# ASI INSURANCE

**Objectives:**

To create a microservice application architecture for an Insurance company through DevOps pipeline and deployment on Docker.

**Problem Statement and Motivation**

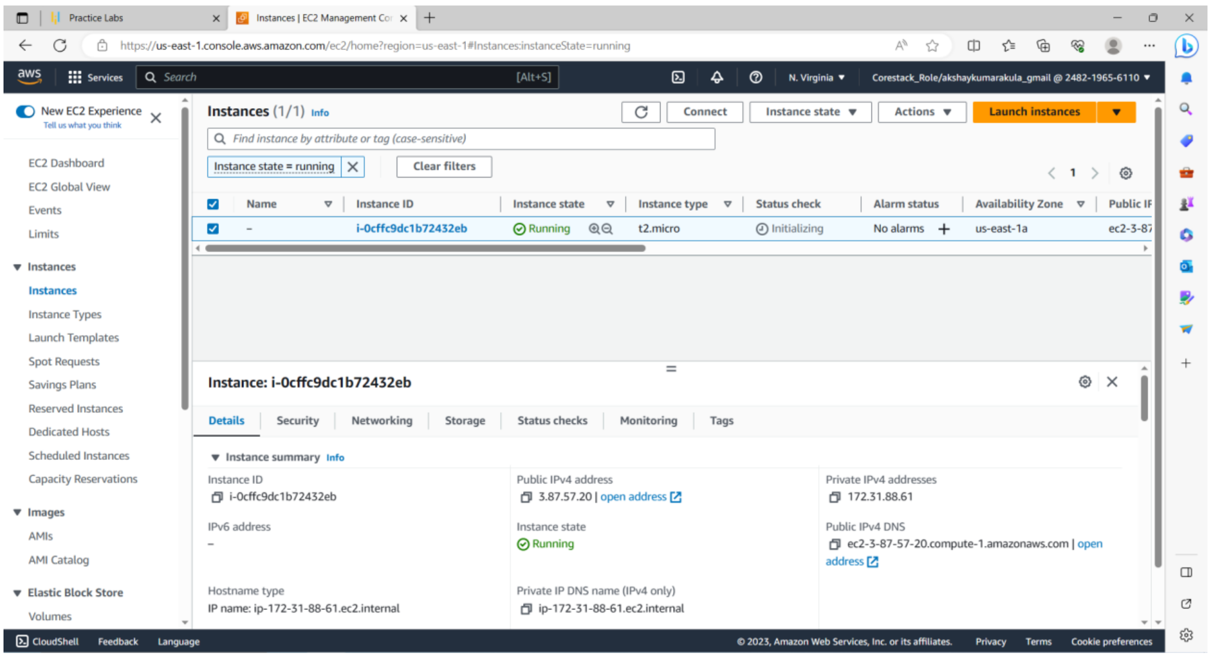
ASI Insurance is facing challenges in improving the SLA to its customers due to its organizational growth and existing monolithic application architecture. It requires transformation of the existing architecture to a microservice application architecture, while also implementing DevOps pipeline and automations.

The successful completion of the project will enable ASI Insurance to improve its overall application deployment process, enhance system scalability, and deliver better products and services to its customers.

**Task (Activities):**

1) Create the Dockerfile, Jenkinsfile, Ansible playbook, and the source file of the static website
2) Upload all the created files to GitHub
3) Go to the terminal and install NodeJS 16
4) Open the browser and access the Jenkins application
5) Create Jenkins pipeline to perform CI/CD for a Docker container
6) Create Docker Hub Credentials and other necessary pre requisites before running build
7) Set up Docker remote host on AWS and configure deploy stage in pipeline
8) Execute Jenkins Build
9) Access deployed application on Docker container

**Step – 1:** Create an ec2 instance on the aws and edit inbound security rules to enable port 8080 and 8081 to run Jenkins and the static website on public ip


https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#Instances:instanceState=running

N. Virginia | Corestack_Role/akshaykumarakula_gmail @ 2482-1965-6110

**Instances (1/1)** Info

Connect | Instance state | Actions | Launch instances

Instance state = running | Clear filters

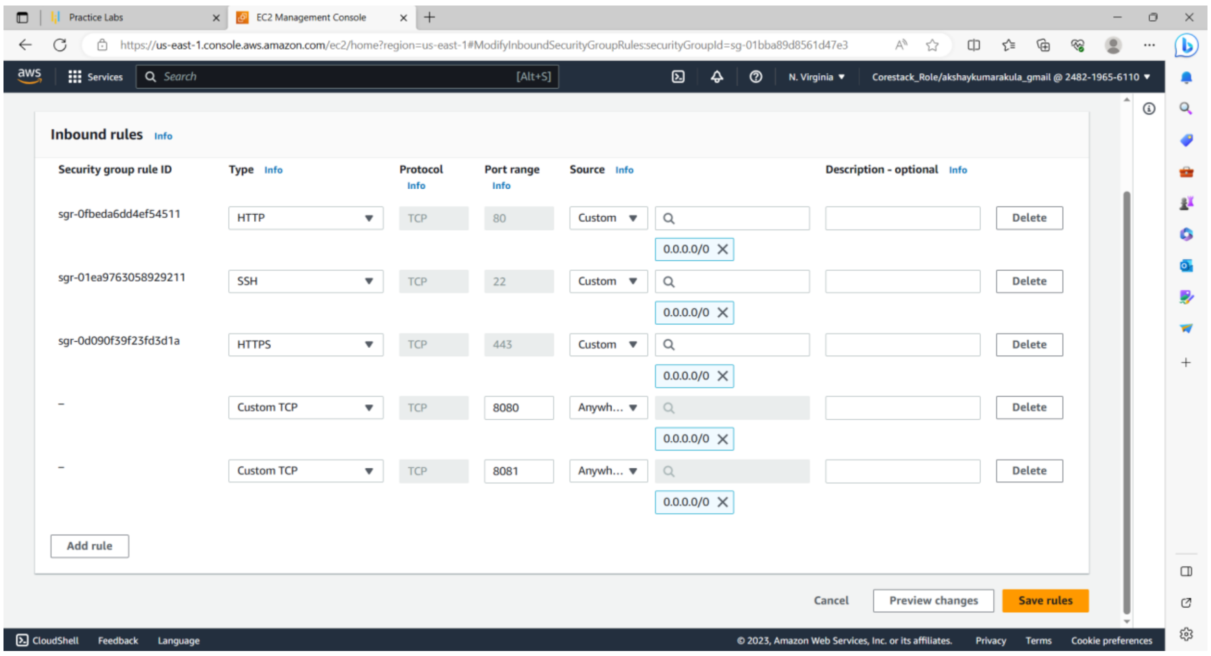
| Name | Instance ID | Instance state | Instance type | Status check | Alarm status | Availability Zone | Public IP |
|---|---|---|---|---|---|---|---|
| – | i-0cffc9dc1b72432eb | Running | t2.micro | Initializing | No alarms | us-east-1a | ec2-3-87 |

**Instance: i-0cffc9dc1b72432eb**

Details | Security | Networking | Storage | Status checks | Monitoring | Tags

Instance summary Info

Instance ID: i-0cffc9dc1b72432eb
Public IPv4 address: 3.87.57.20 | open address
Private IPv4 addresses: 172.31.88.61
IPv6 address: –
Instance state: Running
Public IPv4 DNS: ec2-3-87-57-20.compute-1.amazonaws.com | open address
Hostname type: IP name: ip-172-31-88-61.ec2.internal
Private IP DNS name (IPv4 only): ip-172-31-88-61.ec2.internal

