

## Project Description:-

As soon as the developer pushes the updated code on the GitHub master branch, the Jenkins job should be triggered using a GitHub Webhook and, the code should be checked out, compiled, tested, packaged and containerized and deployed to the preconfigured test-server automatically. The deployment should then be tested using a test automation tool (Selenium), and if the build is successful, it should be deployed to the prod server. All this should happen automatically and should be triggered from a push to the GitHub master branch

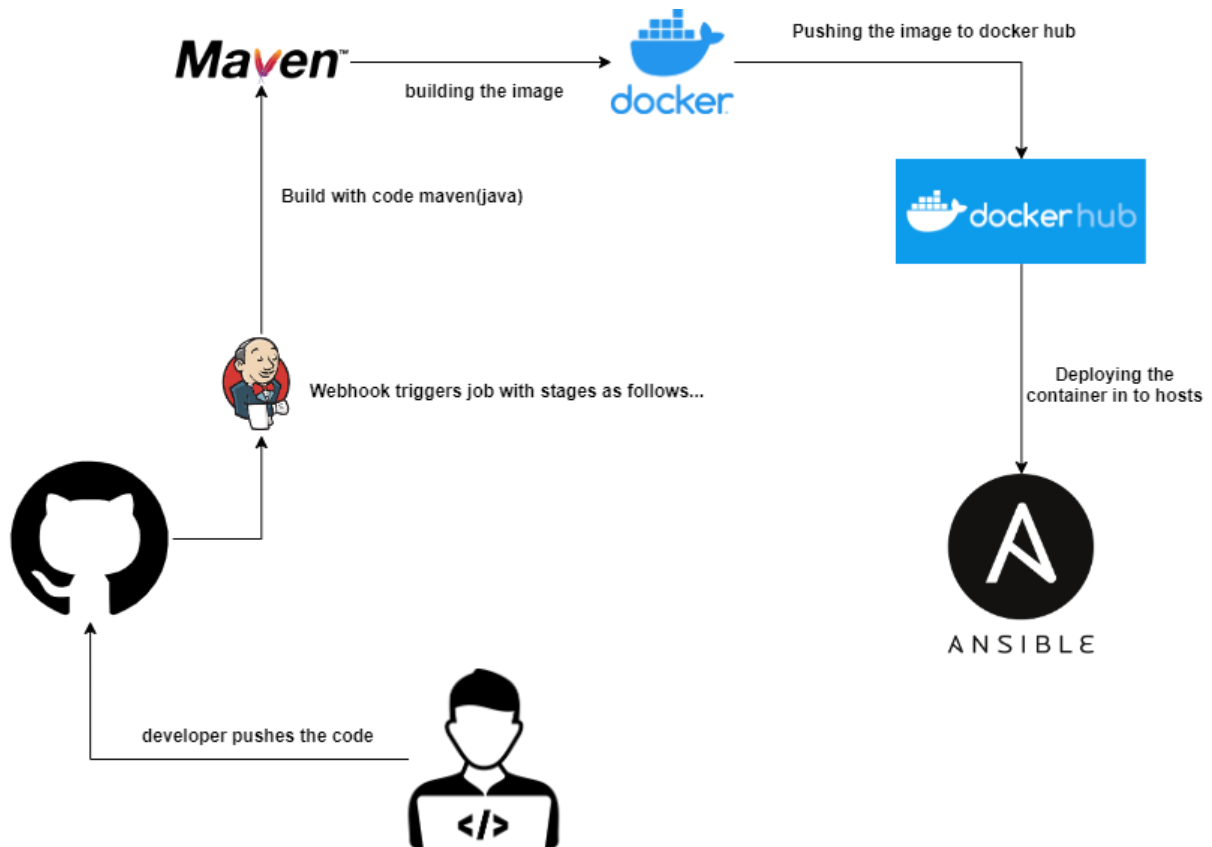
Need to implement Continuous Integration & Continuous Deployment using

- following tools:
- Git - For version control for tracking changes in the code files
- Jenkins - For continuous integration and continuous deployment
- Docker - For deploying containerized applications
- Ansible - Configuration management tools
- Selenium - For automating tests on the deployed web application
- AWS : For creating ec2 machines as servers and deploy the web application.

This project will be about how to test the services and deploy code to dev/stage/prod etc, just on a click of button

**Solution:-**

**Approach-**



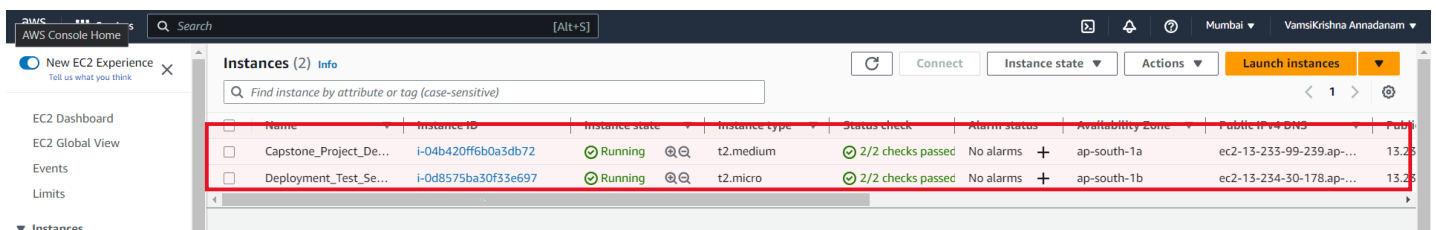
**Source git hub-** [https://github.com/vamsikrishna918/Insurance-Web-Applicaion\\_E2E](https://github.com/vamsikrishna918/Insurance-Web-Applicaion_E2E)

## Step 1:

### 1.A- EC2 instance setup

Creating 2 EC2 instances name as below

1. Capstone\_Project\_Development (T2.Medium with 30gib)
2. Deployment\_Test\_Server ( T2.Micro with 8gib)



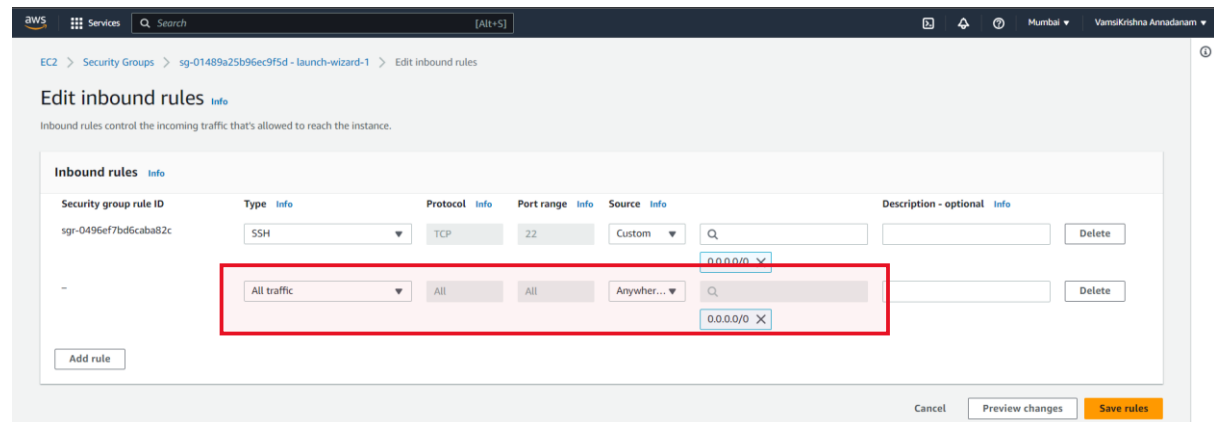
The screenshot shows the AWS Management Console 'Instances' page. Two instances are listed and highlighted with a red box:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IP v4 DNS	Public IP v4
Capstone_Project_De...	i-04b420ff6b0a3db72	Running	t2.medium	2/2 checks passed	No alarms	ap-south-1a	ec2-13-233-99-239.ap-...	13.23...
Deployment_Test_Se...	i-0d8575ba30f33e697	Running	t2.micro	2/2 checks passed	No alarms	ap-south-1b	ec2-13-234-30-178.ap-...	13.23...

