

CI/CD Deployment for SpringBoot Application.

(Application Screenshot's)

Developer: Tushar Khillare

■ Version History:

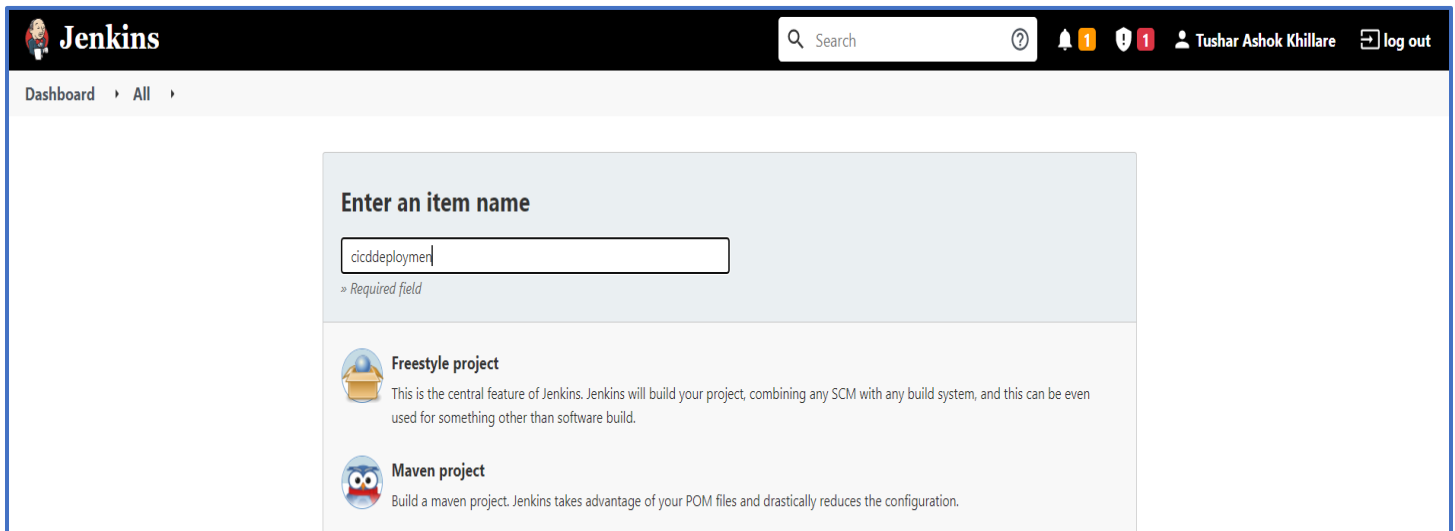
| | | |
|----|-------------|---|
| 1. | Author Name | Tushar Ashok Khillare. |
| 2. | Purpose | Screenshots of Application. |
| 3. | Date | 19 January 2022. |
| 4. | Version | 1.0 |
| 5. | Contact | <u>tusharkhillare2015@gmail.com</u> |

Contents.

| | |
|-------------------------------|---|
| 1. Project Screenshots..... | 3 |
| 2. Appearance of Project..... | 4 |

