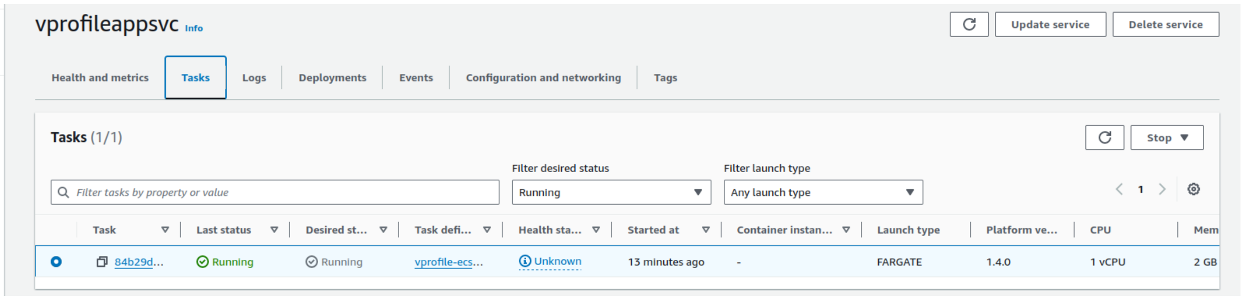
Pending	Running
-	1

Services (1) Info

Filter launch type: Any launch type
Filter service type: Any service type

Service name	ARN	Status	Service type	Deployments and tasks	Last deployment	Task definition	L...
vprofileappsvc	arn:aws:ec...	Active	REPLICA	1/1 Tasks running	Completed	vprofile-ecs-task:1	-

ec2 task running



vprofileappsvc Info

Tasks (1/1)

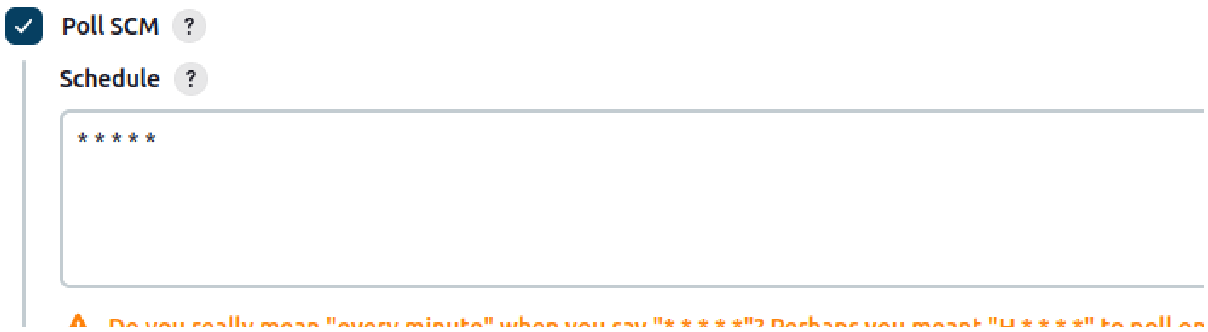
Filter desired status: Running
Filter launch type: Any launch type

Task	Last status	Desired st...	Task defi...	Health sta...	Started at	Container instan...	Launch type	Platform ve...	CPU	Mem
84b29d...	Running	Running	vprofile-ecs...	Unknown	13 minutes ago	-	FARGATE	1.4.0	1 vCPU	2 GB

Copy the DNS endpoint of the app from the service networking tab and check the web site

Edit the jenkins file: set the clustername and service name

configure jenkins Poll SCM ever minute:



Poll SCM ✓

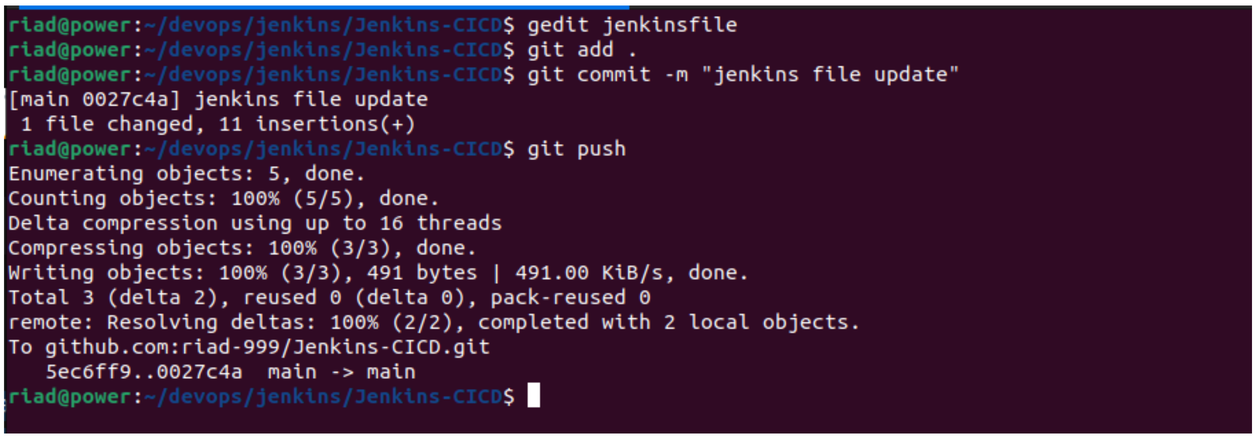
Schedule

```
* * * * *
```

Do you really mean "every minute" when you say "* * * * *"? Perhaps you meant "H * * * *" to poll on here is the final jenkins file for CICD