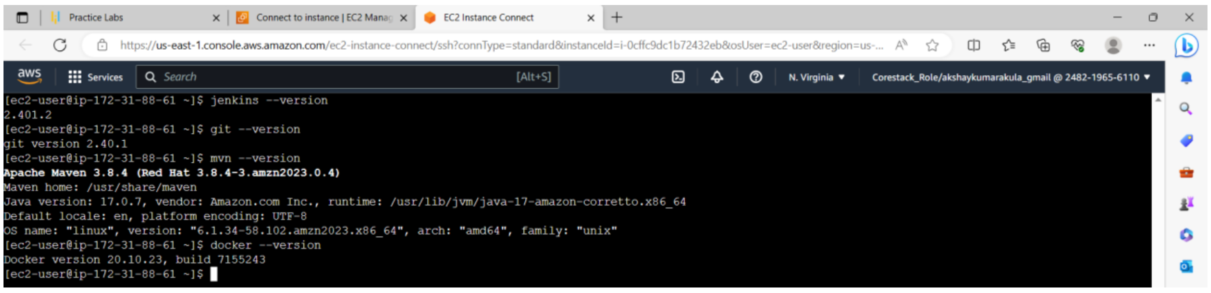

https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#ModifyInboundSecurityGroupRules:securityGroupId=sg-01bba89d8561d47e3

**Inbound rules** Info

| Security group rule ID | Type | Protocol | Port range | Source | | Description - optional |
|---|---|---|---|---|---|---|
| sgr-0fbeda6dd4ef54511 | HTTP | TCP | 80 | Custom | 0.0.0.0/0 | |
| sgr-01ea9763058929211 | SSH | TCP | 22 | Custom | 0.0.0.0/0 | |
| sgr-0d090f39f23fd3d1a | HTTPS | TCP | 443 | Custom | 0.0.0.0/0 | |
| – | Custom TCP | TCP | 8080 | Anywh... | 0.0.0.0/0 | |
| – | Custom TCP | TCP | 8081 | Anywh... | 0.0.0.0/0 | |

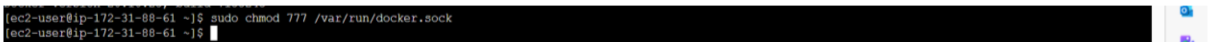
Add rule | Cancel | Preview changes | Save rules

**Step – 2:** Connect to the ec2 instance and install Jenkins, git, maven and docker on the ec2 instance



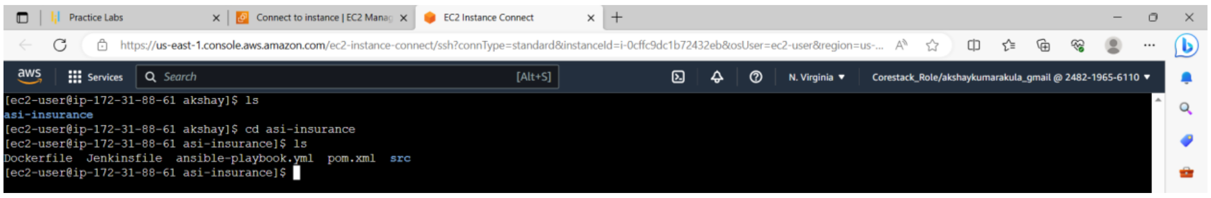
```
[ec2-user@ip-172-31-88-61 ~]$ jenkins --version
2.401.2
[ec2-user@ip-172-31-88-61 ~]$ git --version
git version 2.40.1
[ec2-user@ip-172-31-88-61 ~]$ mvn --version
Apache Maven 3.8.4 (Red Hat 3.8.4-3.amzn2023.0.4)
Maven home: /usr/share/maven
Java version: 17.0.7, vendor: Amazon.com Inc., runtime: /usr/lib/jvm/java-17-amazon-corretto.x86_64
Default locale: en, platform encoding: UTF-8
OS name: "linux", version: "6.1.34-58.102.amzn2023.x86_64", arch: "amd64", family: "unix"
[ec2-user@ip-172-31-88-61 ~]$ docker --version
Docker version 20.10.23, build 7155243
[ec2-user@ip-172-31-88-61 ~]$
```

**Step – 3:** Changed the permissions of docker.sock file to give access to docker to build containers and images



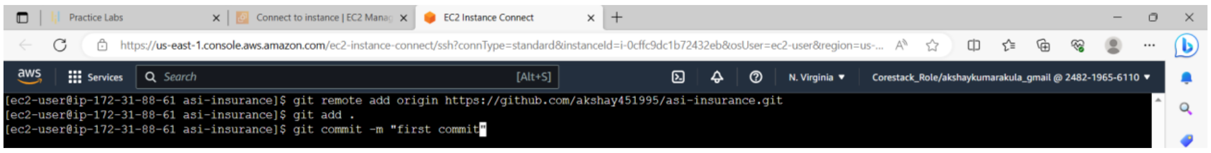
```
[ec2-user@ip-172-31-88-61 ~]$ sudo chmod 777 /var/run/docker.sock
[ec2-user@ip-172-31-88-61 ~]$
```

**Step – 4:** Created a directory named asi-insurance and created the Jenkinsfile, Dockerfile, ansible playbook (ansible-playbook.yml) and the source file of the static website inside the asi-insurance directory

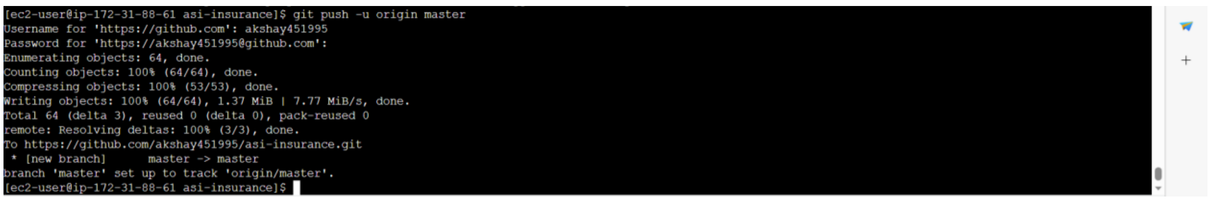


```
[ec2-user@ip-172-31-88-61 akshay]$ ls
asi-insurance
[ec2-user@ip-172-31-88-61 akshay]$ cd asi-insurance
[ec2-user@ip-172-31-88-61 asi-insurance]$ ls
Dockerfile  Jenkinsfile  ansible-playbook.yml  pom.xml  src
[ec2-user@ip-172-31-88-61 asi-insurance]$
```