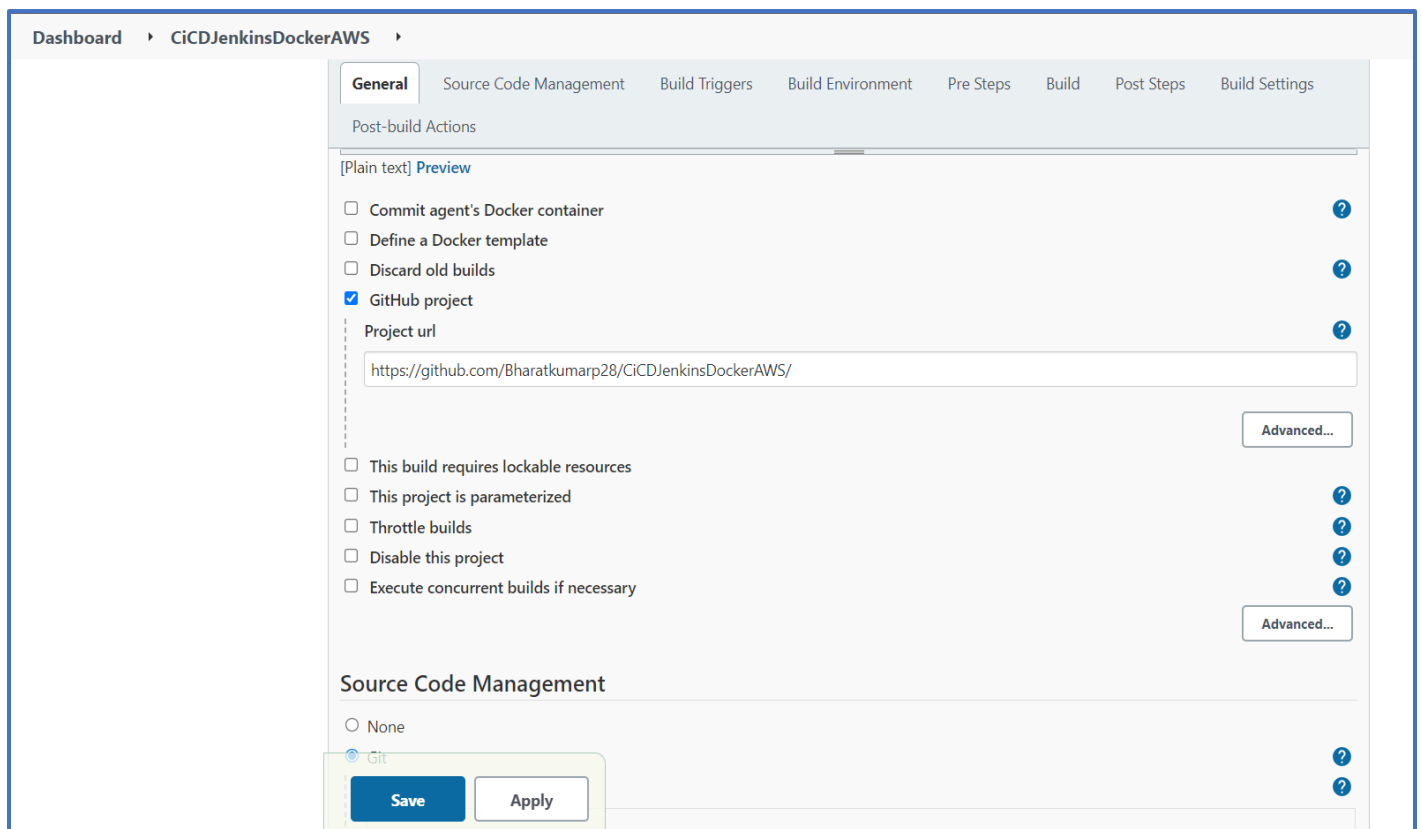
A. Configure Jenkins

- Download and install Jenkins in the local machine.
- Login to Jenkins using the address: <http://localhost:8080/>
- Create a new item in the Jenkins as in the below image:



The image shows the Jenkins 'Enter an item name' dialog. At the top, there's a search bar and a user profile for 'Tushar Ashok Khillare' with a 'log out' button. Below the header, the breadcrumb 'Dashboard > All >' is visible. The main area has a title 'Enter an item name' and a text input field containing 'cicdeploymer'. A small note below the input says '» Required field'. Below the input, there are two project type suggestions: 'Freestyle project' and 'Maven project', each with an icon and a brief description.

- Configure Jenkins by giving GitHub link and Docker credentials as in the below images:



The image shows the Jenkins configuration page for a project named 'CiCDJenkinsDockerAWS'. The breadcrumb is 'Dashboard > CiCDJenkinsDockerAWS >'. The 'General' tab is selected, showing 'Post-build Actions'. Under this section, there are several checkboxes: 'Commit agent's Docker container', 'Define a Docker template', 'Discard old builds', 'GitHub project' (which is checked), and 'This build requires lockable resources'. The 'Project url' field is filled with 'https://github.com/Bharatkumarp28/CiCDJenkinsDockerAWS/'. Below these, there are more checkboxes: 'This project is parameterized', 'Throttle builds', 'Disable this project', and 'Execute concurrent builds if necessary'. At the bottom, the 'Source Code Management' section shows 'Git' selected. There are 'Save' and 'Apply' buttons at the bottom left. On the right side, there are 'Advanced...' buttons for the 'GitHub project' and 'Execute concurrent builds if necessary' options.