### 1.B

Setting up the security groups – allowing all Inbound traffic to both instances

Path: Security->Edit Inbound rules



The screenshot shows the 'Edit inbound rules' page in the AWS console. A new rule is being added, highlighted with a red box:

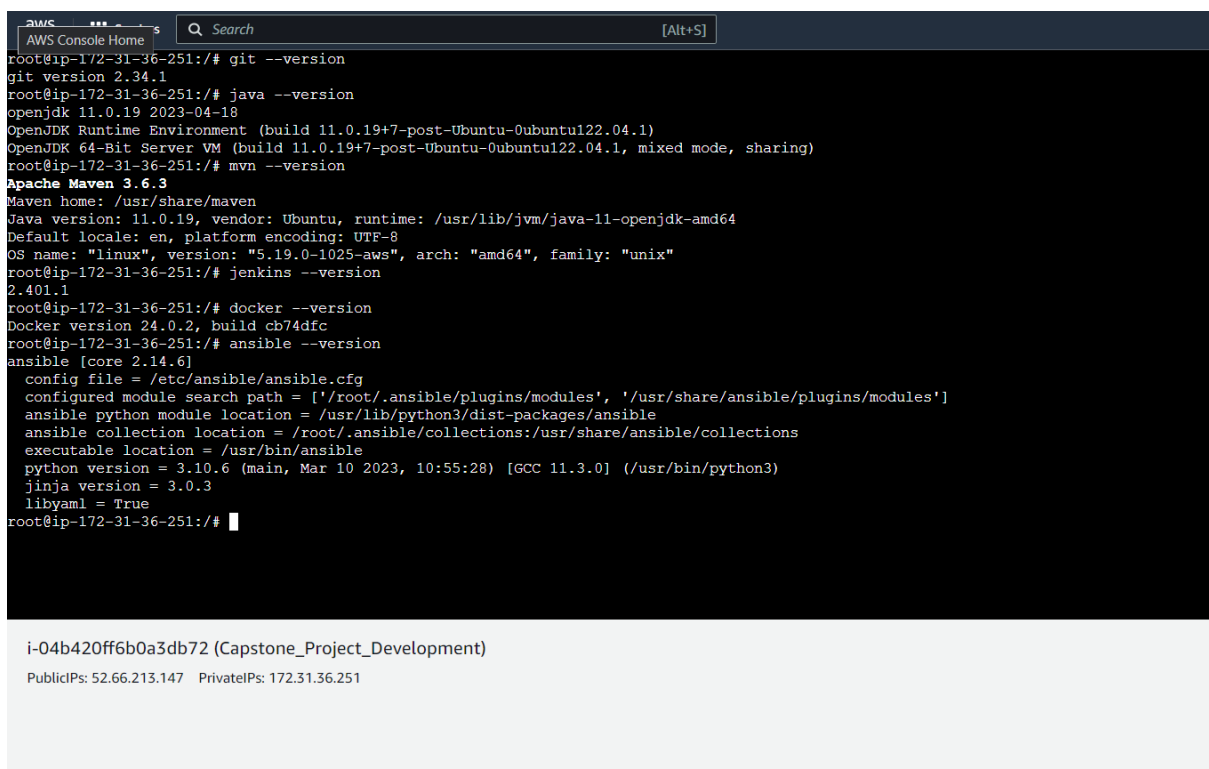
Security group rule ID	Type	Protocol	Port range	Source	Description - optional
sg-0496ef7bd6caba82c	SSH	TCP	22	Custom	
-	All traffic	All	All	Anywhere...	

### 1.C Installing packages

## Installing the necessary packages in instance (Capstone\_Project\_Development)

Follow the below sources.

1. Update packages – apt update
2. Git- <https://git-scm.com/book/en/v2/Getting-Started-Installing-Git>
3. Java- <https://www.digitalocean.com/community/tutorials/how-to-install-java-with-apt-on-ubuntu-22-04>
4. Maven- sudo apt install maven
5. Jenkins - <https://www.jenkins.io/doc/book/installing/linux/#debianubuntu>
6. Docker - <https://docs.docker.com/engine/install/ubuntu/>
7. Ansible- [https://docs.ansible.com/ansible/latest/installation\\_guide/installation\\_distros.html#installing-ansible-on-debian](https://docs.ansible.com/ansible/latest/installation_guide/installation_distros.html#installing-ansible-on-debian)



```
root@ip-172-31-36-251:~# git --version
git version 2.34.1
root@ip-172-31-36-251:~# java --version
openjdk 11.0.19 2023-04-18
OpenJDK Runtime Environment (build 11.0.19+7-post-Ubuntu-0ubuntu122.04.1)
OpenJDK 64-Bit Server VM (build 11.0.19+7-post-Ubuntu-0ubuntu122.04.1, mixed mode, sharing)
root@ip-172-31-36-251:~# mvn --version
Apache Maven 3.6.3
Maven home: /usr/share/maven
Java version: 11.0.19, vendor: Ubuntu, runtime: /usr/lib/jvm/java-11-openjdk-amd64
Default locale: en, platform encoding: UTF-8
OS name: "linux", version: "5.19.0-1025-aws", arch: "amd64", family: "unix"
root@ip-172-31-36-251:~# jenkins --version
2.401.1
root@ip-172-31-36-251:~# docker --version
Docker version 24.0.2, build cb74dfc
root@ip-172-31-36-251:~# ansible --version
ansible [core 2.14.6]
  config file = /etc/ansible/ansible.cfg
  configured module search path = ['/root/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  ansible collection location = /root/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/bin/ansible
  python version = 3.10.6 (main, Mar 10 2023, 10:55:28) [GCC 11.3.0] (/usr/bin/python3)
  jinja version = 3.0.3
  libyaml = True
root@ip-172-31-36-251:~#
```

i-04b420ff6b0a3db72 (Capstone\_Project\_Development)

PublicIPs: 52.66.213.147 PrivateIPs: 172.31.36.251

## 1.d - Password less authentication

Password less authentication b/w 2 servers ( capstone\_project\_development (and) deployment\_test\_server)

On capstone\_project\_development instacne

Do command: ssh-keygen

```
generating public/private rsa key pair.
Enter file in which to save the key (/home/ubuntu/.ssh/id_rsa):
/home/ubuntu/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ubuntu/.ssh/id_rsa
Your public key has been saved in /home/ubuntu/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:dTOLZQqBjiD62h/CdZPwb1mvsEz7Mkc25pcZdtR6TrI ubuntu@ip-172-31-53-60
The key's randomart image is:
+----[RSA 3072]-----+
|          .          |
| . . . . .          |
|.. ..o . . * .      |
| . . .o.. o * +. .   |
| . . = S + . . .    |
| . . . o o=.o + o    |
| oo .   *= o.= *    |
| . . . +o+o.+ E .    |
| . .   +=o.         |
+-----+-----+

```

It will generate public and private keys as below

```
ubuntu@ip-172-31-53-60:~/ansible$ ls /home/ubuntu/.ssh/
authorized_keys  id_rsa  id_rsa.pub  known_hosts
ubuntu@ip-172-31-53-60:~/ansible$
```

Copy the id\_rsa.pub content, paste it in the deployment\_test\_server authotized keys file

On deployment\_test\_server instacne

Do command: ssh-keygen

```

ubuntu@ip-172-31-62-28:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ubuntu/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ubuntu/.ssh/id_rsa
Your public key has been saved in /home/ubuntu/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:AFIPl190A+Zj2rUDuoNXoW8BUVp4QwkkudTphdDLbmw ubuntu@ip-172-31-62-28
The key's randomart image is:
+---[RSA 3072]-----+
|  ..+ +B+OBoo  |
|  . =+.***. .  |
|  .O=O=*..  |
|  ..=B = .  |
|  oS + o  |
|  .E+ . .  |
|  .O+ o  |
|  . o  |
|  |
+-----[SHA256]-----+
ubuntu@ip-172-31-62-28:~$ █

```

```

ubuntu@ip-172-31-62-28:~$ ls ~/.ssh/
authorized_keys  id_rsa  id_rsa.pub
ubuntu@ip-172-31-62-28:~$ vim ~/.ssh/authorized_keys █