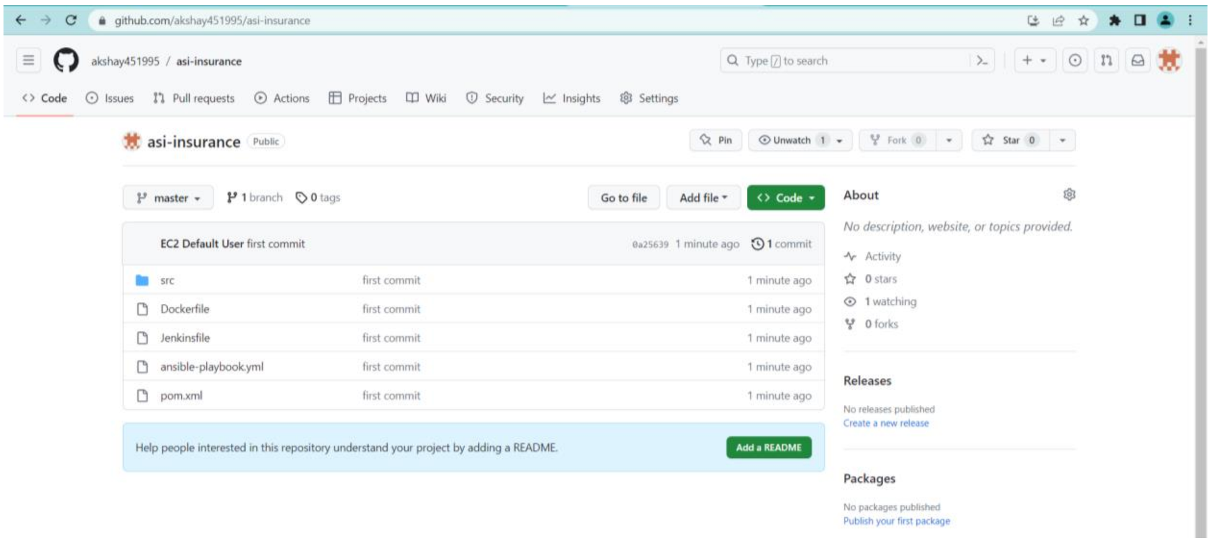
**Step – 5:** Created an empty repository inside the git hub account named asi-insurance and uploaded all the files from the asi-insurance directory to the asi-insurance git repository



```
[ec2-user@ip-172-31-88-61 asi-insurance]$ git remote add origin https://github.com/akshay451995/asi-insurance.git
[ec2-user@ip-172-31-88-61 asi-insurance]$ git add .
[ec2-user@ip-172-31-88-61 asi-insurance]$ git commit -m "first commit"
```



```
[ec2-user@ip-172-31-88-61 asi-insurance]$ git push -u origin master
Username for 'https://github.com': akshay451995
Password for 'https://akshay451995@github.com':
Enumerating objects: 64, done.
Counting objects: 100% (64/64), done.
Compressing objects: 100% (53/53), done.
Writing objects: 100% (64/64), 1.37 MiB | 7.77 MiB/s, done.
Total 64 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (3/3), done.
To https://github.com/akshay451995/asi-insurance.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.
[ec2-user@ip-172-31-88-61 asi-insurance]$
```


github.com/akshay451995/asi-insurance

akshay451995 / asi-insurance

asi-insurance Public

master · 1 branch · 0 tags

EC2 Default User first commit · 0a25639 1 minute ago · 1 commit

| | | |
|---|---|---|
| src | first commit | 1 minute ago |
| Dockerfile | first commit | 1 minute ago |
| Jenkinsfile | first commit | 1 minute ago |
| ansible-playbook.yml | first commit | 1 minute ago |
| pom.xml | first commit | 1 minute ago |

Help people interested in this repository understand your project by adding a README.

About

No description, website, or topics provided.

Releases

No releases published

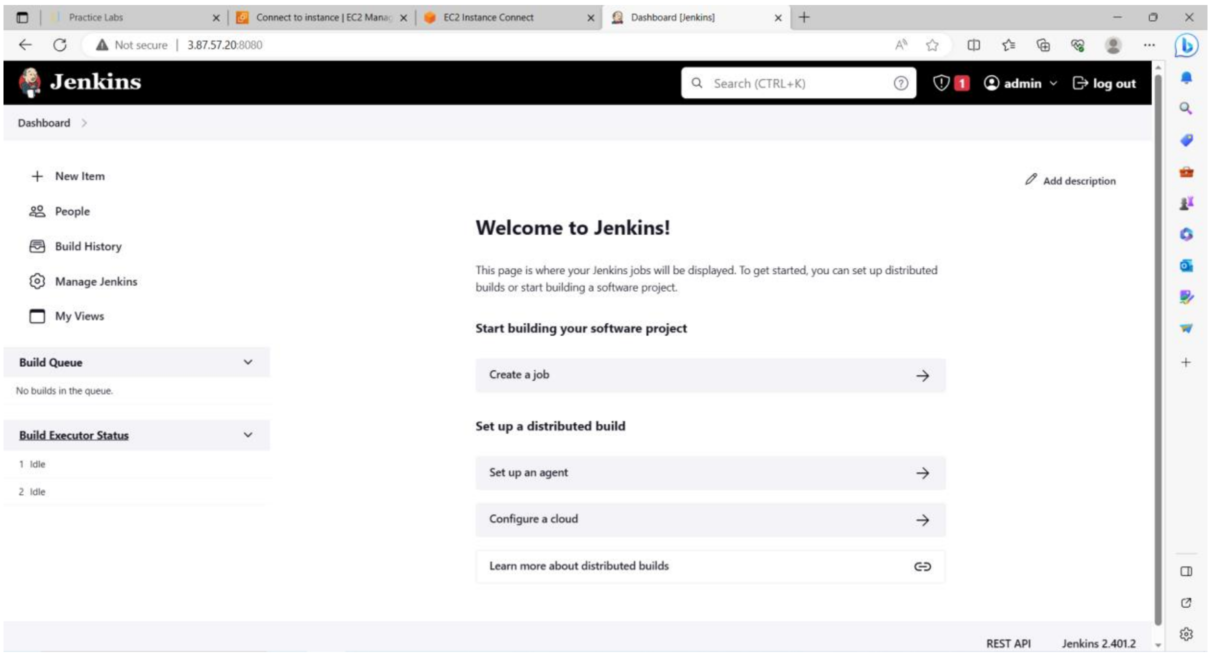
Packages

No packages published

**Step – 6:** Installing nodeJS inside the terminal:

sudo yum install nodejs -y

**Step – 7:** Opened the browser to access Jenkins application and set all the admin credentials and installed required plugins


3.87.57.20:8080

Welcome to Jenkins!

This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.

Start building your software project

Set up a distributed build

Build Queue

No builds in the queue.