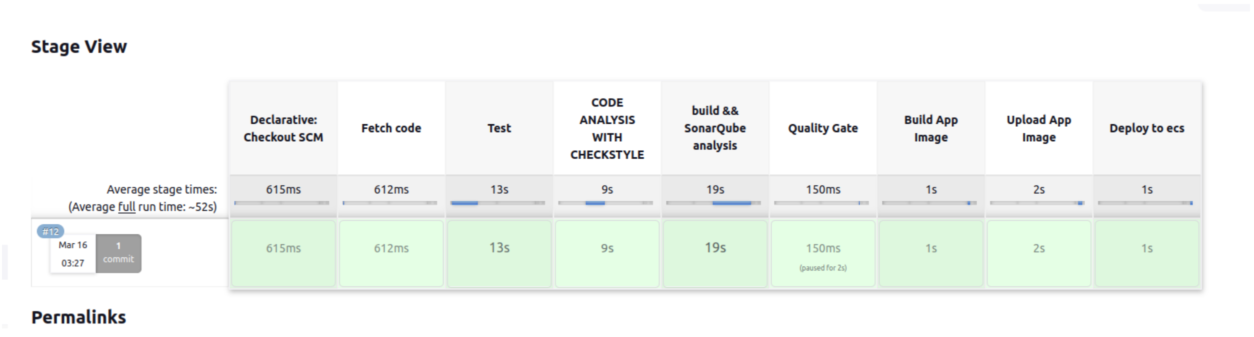
```
pipeline {
    agent any
    environment {
        registryCredential = 'ecr:us-east-1:awscreds'
        appRegistry = "087380772019.dkr.ecr.us-east-1.amazonaws
        vprofileRegistry = "https://087380772019.dkr.ecr.us-eas
        cluster = "vprofile_cluster"
        service = "vprofileappsvc"
    }
  stages {
    stage('Fetch code'){
      steps {
        script {
            // Define the SSH key credentials ID configured in 
            def sshKeyCredentials = 'githubkey'

            // Checkout code from the private GitHub repository
            git credentialsId: sshKeyCredentials, branch: 'main
        }
    }
    }


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

    stage ('CODE ANALYSIS WITH CHECKSTYLE'){
            steps {
                sh 'mvn checkstyle:checkstyle'
            }
            post {
                success {
                    echo 'Generated Analysis Result'
                }
            }
        }

        stage('build && SonarQube analysis') {
            environment {
             scannerHome = tool 'sonar4.7'
          }
            steps {
                withSonarQubeEnv('sonar') {
                 sh '''${scannerHome}/bin/sonar-scanner -Dsonar
                   -Dsonar.projectName=vprofile-repo \
                   -Dsonar.projectVersion=1.0 \
                   -Dsonar.sources=src/ \
                   -Dsonar.java.binaries=target/test-classes/com
                   -Dsonar.junit.reportsPath=target/surefire-rep
                   -Dsonar.jacoco.reportsPath=target/jacoco.exec
                   -Dsonar.java.checkstyle.reportPaths=target/ch
                }
            }
        }

        stage("Quality Gate") {
            steps {
                timeout(time: 1, unit: 'HOURS') {
                    // Parameter indicates whether to set pipeli
                    // true = set pipeline to UNSTABLE, false =
                    waitForQualityGate abortPipeline: true
                }
            }
        }

    stage('Build App Image') {
       steps {
         script {
                dockerImage = docker.build( appRegistry + ":$BU
             }

     }

    }

    stage('Upload App Image') {
          steps{
            script {
              docker.withRegistry( vprofileRegistry, registryCre
                dockerImage.push("$BUILD_NUMBER")
                dockerImage.push('latest')
              }
            }
          }
     }

     stage('Deploy to ecs') {
          steps {
                withAWS(credentials: 'awscreds', region: 'us-eas
                  sh 'aws ecs update-service --cluster ${cluste
                }
              }
           }

  }
}
```