```

**Copy** the id\_rsa.pub content in capstone\_project\_development.

**paste** it in the deployment\_test\_server **authorotized\_keys** file

## 1.e ansible setup

On capstone\_project\_development instacne

Do command: sudo su –

Cd /ect/ansible

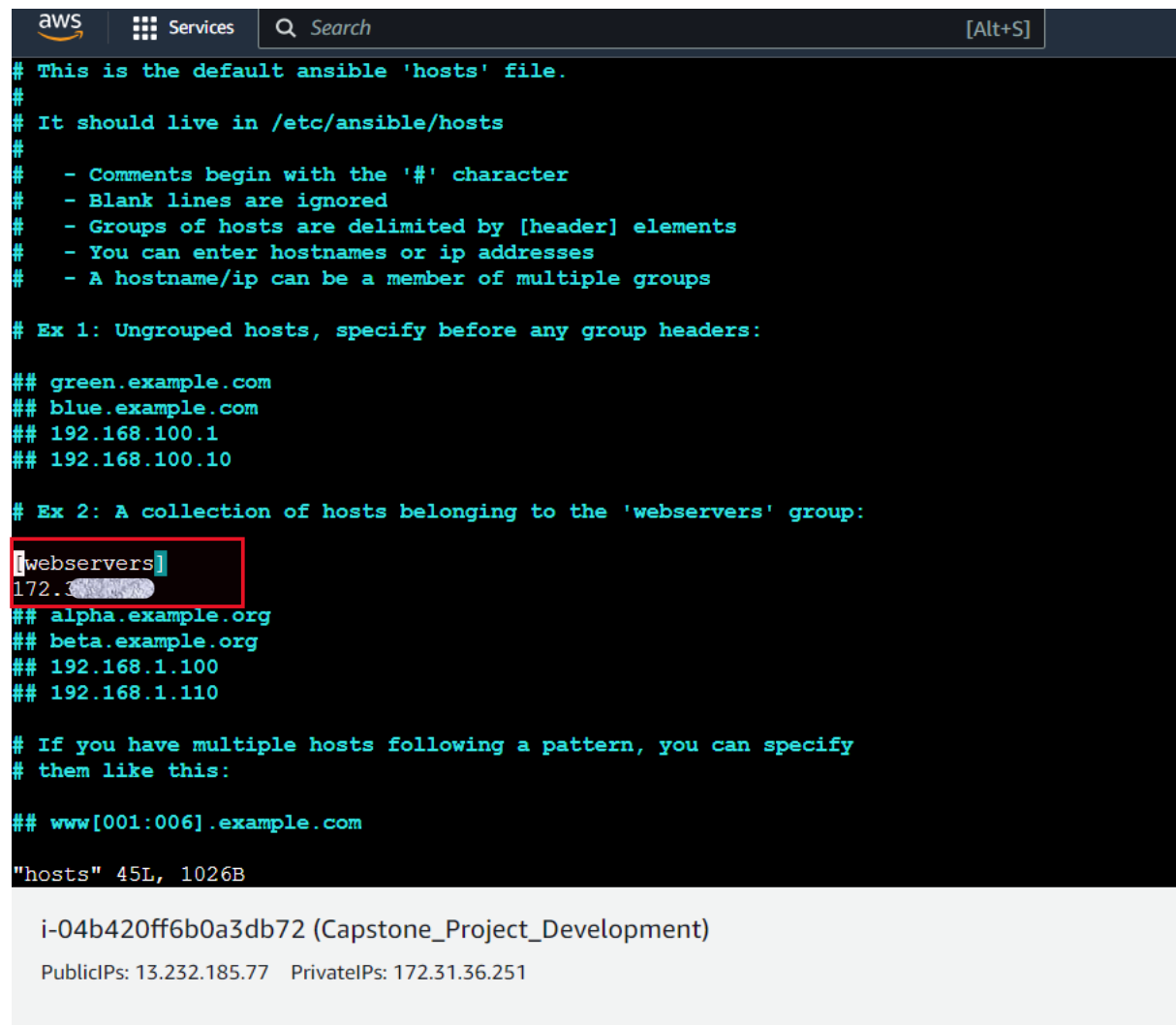
```

root@ip-172-31-36-251:/etc# cd /etc
r AWS Console Home 5-251:/etc# cd ansible/
root@ip-172-31-36-251:/etc/ansible# ls
ansible.cfg  hosts  roles
root@ip-172-31-36-251:/etc/ansible# █

```

Command:- Vi hosts

Insert the deployment server(deployment\_test\_server) ip



```
aws Services Search [Alt+S]
# This is the default ansible 'hosts' file.
#
# It should live in /etc/ansible/hosts
#
# - Comments begin with the '#' character
# - Blank lines are ignored
# - Groups of hosts are delimited by [header] elements
# - You can enter hostnames or ip addresses
# - A hostname/ip can be a member of multiple groups
#
# Ex 1: Ungrouped hosts, specify before any group headers:
## green.example.com
## blue.example.com
## 192.168.100.1
## 192.168.100.10
#
# Ex 2: A collection of hosts belonging to the 'webserver' group:
[webserver]
172.31.36.251
## alpha.example.org
## beta.example.org
## 192.168.1.100
## 192.168.1.110
#
# If you have multiple hosts following a pattern, you can specify
# them like this:
## www[001:006].example.com
"hosts" 45L, 1026B

i-04b420ff6b0a3db72 (Capstone_Project_Development)
PublicIPs: 13.232.185.77 PrivateIPs: 172.31.36.251
```

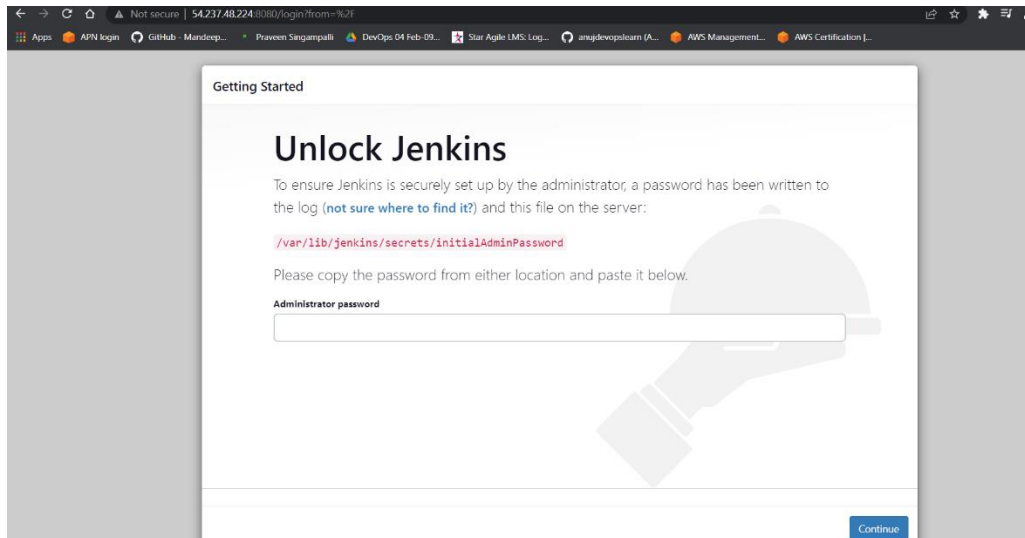
## 1.f Jenkins setup

Setting up the Jenkins (default running on 8080)

User- vams\*\*\*\*\*

Pas- pas\*\*\*\*\*

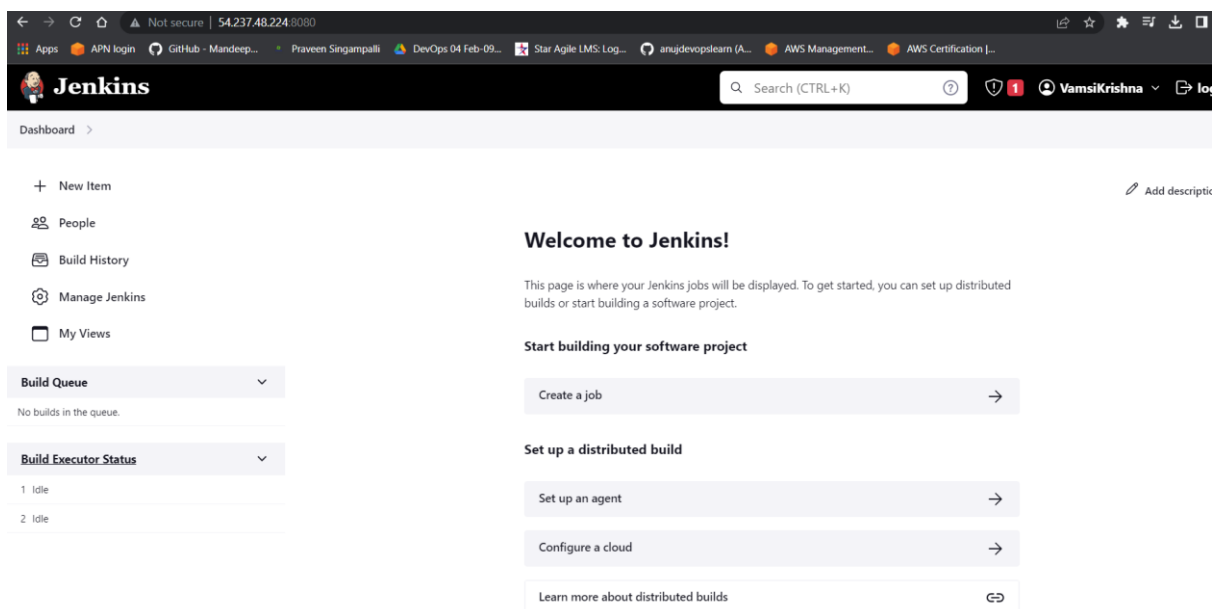
We can access Jenkins using http://IP :8080



### Command :

Sudo cat /var/lib/Jenkins/secret/initialAdminPassword

After completing the initial setup



1.e

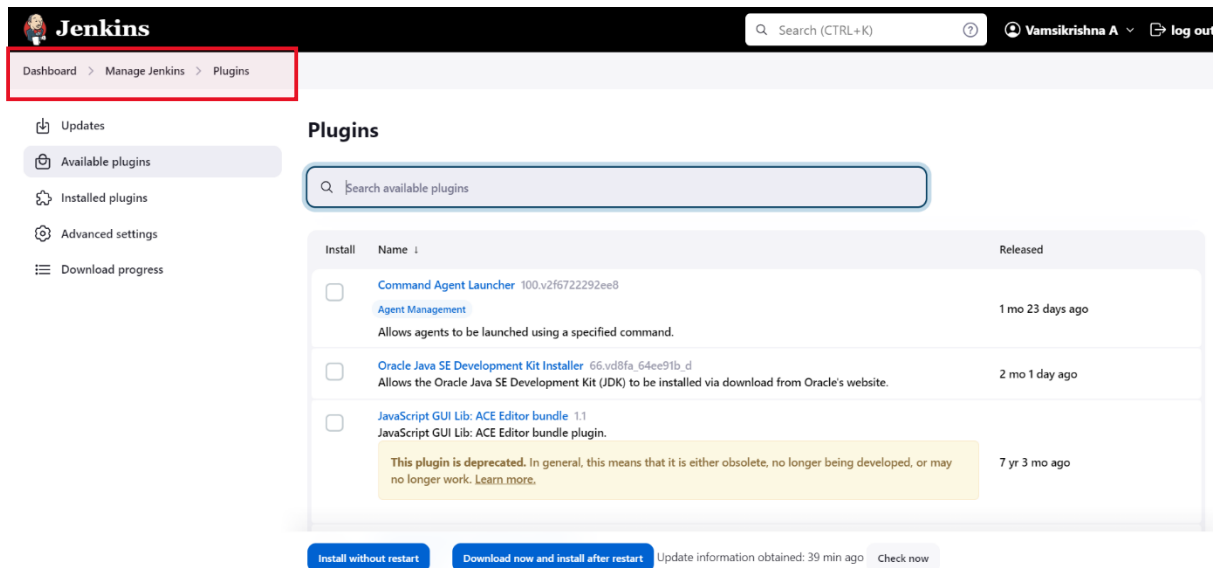
### Plugin installation :