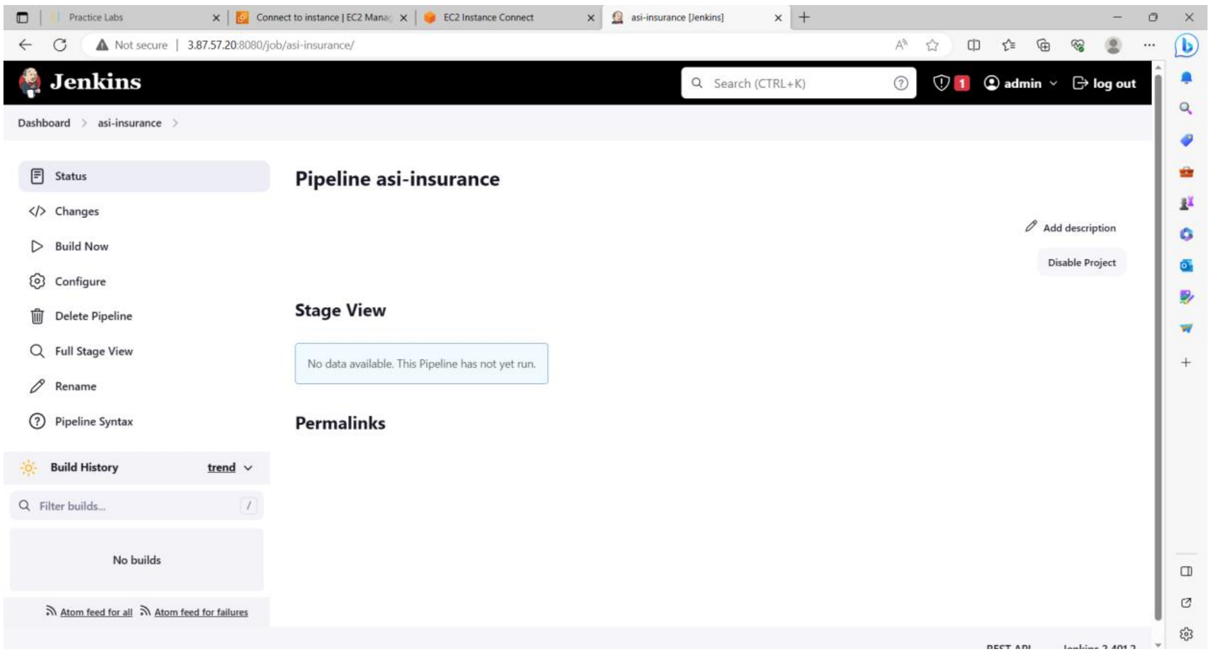
Build Executor Status

1 Idle

2 Idle

Jenkins 2.401.2

**Step – 8:** Created asi-insurance pipeline to perform CI/CD for Docker container



Pipeline asi-insurance

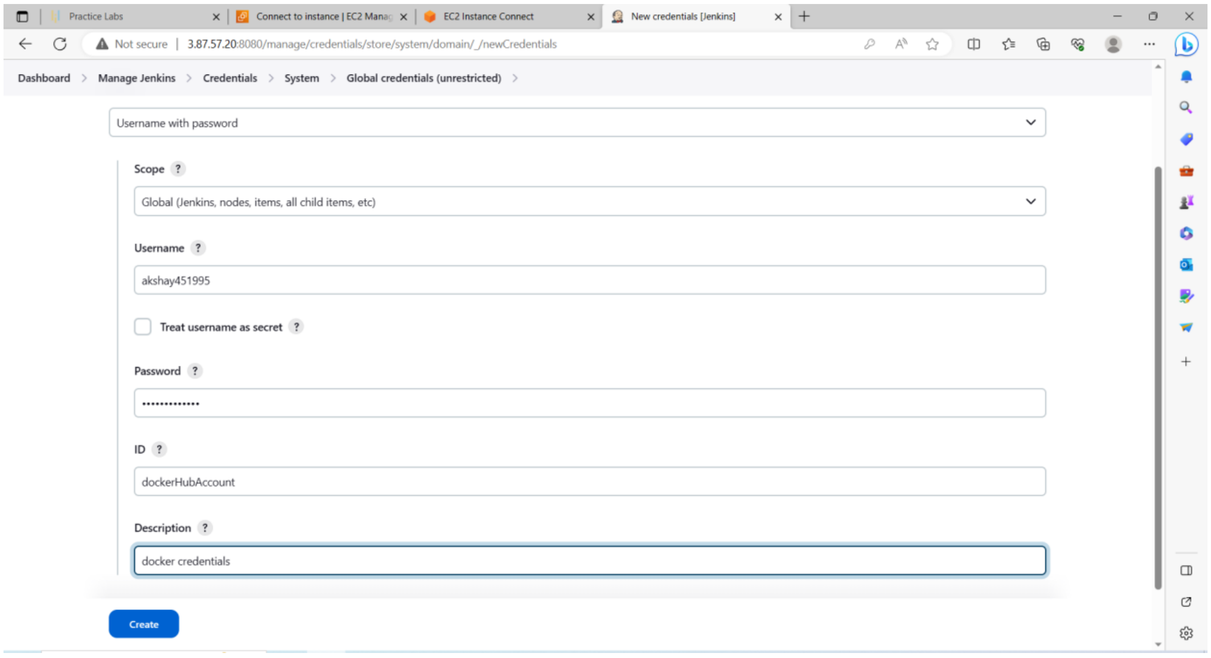
Stage View

No data available. This Pipeline has not yet run.

Permalinks

**Step – 9:** Created docker hub credentials and configured maven tool required for building the source code from SCM

Docker hub credentials:


Practice Labs
Connect to instance | EC2 Manag
EC2 Instance Connect
New credentials [Jenkins]
Not secure | 3.87.57.20:8080/manage/credentials/store/system/domain/_/newCredentials
Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) >
Create

Username with password

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

Username: akshay451995

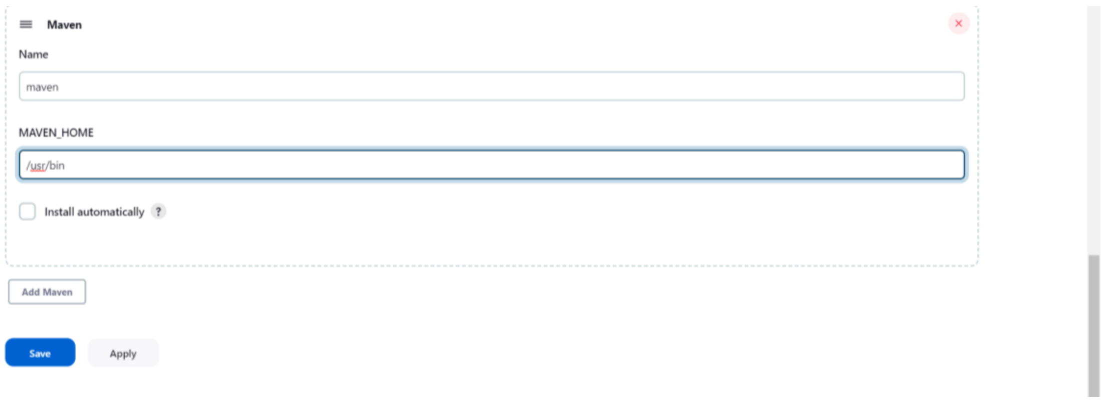
Treat username as secret

Password: •••••••••••••

ID: dockerHubAccount

Description: docker credentials

Maven tool configuration:

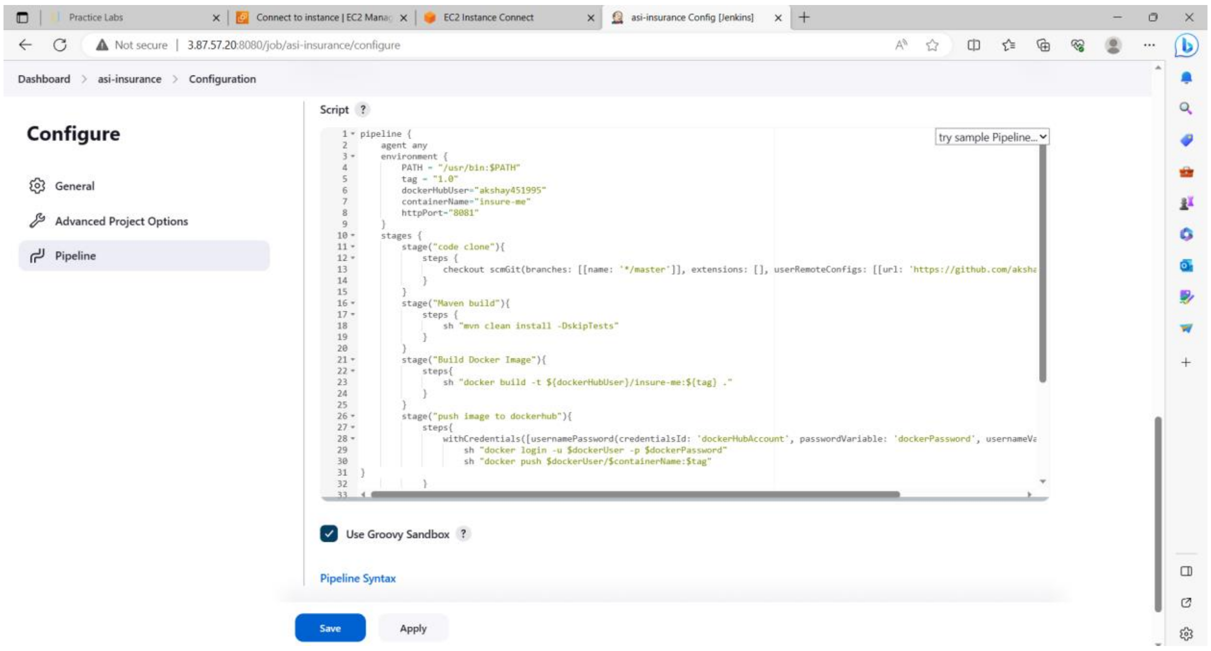


Name: maven

MAVEN_HOME: /usr/bin

☐ Install automatically

**Step – 10:** Configure the asi-insurance pipeline job and write the pipeline script using pipeline syntax option below:


Practice Labs
Connect to instance | EC2 Manag
EC2 Instance Connect
asi-insurance Config [Jenkins]
Not secure | 3.87.57.20:8080/job/asi-insurance/configure
Dashboard > asi-insurance > Configuration
Configure
General
Advanced Project Options
Pipeline
Script
try sample Pipeline...
Pipeline Syntax
Save
Apply

```
pipeline {
    agent any
    environment {
        PATH = "/usr/bin:$PATH"
        tag = "1.0"
        dockerHubUser="akshay451995"
        containerName="insure-me"
        httpPort="8081"
    }
    stages {
        stage("code clone"){
            steps {
                checkout scmGit(branches: [[name: '*/master']], extensions: [], userRemoteConfigs: [[url: 'https://github.com/aksha
            }
        }
        stage("Maven build"){
            steps {
                sh "mvn clean install -DskipTests"
            }
        }
        stage("Build Docker Image"){
            steps{
                sh "docker build -t ${dockerHubUser}/insure-me:${tag} ."
            }
        }
        stage("push image to dockerhub"){
            steps{
                withCredentials([usernamePassword(credentialsId: 'dockerHubAccount', passwordVariable: 'dockerPassword', usernameVa
                    sh "docker login -u $dockerUser -p $dockerPassword"
                    sh "docker push $dockerUser/$containerName:$tag"
}
            }
```

☑ Use Groovy Sandbox

Inside the pipeline script:

```
pipeline {
    agent any
    environment {
        PATH = "/usr/bin:$PATH"
        tag = "1.0"
        dockerHubUser="akshay451995"
        containerName="insure-me"
        httpPort="8081"
    }
    stages {
        stage("code clone"){
            steps {
                checkout scmGit(branches: [[name: '*/master']], extensions: [], userRemoteConfigs: [[url: 'https://github.com/akshay451995/asi-insurance.git']])
            }
        }
        stage("Maven build"){
            steps {
                sh "mvn clean install -DskipTests"
            }
        }
        stage("Build Docker Image"){
            steps{
                sh "docker build -t ${dockerHubUser}/insure-me:${tag} ."
            }
        }
        stage("push image to dockerhub"){
            steps{
                withCredentials([usernamePassword(credentialsId: 'dockerHubAccount', passwordVariable: 'dockerPassword', usernameVariable: 'dockerUser')]) {
                    sh "docker login -u $dockerUser -p $dockerPassword"
                    sh "docker push $dockerUser/$containerName:$tag"
}
            }
        }
        stage("Docker container deployment"){
            steps{
                sh "docker rm $containerName -f"
                sh "docker pull $dockerHubUser/$containerName:$tag"
                sh "docker run -d --rm -p $httpPort:$httpPort --name $containerName
$dockerHubUser/$containerName:$tag"
                echo "Application started on port: ${httpPort} (http)"
            }
        }
    }
}
```

Description of pipeline:

1st step – setting of environment variables of the pipeline

2nd step – cloning the code from github

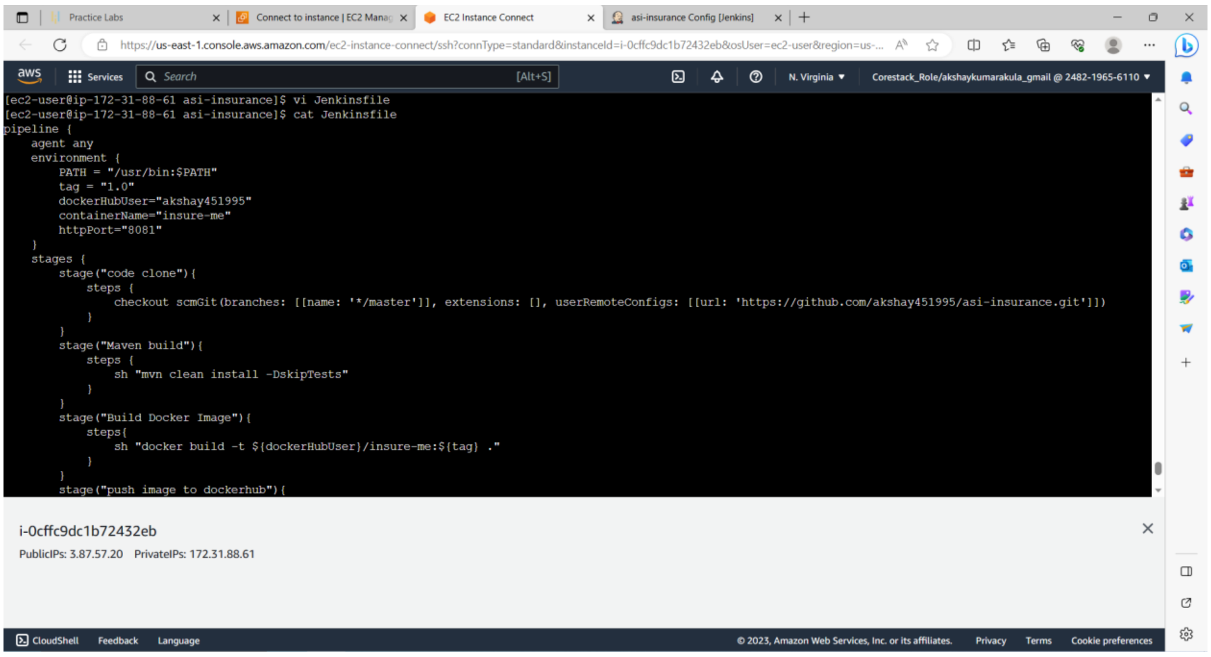
3rd step – building the maven project to install the static website application