Dashboard > cicddployment >

GeneralSource Code ManagementBuild TriggersBuild EnvironmentPre StepsBuildPost StepsBuild Settings

Post-build Actions

None

Git

Repositories

Repository URL

https://github.com/tuskhillare/Phase-5-CI-CD-Deployment.git

Credentials

tuskhillare/***** (github)Add

Advanced...

Add Repository

Branches to build

Branch Specifier (blank for 'any')

*/main

Add Branch

Repository browser

SaveApply

Dashboard > CiCDJenkinsDockerAWS >

GeneralSource Code ManagementBuild TriggersBuild EnvironmentPre StepsBuildPost StepsBuild Settings

Post-build Actions

Service hook trigger for GitHub polling

Poll SCM

Schedule

⚠ Do you really mean "every minute" when you say "*****"? Perhaps you meant "H*****" to poll once per hour

Would last have run at Saturday, January 22, 2022 2:22:48 PM IST; would next run at Saturday, January 22, 2022 2:22:48 PM IST.

Ignore post-commit hooks

Build Environment

Delete workspace before build starts

Use secret text(s) or file(s)

Abort the build if it's stuck

Add timestamps to the Console Output

Inspect build log for published Gradle build scans

Unleash

With Ant

Post Steps

SaveApply

Post-build Actions

Invoke top-level Maven targets X ?

Maven Version

Maven Install

Goals

install

Advanced...

Docker Build and Publish X ?

Repository Name ?

ponugub416/cicd

Tag

Docker Host URI ?

Server credentials

Save

Apply

Post-build Actions

Docker Build and Publish X ?

Repository Name ?

tuskillare/cicd

Tag

Docker Host URI ?

Server credentials

- none -

Add

Docker registry URL ?

Registry credentials

tuskillare/***** (Dockerhub)

Add

Advanced...