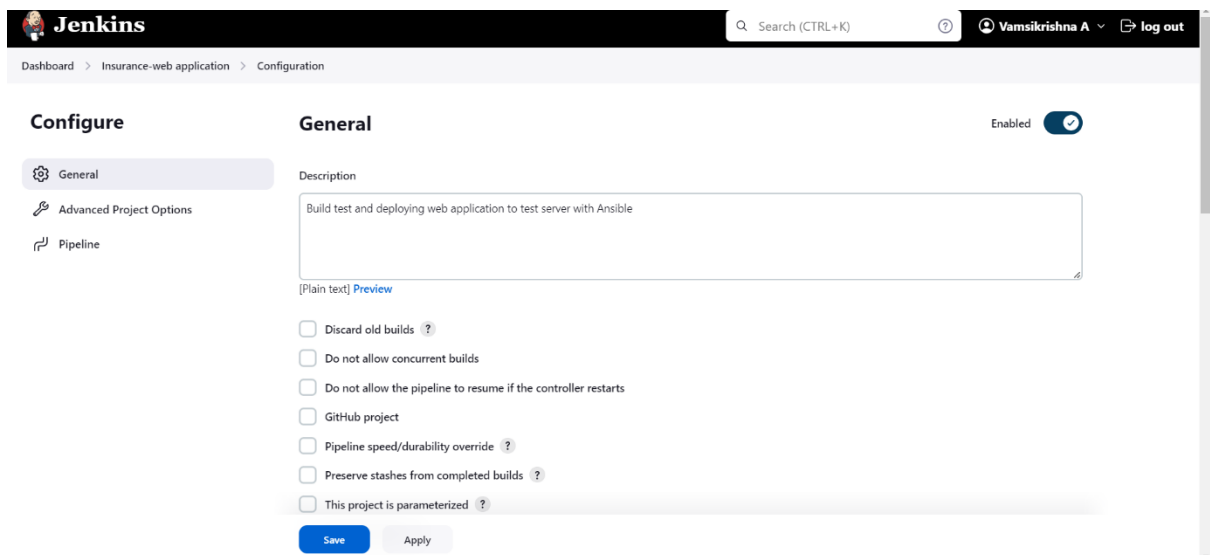
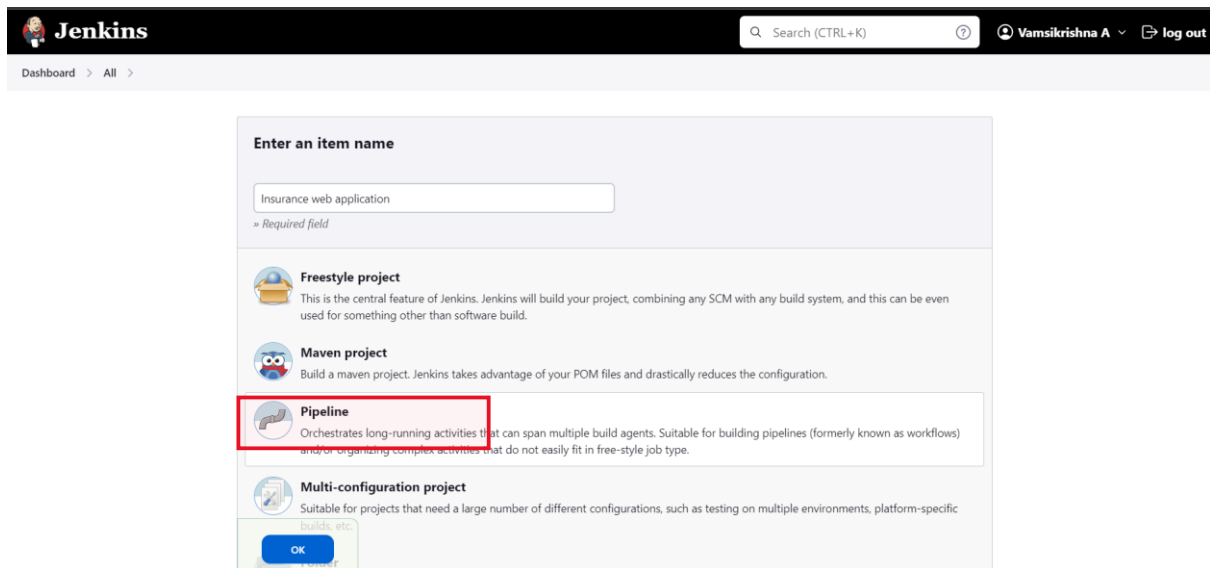
For integration of setup install the following plugins

**Path** – dashboard->manage Jenkins-> plugins->available plugins

- Ansible
- Html publisher
- Docker
- Git
- Maven

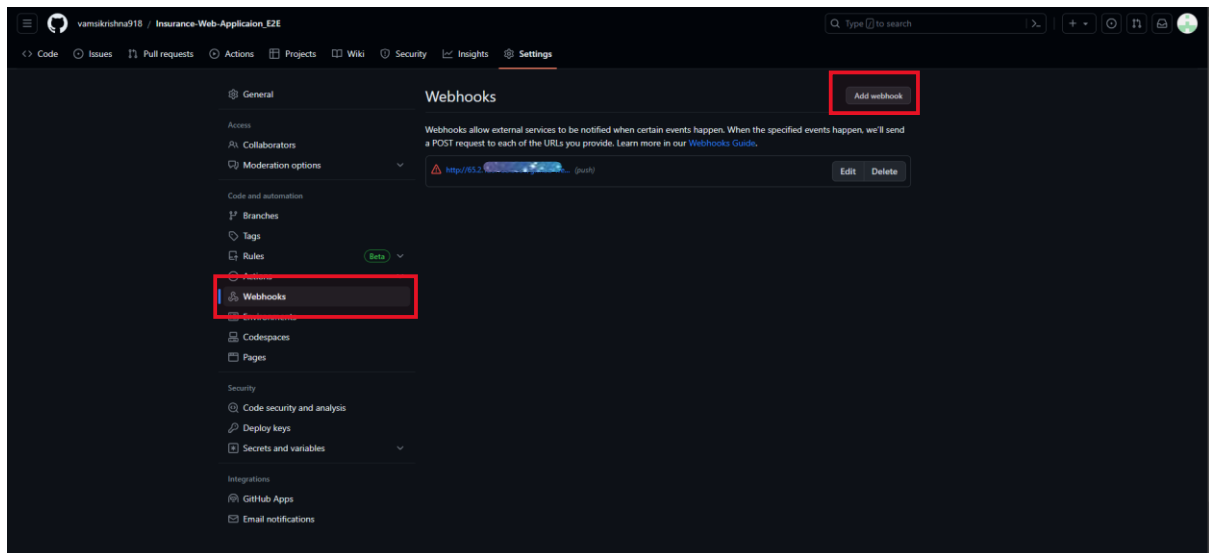


## Step 2: Creating job

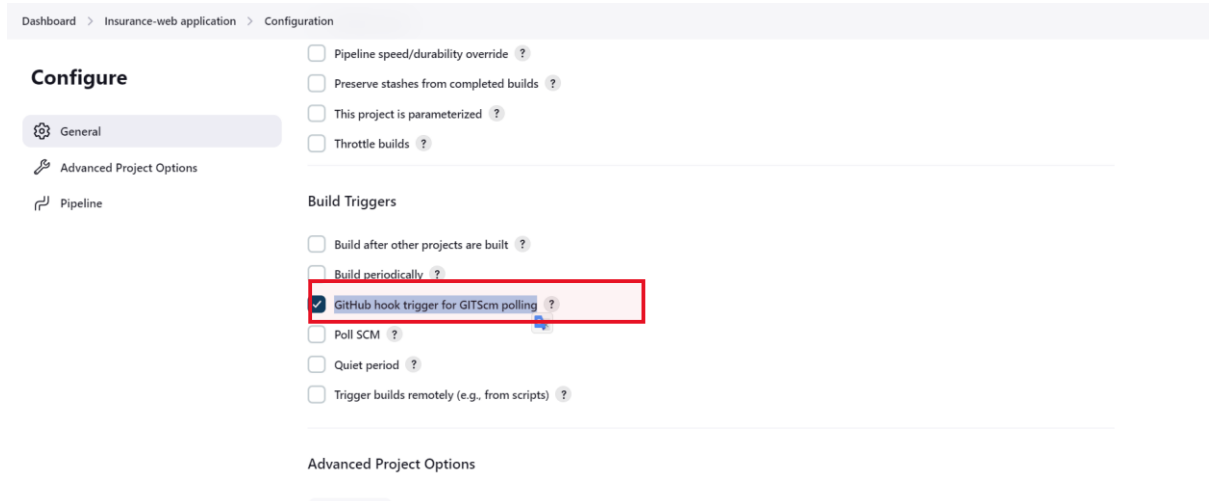


**Setting up the webhook triggers-** (The Jenkins will always watches the git repository if any commit happens it will automatically Runs the Jenkins job.)