push the modifications :



```
riad@power:~/devops/jenkins/Jenkins-CICD$ gedit jenkinsfile
riad@power:~/devops/jenkins/Jenkins-CICD$ git add .
riad@power:~/devops/jenkins/Jenkins-CICD$ git commit -m "jenkins file update"
[main 0027c4a] jenkins file update
 1 file changed, 11 insertions(+)
riad@power:~/devops/jenkins/Jenkins-CICD$ git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 16 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 491 bytes | 491.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To github.com:riad-999/Jenkins-CICD.git
   5ec6ff9..0027c4a  main -> main
riad@power:~/devops/jenkins/Jenkins-CICD$ 
```

jenkins pipeline

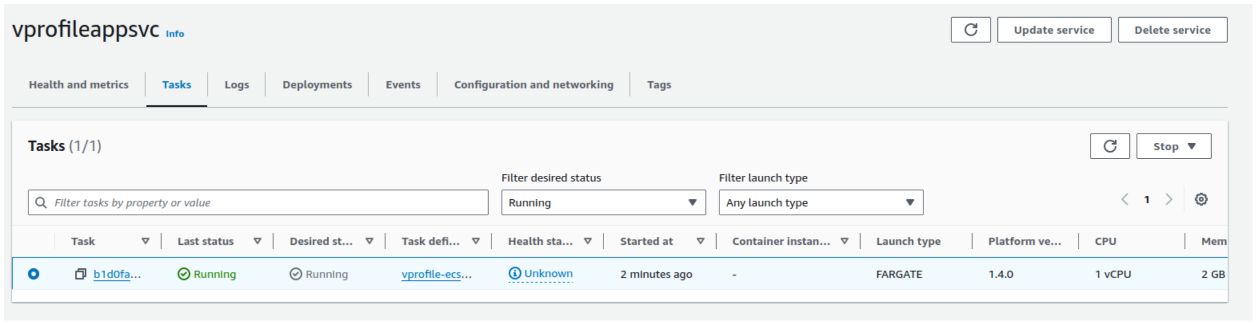


Stage View

	Declarative: Checkout SCM	Fetch code	Test	CODE ANALYSIS WITH CHECKSTYLE	build && SonarQube analysis	Quality Gate	Build App Image	Upload App Image	Deploy to ecs
Average stage times: (Average full run time: ~52s)	615ms	612ms	13s	9s	19s	150ms	1s	2s	1s
#12 Mar 16 03:27 1 commit	615ms	612ms	13s	9s	19s	150ms (paused for 2s)	1s	2s	1s

Permalinks

here is the current new running task :



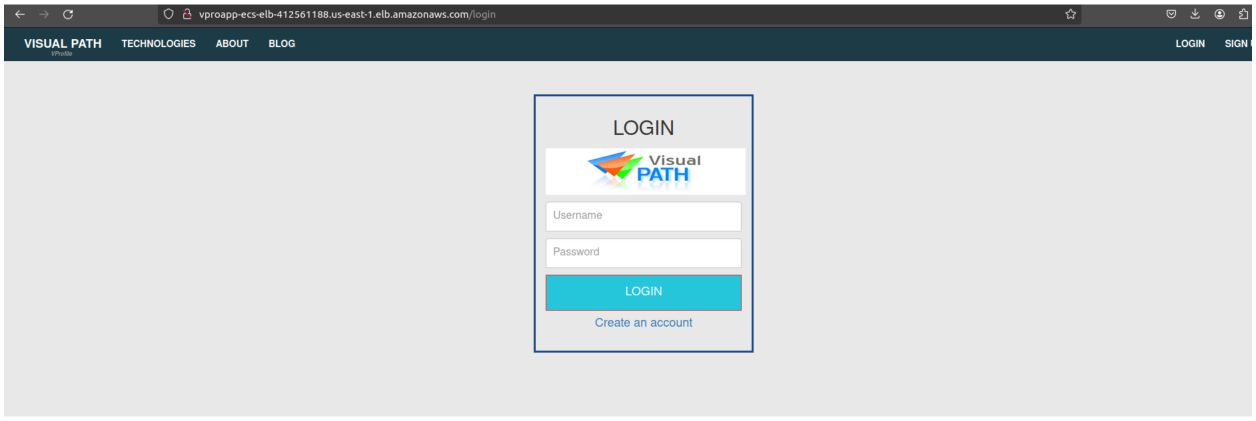
vprofileappsvc Info

Tasks (1/1)

Filter desired status: Running
Filter launch type: Any launch type

Task	Last status	Desired st...	Task defi...	Health sta...	Started at	Container instan...	Launch type	Platform ve...	CPU	Mem
b1d0fa...	Running	Running	vprofile-ecs...	Unknown	2 minutes ago	-	FARGATE	1.4.0	1 vCPU	2 GB

application is deployed successfully:


vproapp-ecs-elb-412561188.us-east-1.elb.amazonaws.com/login
VISUAL PATH
TECHNOLOGIES
ABOUT
BLOG
LOGIN
SIGN
LOGIN
Visual PATH
Username
Password
LOGIN
Create an account