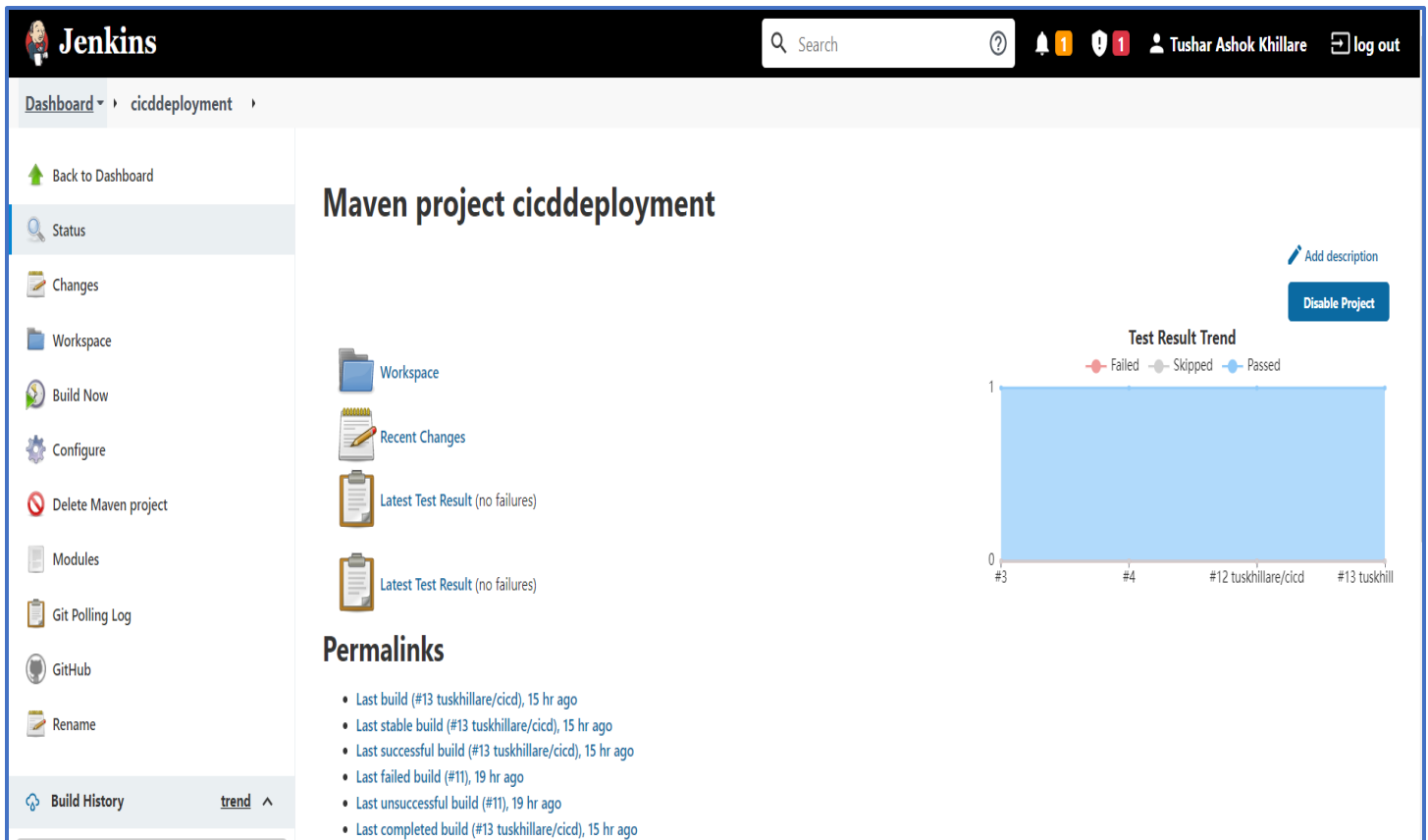
Add pre-build step

Save

Apply

B. Build Docker Image File

- We have configured GitHub file with Docker File included and given the docker hub credential to initiate the docker image file.
- Once done with the configuration of Jenkins apply and save the configs and click on Build Now:



The screenshot shows the Jenkins web interface for a project named 'cicddployment'. The top navigation bar includes the Jenkins logo, a search bar, and user information for 'Tushar Ashok Khillare'. The left sidebar contains various navigation options like 'Back to Dashboard', 'Status', 'Changes', 'Workspace', 'Build Now', 'Configure', 'Delete Maven project', 'Modules', 'Git Polling Log', 'GitHub', and 'Rename'. The main content area is titled 'Maven project cicddployment' and features a 'Test Result Trend' graph showing a 100% pass rate for builds #3, #4, #12, and #13. Below the graph, there are links to 'Workspace', 'Recent Changes', and 'Latest Test Result' (no failures). A 'Permalinks' section lists various build links, including the last build (#13) and the last stable build (#13).

Jenkins Dashboard

Dashboard > cicddployment

Back to Dashboard

Status

Changes

Workspace

Build Now

Configure

Delete Maven project

Modules

Git Polling Log

GitHub

Rename

Build History trend ^

Maven project cicddployment

Add description

Disable Project

Test Result Trend

Failed Skipped Passed

1

0

#3 #4 #12 tuskhillare/cicd #13 tuskhill

Workspace

Recent Changes

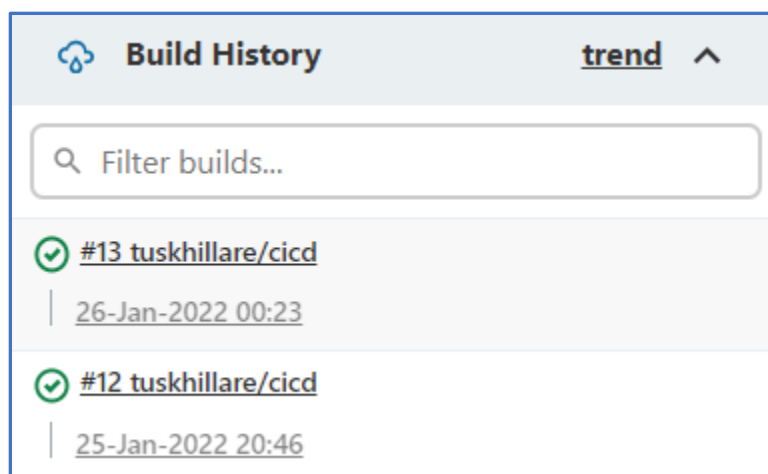
Latest Test Result (no failures)

Latest Test Result (no failures)

Permalinks

- Last build (#13 tuskhillare/cicd), 15 hr ago
- Last stable build (#13 tuskhillare/cicd), 15 hr ago
- Last successful build (#13 tuskhillare/cicd), 15 hr ago
- Last failed build (#11), 19 hr ago
- Last unsuccessful build (#11), 19 hr ago
- Last completed build (#13 tuskhillare/cicd), 15 hr ago

We can see the Build History in the Build history tab:



The screenshot shows the 'Build History' tab in Jenkins. It features a search bar labeled 'Filter builds...'. Below the search bar, there are two build entries, both marked with a green checkmark. The first entry is '#13 tuskhillare/cicd' with a timestamp of '26-Jan-2022 00:23'. The second entry is '#12 tuskhillare/cicd' with a timestamp of '25-Jan-2022 20:46'.