GitHub -> repo-> settings-> webhooks-> add webhook -> add the URL of Jenkins



And in Jenkins check the **GitHub hook trigger for GITScm polling**



Create pipeline with the following stages:

1. Code checkout –  
checking out the code repo in **GitHub**  
use **git:Git** as sample step in syntax generator
2. Build –  
building the code, we have checkout with building tools like **maven**
3. publish the report-  
storing the generated reports in a directory  
use **publishHTML:publishHTML reports** as sample step in syntax generator
4. Image prune-  
docker image prune command allows you to clean up unused images
5. Build Docker image-  
building the docker image with Dockerfile  
#Refer [Docker file](#) in repository
6. Push Docker image to Hub-  
pushing the created image to docker registry ( Docker Hub)  
use **withCredentials:Bind credentials to variables** as sample step in syntax generator
7. deploying to Test server –  
containerizing the image and deploying it into all hosts servers using Ansible.  
Use **ansiblePlaybook:Invoke an ansible Playbook** as sample step in syntax generator  
#Refer [ansible-playbook file](#) in repository

#refer the [Jenkins file](#) from git repo

Note- we have used Declarative pipeline syntax,

use **syntax generator/snippet generator** to generate the syntax as mentioned in steps.

```
pipeline{
    agent any

    stages{
        stage("Code checkout"){
            steps{
                checkout scmGit(branches: [[name: '*/main']], extensions: [],
userRemoteConfigs: [[url: 'https://github.com/vamsikrishna918/Insurance-
Web-Applicaion_E2E']])
            }
        }
        stage("Build"){
            steps{
                echo "*****building with maven*****"
                sh "'mvn clean package '"
            }
        }
        stage("publish the report")
```

```

{
  steps{
    echo "generating test reports"

    publishHTML([allowMissing: false, alwaysLinkToLastBuild: false, keepAll:
false, reportDir: '/var/lib/jenkins/workspace/insureme project/target/surefire-
reports', reportFiles: 'index.html', reportName: 'HTML Report', reportTitles: '',
useWrapperFileDirectly: true])
  }
}

stage("Image prune"){
  steps{
    echo "*****deleting the previous images*****"
    sh ' docker image prune -af '

  }
}

stage("Build Docker image"){
  steps{
    script {
      echo "*****Creating Docker image*****"
      sh 'docker build -t vamsi12358/insureme .'
      sh 'docker tag vamsi12358/insureme vamsi12358/insuremeapp:v7'
    }
  }
}

```

```

stage("Push Docker image to Hub"){
  steps{
    script {
      echo "****Pushing Docker image to Hub****"

      withCredentials([string(credentialsId: 'dockercreds', variable:
'dockerhubpwd'))] {
        sh "docker login -u vamsi12358 -p ${dockerhubpwd} docker.io"
        sh 'docker push vamsi12358/insuremeapp:v7'
      }

    }
  }
}

stage("deploying to Test server"){
  steps{
    script {
      echo "****Deploying Application to Test server****"


      ansiblePlaybook become: true, credentialsId: 'ansiblecreds',
disableHostKeyChecking: true, inventory: '/etc/ansible/hosts', playbook:
'ansible-playbook.yml'


    }
  }
}
}


```

## Credentials setup

- Docker
- Deploying server(deployment\_test\_server) ssh key setup









 **Jenkins**

Search (CTRL+K) 


Vamsikrishna A  [log out](#)

[Dashboard](#) > [Manage Jenkins](#) > [Credentials](#)

### Credentials

T	P	Store	Domain	ID	Name
		System	(global)	Dockerhub	vamsi12358/***** (Dockerhub)
		System	(global)	dockercreds	dockercreds
		System	(global)	testcreds	jenkins (test server pem file)
		System	(global)	ansiblecreds	ubuntu (ansible credentials)

### Stores scoped to Jenkins

P	Store	Domains
	System	(global)

Icons: S M **L**



# Jenkins pipeline

Dashboard > Insurance-web application > Configuration

## Configure

- General
- Advanced Project Options
- Pipeline**

### Pipeline

Definition

Pipeline script

```
Script ?
1 pipeline{
2   agent any
3
4   stages{
5     stage("Code checkout"){
6       steps{
7         checkout scmGit(branches: [[name: '**/main']], extensions: [], userRemoteConfigs: [[url: 'https://github.com/vamsikrishna
8       ]
9     }
10    stage("Build"){
11      steps{
12        echo "****building with maven****"
13        sh "mvn clean package"
14      }
15    }
16    stage("publish the report"){
17      steps{
18        echo "generating test reports"
19        publishHTML([allowMissing: false, alwaysLinkToLastBuild: false, keepAll: false, reportDir: '/var/lib/jenkins/workspace/insureme
20      ])
21    }
22    stage("Image prune"){
23      steps{
24        echo "****deleting the previous images****"
25        sh "docker image prune -af"
26      }
27    }
28  }
29 }
```

Save Apply

Dashboard > Insurance-web application > Configuration

## Configure

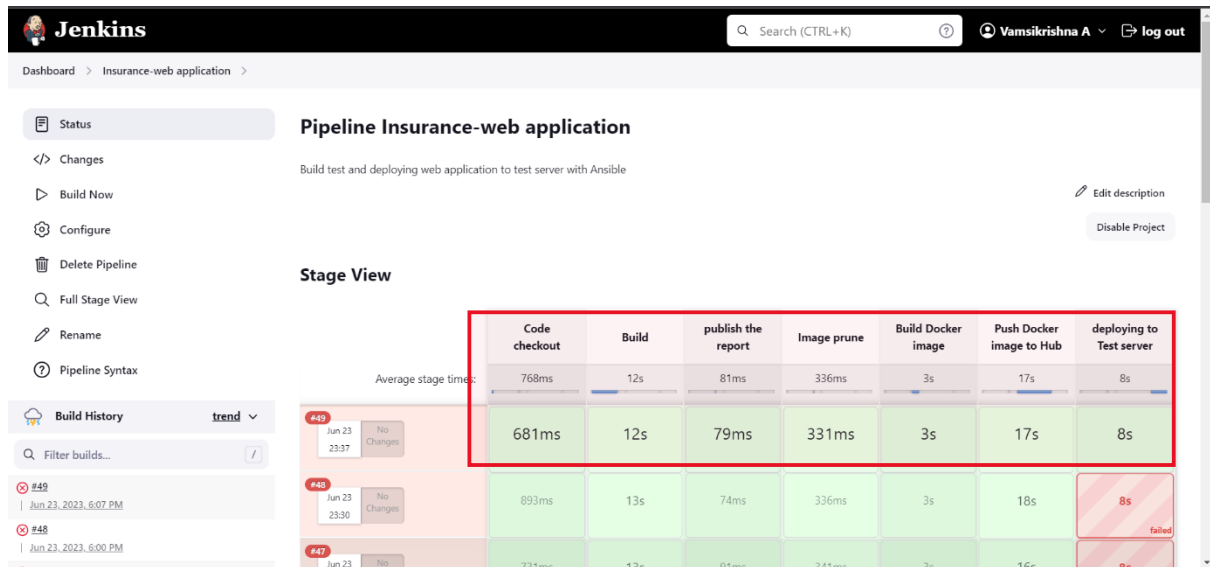
- General
- Advanced Project Options
- Pipeline**

```
24 stage("Image prune"){
25   steps{
26     echo "****deleting the previous images****"
27     sh "docker image prune -af"
28   }
29 }
30
31
32
33 stage("Build Docker image"){
34   steps{
35     script {
36       echo "****Creating Docker image****"
37       sh "docker build -t vamsi12358/insureme ."
38       sh "docker tag vamsi12358/insureme vamsi12358/insuremeapp:v7"
39     }
40   }
41 }
42 stage("Push Docker image to Hub"){
43   steps{
44     script {
45       echo "****Pushing Docker image to Hub****"
46
47       withCredentials([string(credentialsId: 'dockercrds', variable: 'dockerhubpwd')]) {
48         sh "docker login -u vamsi12358 -p ${dockerhubpwd} docker.io"
49         sh "docker push vamsi12358/insuremeapp:v7"
50       }
51     }
52   }
53 }
54 stage("deploying to Test server"){
55   steps{
56     script {
57       echo "****Deploying Application to Test server****"
58       ansiblePlaybook become: true, credentialsId: 'ansiblecrds', disableHostKeyChecking: true, inventory: '/etc/ansible/hos
59     }
60   }
61 }
```

☒ Use Groovy Sandbox ?