4th step – building docker image

5th step – logging into the docker hub using docker credentials and pushing the docker image to the docker hub

6th step – creating and deploying docker container from the pulled docker image

**Step – 11:** Copy this script into the Jenkinsfile inside the asi-insurance directory from the teriminal and commit the changes to the git repository:

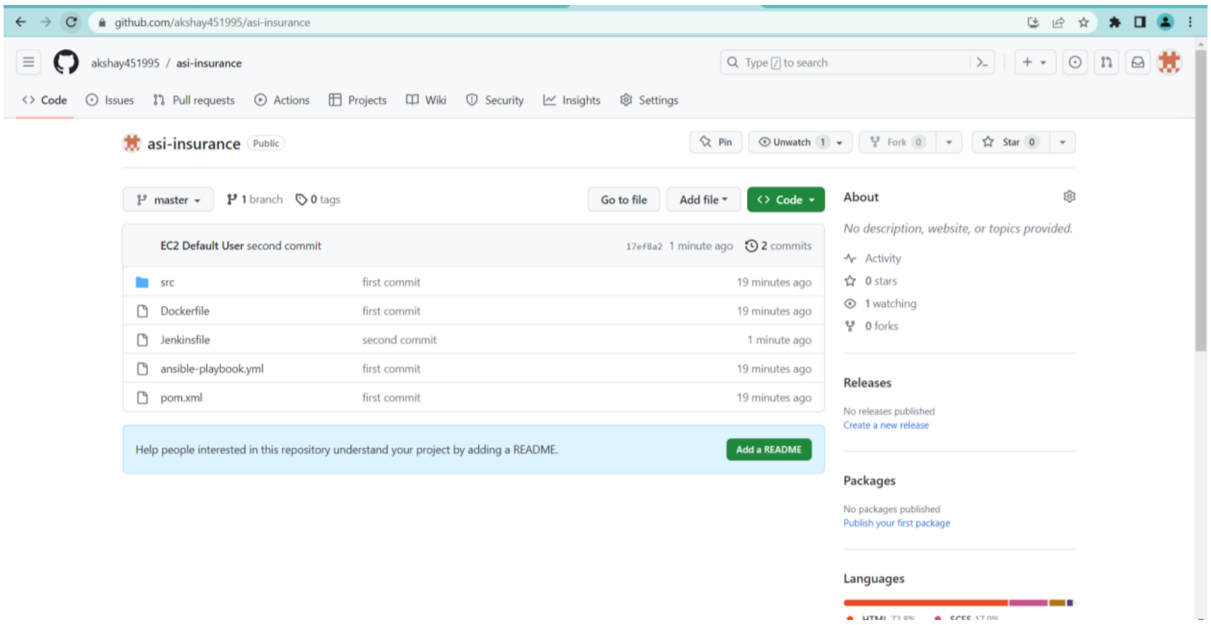


```
[ec2-user@ip-172-31-88-61 asi-insurance]$ vi Jenkinsfile
[ec2-user@ip-172-31-88-61 asi-insurance]$ cat Jenkinsfile
pipeline {
    agent any
    environment {
        PATH = "/usr/bin:$PATH"
        tag = "1.0"
        dockerHubUser="akshay451995"
        containerName="insure-me"
        httpPort="8081"
    }
    stages {
        stage("code clone"){
            steps {
                checkout scmGit(branches: [[name: '*/master']], extensions: [], userRemoteConfigs: [[url: 'https://github.com/akshay451995/asi-insurance.git']])
            }
        }
        stage("Maven build"){
            steps {
                sh "mvn clean install -DskipTests"
            }
        }
        stage("Build Docker Image"){
            steps{
                sh "docker build -t ${dockerHubUser}/insure-me:${tag} ."
            }
        }
        stage("push image to dockerhub"){
```

i-0cffc9dc1b72432eb
PublicIPs: 3.87.57.20 PrivateIPs: 172.31.88.61

Practice Labs
Connect to instance | EC2 Manag
EC2 Instance Connect
asi-insurance Config [Jenkins]
https://us-east-1.console.aws.amazon.com/ec2-instance-connect/ssh?connType=standard&instanceId=i-0cffc9dc1b72432eb&osUser=ec2-user&region=us-...
Services
Search [Alt+S]
N. Virginia
Corestack_Role/akshaykumarakula_gmail @ 2482-1965-6110
CloudShell Feedback Language
© 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Commited the pipeline script to the Jenkinsfile in github as “second commit”:



EC2 Default User second commit 17ef8a2 1 minute ago 2 commits

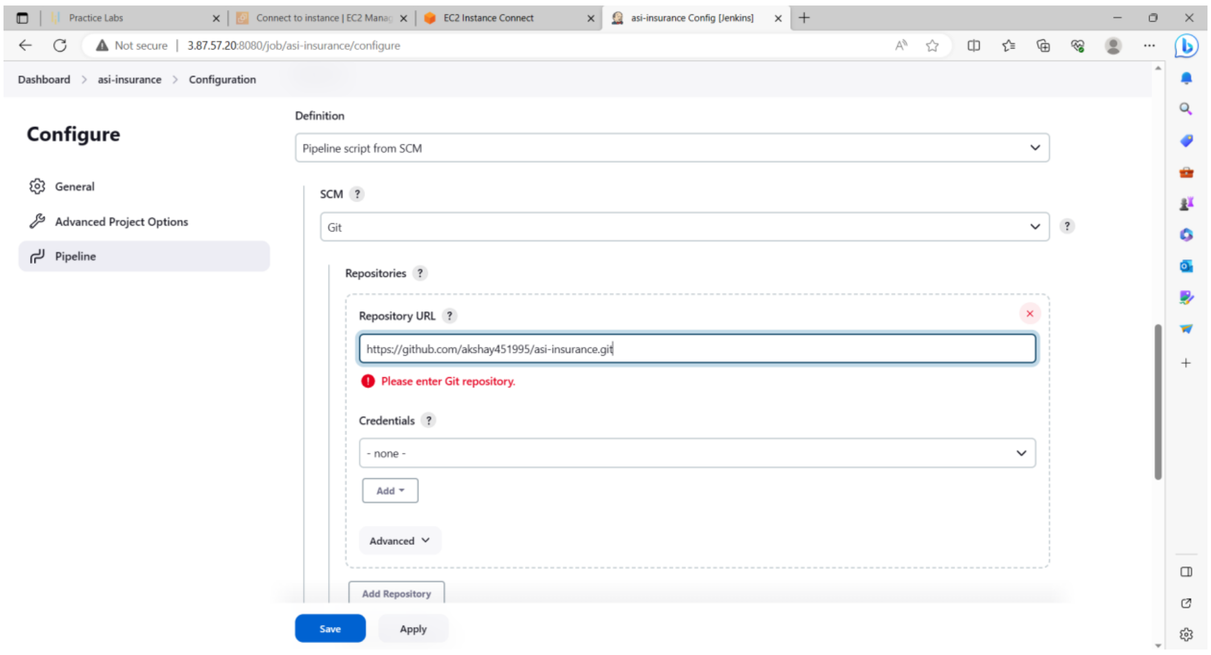
| | | |
|---|---|---|
| src | first commit | 19 minutes ago |
| Dockerfile | first commit | 19 minutes ago |
| Jenkinsfile | second commit | 1 minute ago |
| ansible-playbook.yml | first commit | 19 minutes ago |
| pom.xml | first commit | 19 minutes ago |

Help people interested in this repository understand your project by adding a README.