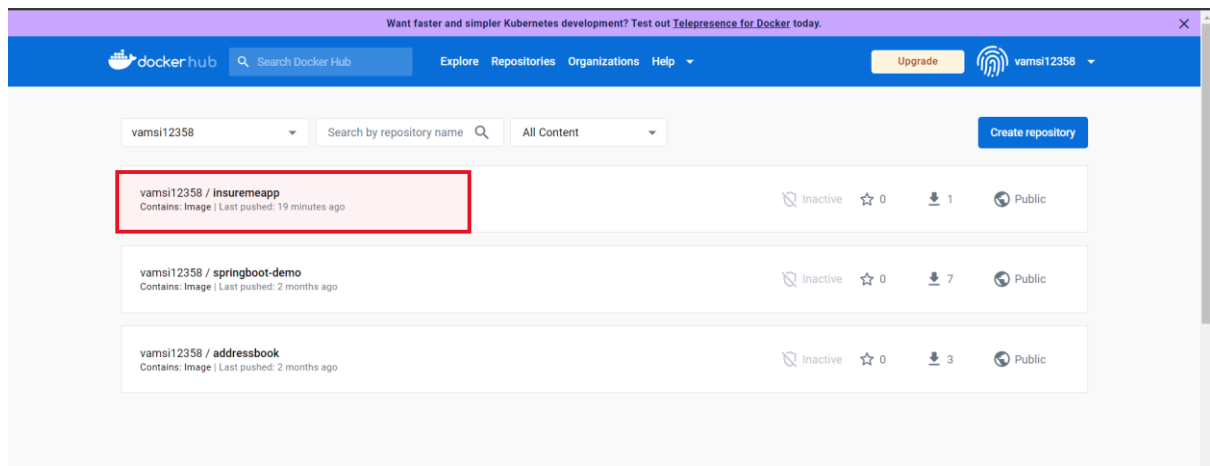
Save Apply

## Build the job-



## Docker hub-

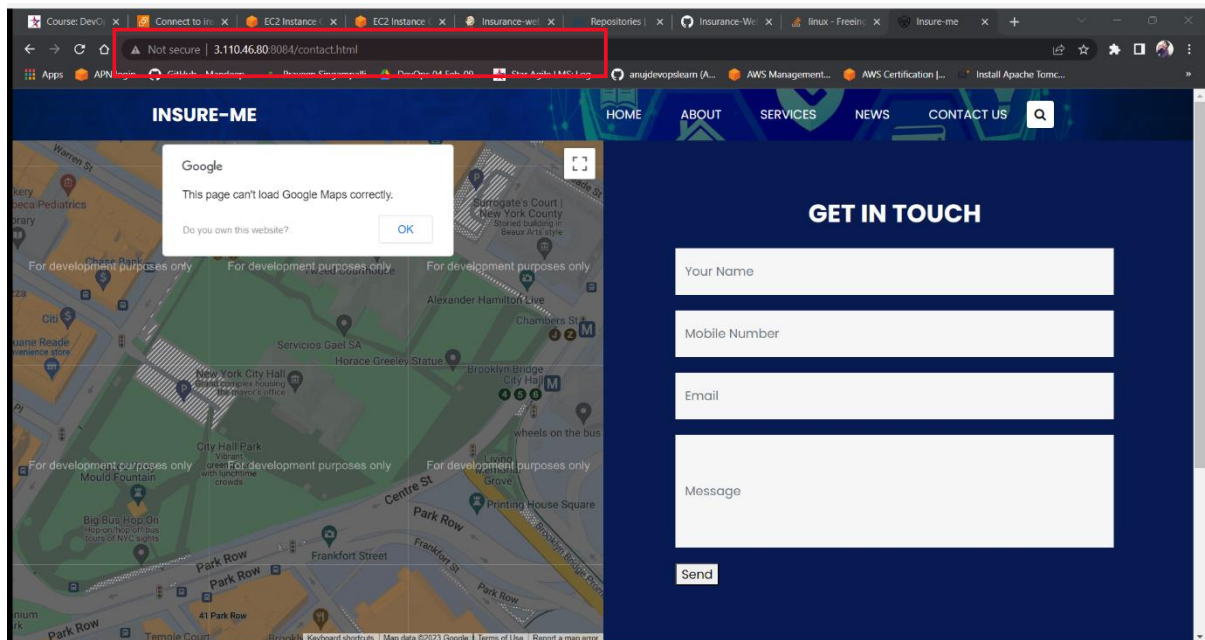
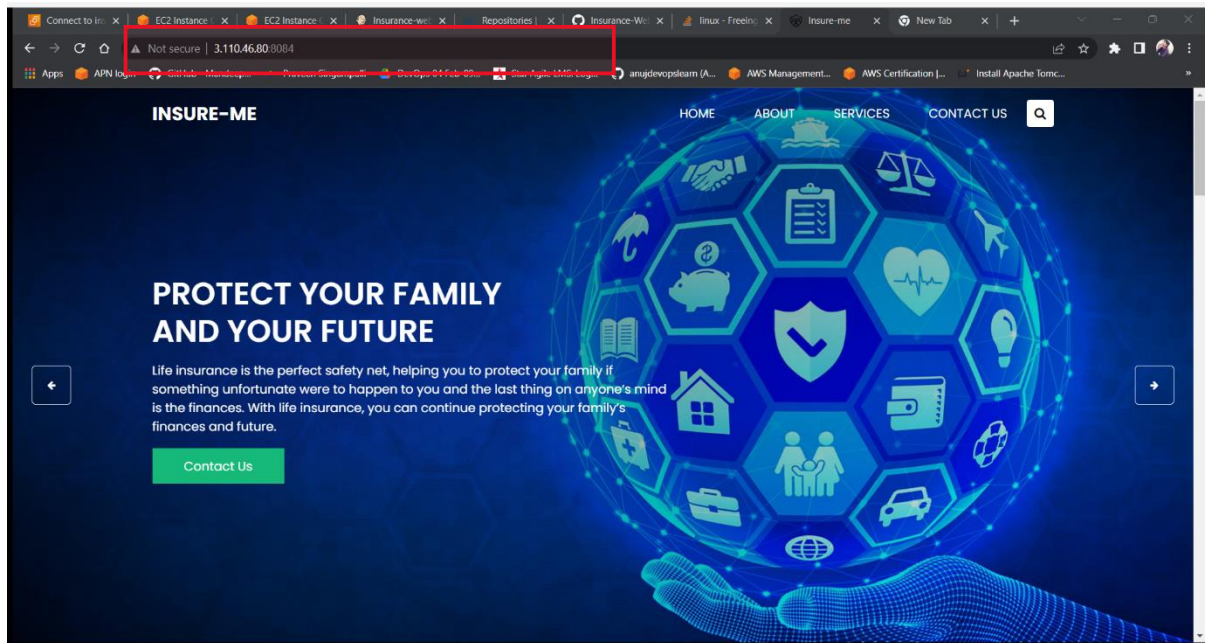
<https://hub.docker.com/>



## Deployed Application on test server

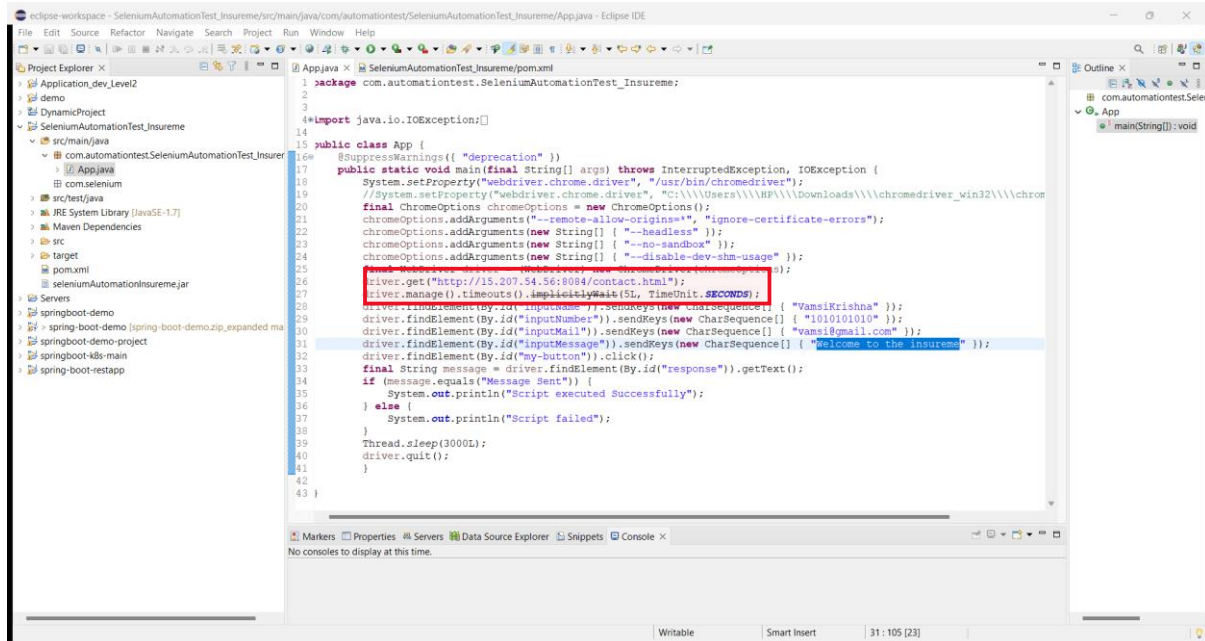
Accessing the web-application on test server

Deployment\_test\_server public ip:8084/

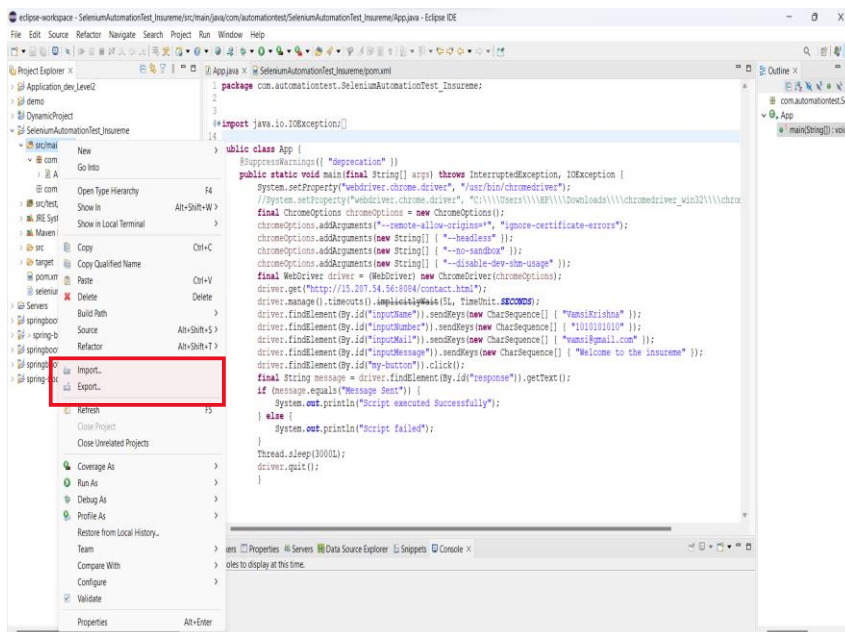


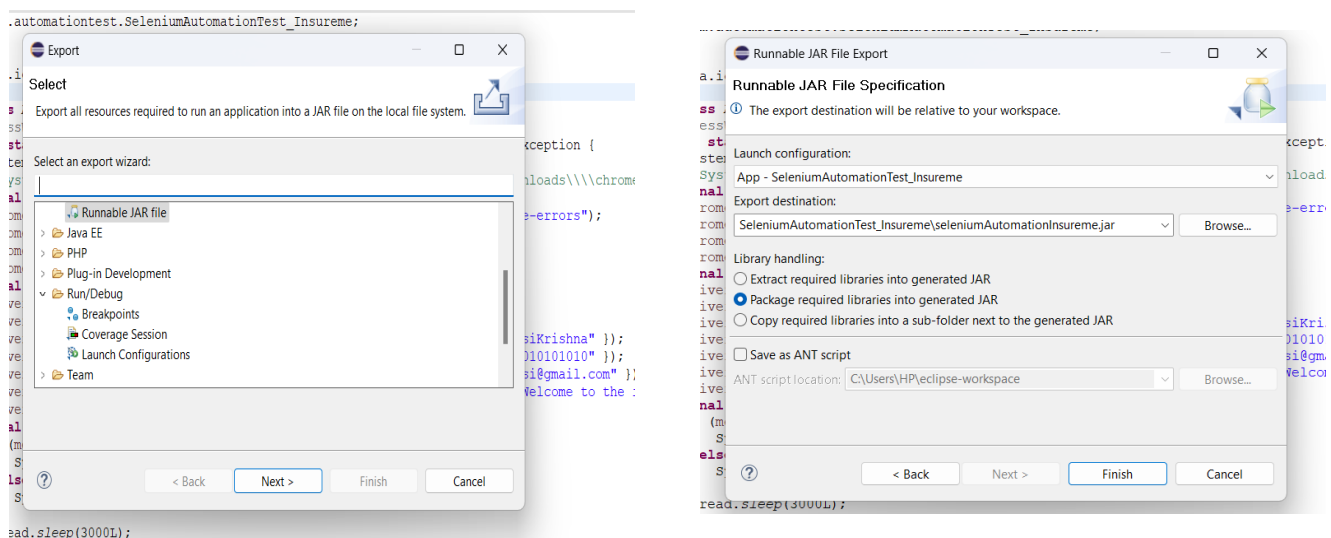
# Test -Automation with selenium

Refer - <https://github.com/vamsikrishna918/Selenium-TestAutomation-Insureme>



## Export to Runnable jar





Push the jar to git hub, we can crate a job and use in pipeline

## Installing the Chrome driver in our ubuntu instance

(Capstone\_Project\_Development)

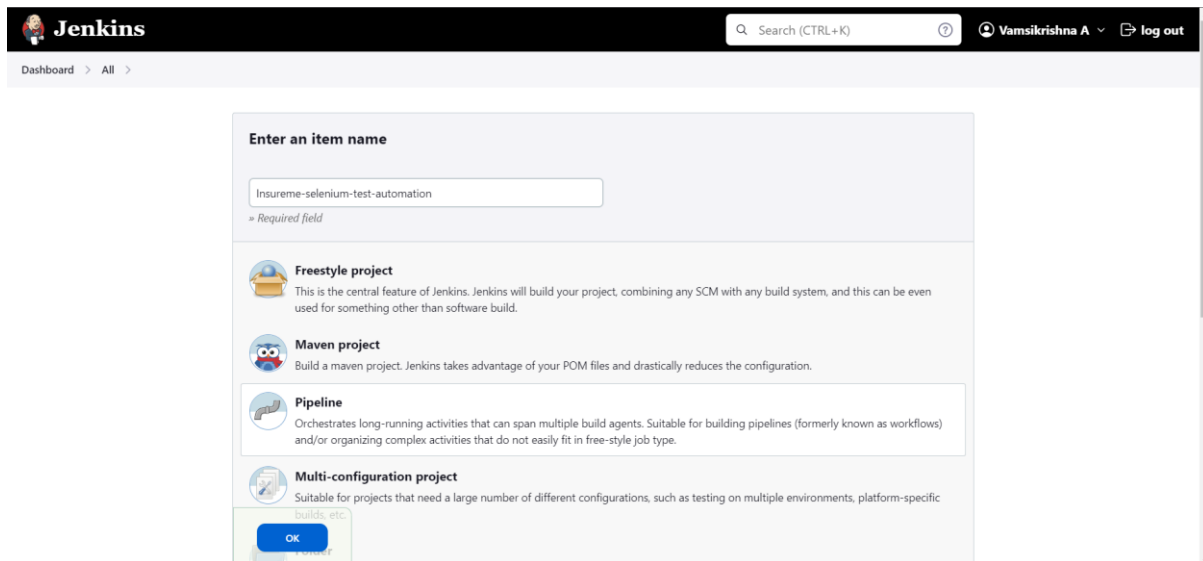
### On Capstone\_Project\_Development instance-

Run the below commands and install.

- `sudo apt install -y unzip xvfb libxi6 libgconf-2-4`
- `sudo curl -sS -o - https://dl-ssl.google.com/linux/linux_signing_key.pub | apt-key add`
- `sudo bash -c "echo 'deb [arch=amd64] http://dl.google.com/linux/chrome/deb/stable main' >> /etc/apt/sources.list.d/google-chrome.list"`
- `sudo apt -y update`
- `sudo apt -y install google-chrome-stable`
- `google-chrome --version`
- `wget https://chromedriver.storage.googleapis.com/114.0.5735.90/chromedriver_linux64.zip`
- `ls`
- `unzip chromedriver_linux64.zip`
- `ls`
- `sudo mv chromedriver /usr/bin/chromedriver`