About

No description, website, or topics provided.

**Step – 12:** Configure the Jenkins asi-insurance pipeline by changing from Pipeline script to Pipeline script from SCM



Dashboard > asi-insurance > Configuration

Configure

Definition: Pipeline script from SCM

SCM: Git

Repositories

Repository URL: https://github.com/akshay451995/asi-insurance.git

Please enter Git repository.

Credentials: - none -

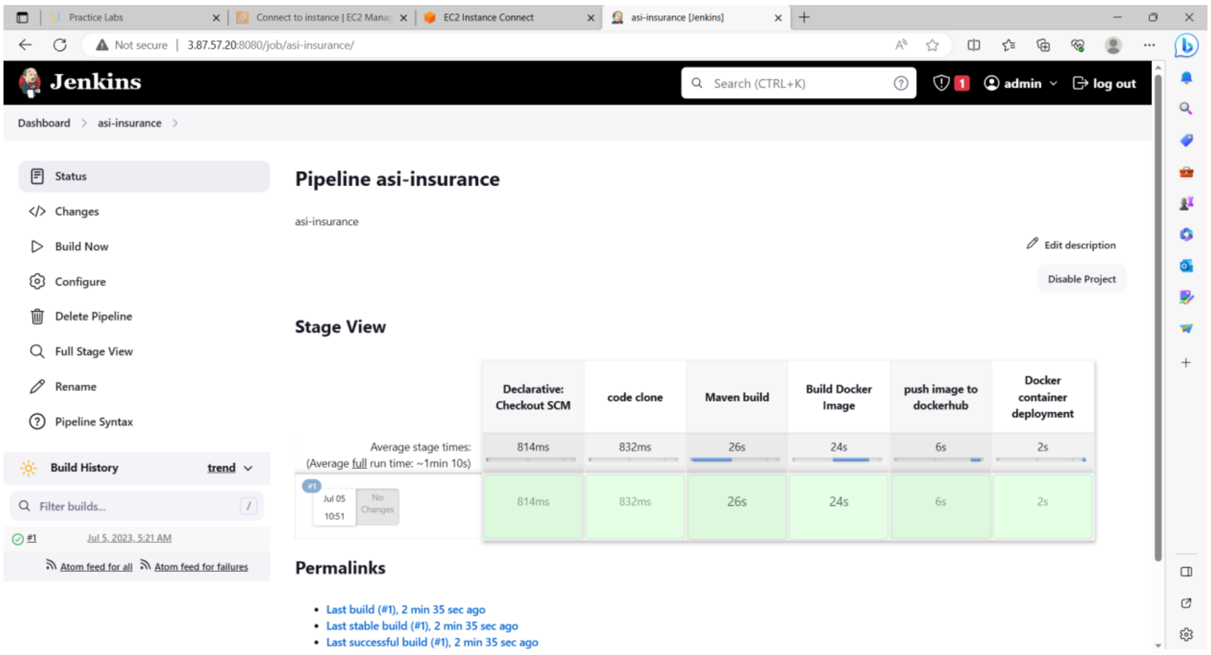
Add Repository

Save Apply

Add the git repository and its credentials if any

**Step – 13:** Execute the Jenkins build by clicking on build now for the asi-insurance pipeline

Stage view of the pipeline: pipeline executed successfully



Pipeline asi-insurance

asi-insurance

Stage View

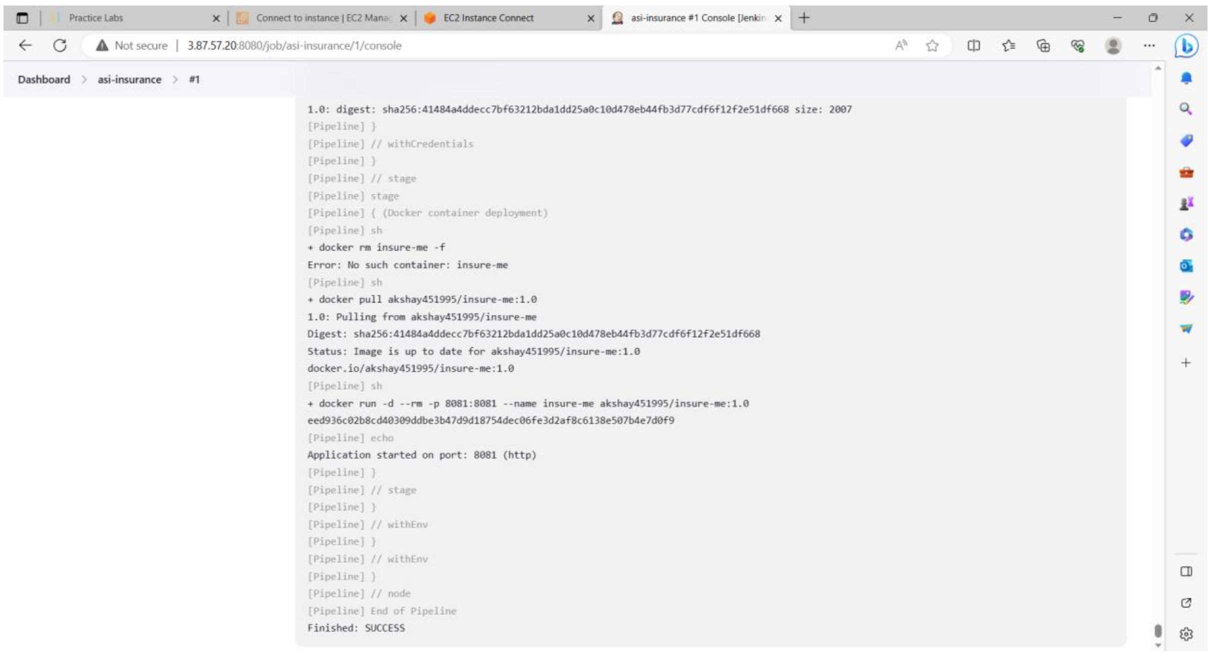
| | Declarative: Checkout SCM | code clone | Maven build | Build Docker Image | push image to dockerhub | Docker container deployment |
|---|---|---|---|---|---|---|
| Average stage times: (Average full run time: ~1min 10s) | 814ms | 832ms | 26s | 24s | 6s | 2s |
| #1 Jul 05 10:51 No Changes | 814ms | 832ms | 26s | 24s | 6s | 2s |

Build History: #1 Jul 5, 2023, 5:21 AM

Permalinks

- Last build (#1), 2 min 35 sec ago
- Last stable build (#1), 2 min 35 sec ago
- Last successful build (#1), 2 min 35 sec ago

Console output: docker container with the application built successfully on port 8081


Practice Labs
Connect to instance | EC2 Manag
EC2 Instance Connect
asi-insurance #1 Console [Jenkin
Not secure | 3.87.57.20:8080/job/asi-insurance/1/console
Dashboard > asi-insurance > #1

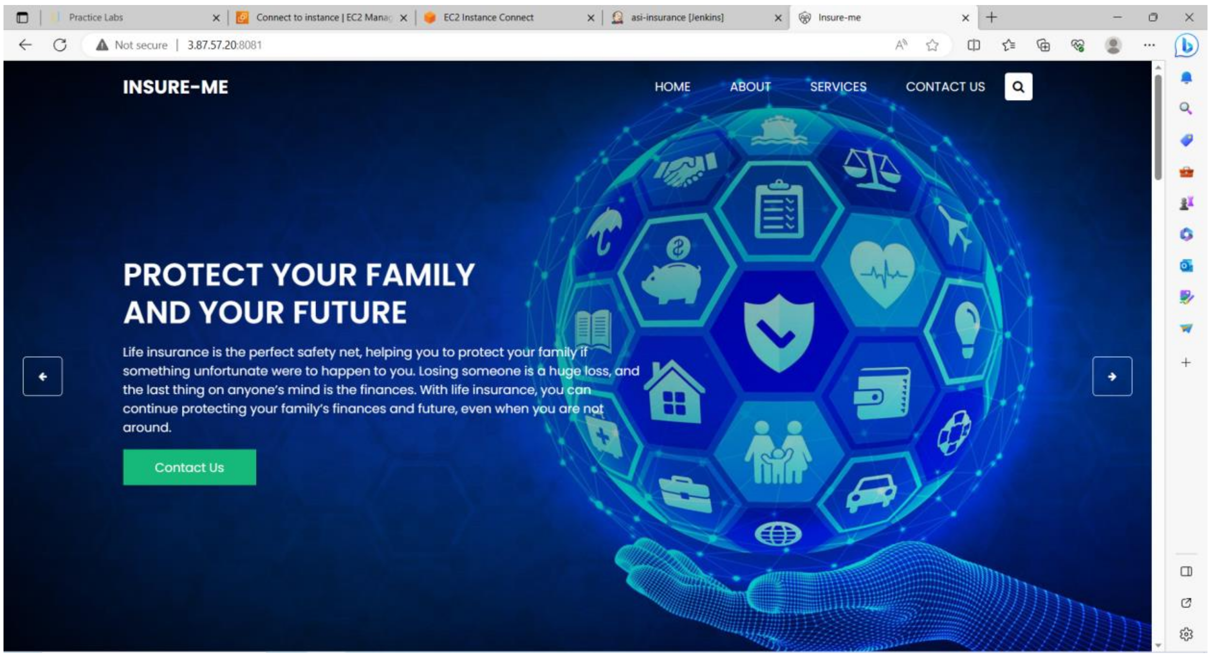
```
1.0: digest: sha256:41484a4ddecc7bf63212bda1dd25a0c10d478eb44fb3d77cdf6f12f2e51df668 size: 2007
[Pipeline] }
[Pipeline] // withCredentials
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Docker container deployment)
[Pipeline] sh
+ docker rm insure-me -f
Error: No such container: insure-me
[Pipeline] sh
+ docker pull akshay451995/insure-me:1.0
1.0: Pulling from akshay451995/insure-me
Digest: sha256:41484a4ddecc7bf63212bda1dd25a0c10d478eb44fb3d77cdf6f12f2e51df668
Status: Image is up to date for akshay451995/insure-me:1.0
docker.io/akshay451995/insure-me:1.0
[Pipeline] sh
+ docker run -d --rm -p 8081:8081 --name insure-me akshay451995/insure-me:1.0
eed936c02b8cd40309ddbe3b47d9d18754dec06fe3d2af8c6138e507b4e7d0f9
[Pipeline] echo
Application started on port: 8081 (http)
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

**Step – 14:** Now access the static website using the public IP of the machine with port 8081

http://<public-ip>:8081

http://3.87.57.20:8081

Output Screenshot:



Practice Labs | Connect to instance | EC2 Manag | EC2 Instance Connect | asi-insurance [Jenkins] | Insure-me

Not secure | 3.87.57.20:8081

INSURE-ME

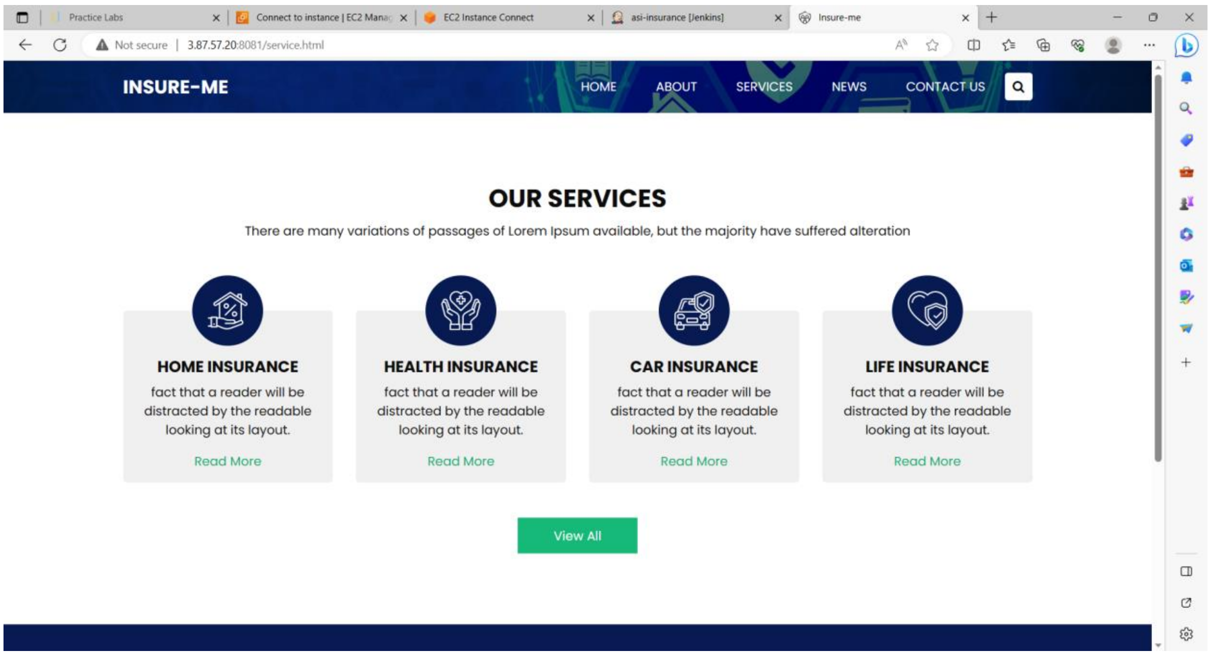
HOME ABOUT SERVICES CONTACT US

PROTECT YOUR FAMILY
AND YOUR FUTURE

Life insurance is the perfect safety net, helping you to protect your family if something unfortunate were to happen to you. Losing someone is a huge loss, and the last thing on anyone's mind is the finances. With life insurance, you can continue protecting your family's finances and future, even when you are not around.

Contact Us

Service section page:



Practice Labs | Connect to instance | EC2 Manag | EC2 Instance Connect | asi-insurance [Jenkins] | Insure-me

Not secure | 3.87.57.20:8081/service.html

INSURE-ME

HOME ABOUT SERVICES NEWS CONTACT US

OUR SERVICES

There are many variations of passages of Lorem Ipsum available, but the majority have suffered alteration

HOME INSURANCE
fact that a reader will be distracted by the readable looking at its layout.
Read More

HEALTH INSURANCE
fact that a reader will be distracted by the readable looking at its layout.
Read More

CAR INSURANCE
fact that a reader will be distracted by the readable looking at its layout.
Read More

LIFE INSURANCE
fact that a reader will be distracted by the readable looking at its layout.
Read More

View All