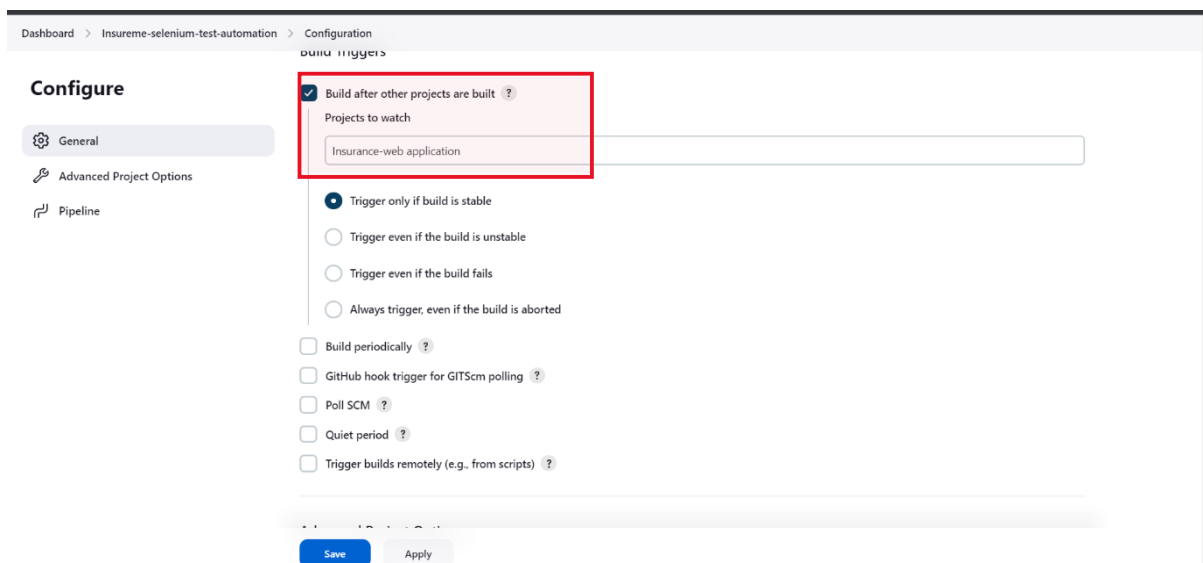
- sudo chown root:root /usr/bin/chromedriver
- sudo chmod +x /usr/bin/chromedriver
- which chromedriver

## Creating a job for automation test-



The image shows the Jenkins 'Create new item' dialog. At the top, there's a search bar and a user profile 'Vamsikrishna A' with a 'log out' button. Below the header, the breadcrumb is 'Dashboard > All >'. The main section is titled 'Enter an item name' and contains a text input field with the value 'Insureme-selenium-test-automation'. Below the input field, it says '» Required field'. There are four project type options: 'Freestyle project', 'Maven project', 'Pipeline', and 'Multi-configuration project'. Each option has a brief description. At the bottom left of the dialog is an 'OK' button.

Setup the build trigger, which the automation job builds after the Insure me application deployed on test server(deployment\_test\_server).



The image shows the Jenkins 'Configure' page for the job 'Insureme-selenium-test-automation'. The breadcrumb is 'Dashboard > Insureme-selenium-test-automation > Configuration'. On the left, there's a 'Configure' section with three tabs: 'General', 'Advanced Project Options', and 'Pipeline'. The 'General' tab is selected. In the 'Build after other projects are built' section, the checkbox is checked, and the 'Projects to watch' field contains 'Insurance-web application'. Below this, there are four radio button options for triggering: 'Trigger only if build is stable' (selected), 'Trigger even if the build is unstable', 'Trigger even if the build fails', and 'Always trigger, even if the build is aborted'. At the bottom, there are several unchecked checkboxes for other triggers: 'Build periodically', 'GitHub hook trigger for GITScm polling', 'Poll SCM', 'Quiet period', and 'Trigger builds remotely (e.g., from scripts)'. At the very bottom are 'Save' and 'Apply' buttons.

Pipeline script-

```
pipeline{
    agent any

    stages{
        stage("Test auomation Code checkout"){
            steps{
                git branch: 'main', url: 'https://github.com/vamsikrishna918/Selenium-TestAutomation-Insureme'
            }
        }
        stage("Executing jar"){
            steps{
                sh 'sudo java -jar seleniumAutomationInsureme.jar'
            }
        }
    }
}
```

The screenshot shows the Jenkins 'Configure' page for a pipeline named 'Insureme-selenium-test-automation'. The 'Pipeline' tab is selected. Under 'Definition', 'Pipeline script' is chosen. A 'Script' section contains a Groovy script for a pipeline with two stages: 'Code checkout' and 'Executing jar'. The 'Code checkout' stage uses the 'git' step to checkout the 'main' branch from a specific GitHub repository. The 'Executing jar' stage uses the 'sh' step to run a Java command. Below the script, the 'Use Groovy Sandbox' checkbox is checked. At the bottom, there are 'Save' and 'Apply' buttons.

Dashboard > Insureme-selenium-test-automation > Configuration

### Configure

- General
- Advanced Project Options
- Pipeline

**Pipeline**

Definition

Pipeline script

Script ?

```
1 pipeline{
2   agent any
3
4   stages{
5     stage("Code checkout"){
6       steps{
7         git branch: 'main', url: 'https://github.com/vamsikrishna918/Selenium-TestAutomation-Insureme'
8       }
9     }
10    stage("Executing jar"){
11      steps{
12        sh 'sudo java -jar seleniumAutomationInsureme.jar'
13      }
14    }
15  }
16 }
17 }
```

☒ Use Groovy Sandbox ?

[Pipeline Syntax](#)

Save Apply

Search (CTRL+K)

Vamsikrishna A

log out

Dashboard > Insureme-selenium-test-automation >

Status

</> Changes

Build Now

Configure

Delete Pipeline

Full Stage View

Rename

Pipeline Syntax

Build History

trend

Filter builds...

#1

Jun 24, 2023 9:52 AM

Atom feed for all

Atom feed for failures

Pipeline Insureme-selenium-test-automation

automation test job for Insureme application

Edit description

Disable Project

Stage View

Average stage times:

(Average full run time: ~21s)

Code checkout

11s

Executing jar

9s

#1

Jun 24 15:22

No Changes

11s

9s

Permalinks

Last build (#1), 42 min ago

Last stable build (#1), 42 min ago

Last successful build (#1), 42 min ago

insrd > Insureme-selenium-test-automation > #1

```

[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Executing jar)
[Pipeline] sh
+ sudo java -jar seleniumAutomationInsureme.jar
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
Starting ChromeDriver 114.0.5735.90 (386bc89e8f4f2e025eddae123f36f6263096ae49-refs/branch-heads/5735@(#1052)) on port 51918
Only local connections are allowed.
Please see https://chromedriver.chromium.org/security-considerations for suggestions on keeping ChromeDriver safe.
ChromeDriver was started successfully.
Jun 24, 2023 9:53:04 AM org.openqa.selenium.remote.ProtocolHandshake createSession
INFO: Detected dialect: W3C
Jun 24, 2023 9:53:04 AM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch
WARNING: Unable to find an exact match for CDP version 114, so returning the closest version found: a no-op implementation
Jun 24, 2023 9:53:04 AM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch
INFO: Unable to find CDP implementation matching 114.
Jun 24, 2023 9:53:04 AM org.openqa.selenium.chromium.ChromiumDriver lambda$new$3
WARNING: Unable to find version of CDP to use for . You may need to include a dependency on a specific version of the CDP using something similar to
org.seleniumhq.selenium:selenium-devtools-v80-4-0-0, where the ver
the version number of the artifact is the same as Selenium's.
Script executed Successfully
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS

```

Search (CTRL+K)

Vamsikrishna A

log out

Dashboard >

+ New Item

People

Build History

Project Relationship

Check File Fingerprint

Manage Jenkins

My Views

Build Queue (1)

Insureme-selenium-test-automation

Build Executor Status

1 Idle

2 Idle

All

+

S

W

Name ↓

Last Success

Last Failure

Last Duration

✗

☁

Insureweb application

17 hr #43

1 min 34 sec #59

1 min 49 sec

▶

✓

☀

Insureme-selenium-test-automation

49 min #1

N/A

21 sec

▶

Icon: S M L

Icon legend

Atom feed for all

Atom feed for failures

Atom feed for just latest builds

REST API

Jenkins 2.401.1

24 | Page



The screenshot shows the Jenkins web interface. At the top, there's a search bar and user information for 'Vamsikrishna A'. The main header shows the pipeline name 'Pipeline Insureme-selenium-test-automation' with a subtitle 'automation test job for insureme application'. On the left, a sidebar contains navigation links: Status, Changes, Build Now, Configure, Delete Pipeline, Full Stage View, Rename, and Pipeline Syntax. Below this is the 'Build History' section showing a single build (#1) from June 24, 2023, at 9:52 AM, with links for 'Atom feed for all' and 'Atom feed for failures'. The main content area is titled 'Stage View' and displays a table of stage execution times:

	Code checkout	Executing jar
Average stage times: (Average full run time: ~21s)	11s	9s
#1 Jun 24 15:22 No Changes	11s	9s

Below the table, there's a 'Permalinks' section with three links: 'Last build (#1), 42 min ago', 'Last stable build (#1), 42 min ago', and 'Last successful build (#1), 42 min ago'. On the right side of the stage view, there are links for 'Edit description' and 'Disable Project'.

Sample executed on local machine-

The screenshot shows a web browser window with the URL '15.207.54.56:8084/contact.html'. The website has a dark blue header with the 'INSURE-ME' logo and navigation links: HOME, ABOUT, SERVICES, NEWS, and CONTACT US. A Google Maps error dialog is displayed over the map area, stating 'This page can't load Google Maps correctly. Do you own this website?' with an 'OK' button. On the right side, there's a 'GET IN TOUCH' section with input fields for 'VamsiKrishna', '1010101010', and 'vamsi@gmail.com'. Below these fields is a text area containing 'Welcome to the insureme' and a 'Send' button. At the bottom of the contact form, it says 'Message Sent'.