Build History trend ^

Filter builds...

✓ #13 tuskhillare/cicd

26-Jan-2022 00:23

✓ #12 tuskhillare/cicd

25-Jan-2022 20:46

- Check the Console output for the any errors/ build completed log:

Dashboard

cicddployment

#13 tuskhillare/cicd

Back to Project

Status

Changes

Console Output

View as plain text

Edit Build Information

Delete build '#13 tuskhillare/cicd'

Polling Log

Git Build Data

Test Result

Redeploy Artifacts

See Fingerprints

Previous Build

Console Output

Started by an SCM change

Running as SYSTEM

Building in workspace C:\ProgramData\Jenkins\jenkins\workspace\cicddployment

The recommended git tool is: NONE

using credential github

> C:\Program Files\Git\bin\git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\jenkins\workspace\cicddployment\.git # timeout=10

Fetching changes from the remote Git repository

> C:\Program Files\Git\bin\git.exe config remote.origin.url <https://github.com/tuskhillare/Phase-5-CI-CD-Deployment.git> # timeout=10

Fetching upstream changes from <https://github.com/tuskhillare/Phase-5-CI-CD-Deployment.git>

> C:\Program Files\Git\bin\git.exe --version # timeout=10

> git --version # 'git version 2.32.0.windows.2'

using GIT_ASKPASS to set credentials github

> C:\Program Files\Git\bin\git.exe fetch --tags --force --progress -- <https://github.com/tuskhillare/Phase-5-CI-CD-Deployment.git> +refs/heads/*:refs/remotes/origin/* # timeout=10

> C:\Program Files\Git\bin\git.exe rev-parse "refs/remotes/origin/main^{commit}" # timeout=10

Checking out Revision a47487d735f181a1feb6bd6bd15553a28d49a1fcf (refs/remotes/origin/main)

> C:\Program Files\Git\bin\git.exe config core.sparsecheckout # timeout=10

> C:\Program Files\Git\bin\git.exe checkout -f a47487d735f181a1feb6bd6bd15553a28d49a1fcf # timeout=10

Commit message: "Create README.md"

> C:\Program Files\Git\bin\git.exe rev-list --no-walk 6920d78556c997cb283be6f35c67f85adf795612 # timeout=10

[cicddployment] \$ cmd.exe /C "C:\ProgramData\Jenkins\jenkins\tools\hudson.tasks.Maven_MavenInstallation\maven_install\bin\mvn.cmd install && exit %ERRORLEVEL%"

[INFO] Scanning for projects...

[INFO]

[INFO] -----< com.example:CICDDeployemnt >-----

[INFO] Building CICDDeployemnt 0.0.1-SNAPSHOT

[INFO] -----[war]-----

[INFO]

[INFO] --- maven-resources-plugin:3.2.0:resources (default-resources) @ CICDDeployemnt ---

[INFO] Using 'UTF-8' encoding to copy filtered resources

```
( _ |   _ |   _ )   _ |   _ |   _ \ / \ 
( ( |\___| |'_|_|_|'_._V_-'|\_____\ 
\ \ ___) |__)| |_| |||| || (__| )))) ) 
'| __|_|_.|_|_|_|_|_\\_,|///|/|/|/|/| 
=====|_|=====|_|_/=///_//_/ 

:: Spring Boot ::                (v2.6.3)
```

```
2022-01-26 00:23:49.176 INFO 15072 --- [main] c.e.d.CicdDeploymenntApplicationTests : Starting CicdDeploymenntApplicationTests using Java 1.8.0_271 on LAPTOP-H1JD0KS8 with PID 15072 (started by LAPTOP-H1JD0KS8$ in C:\ProgramData\Jenkins\.jenkins\workspace\cicddployment)
2022-01-26 00:23:49.180 INFO 15072 --- [main] c.e.d.CicdDeploymenntApplicationTests : No active profile set, falling back to default profiles: default
2022-01-26 00:24:00.389 INFO 15072 --- [main] c.e.d.CicdDeploymenntApplicationTests : Started CicdDeploymenntApplicationTests in 13.044 seconds (JVM running for 17.785)

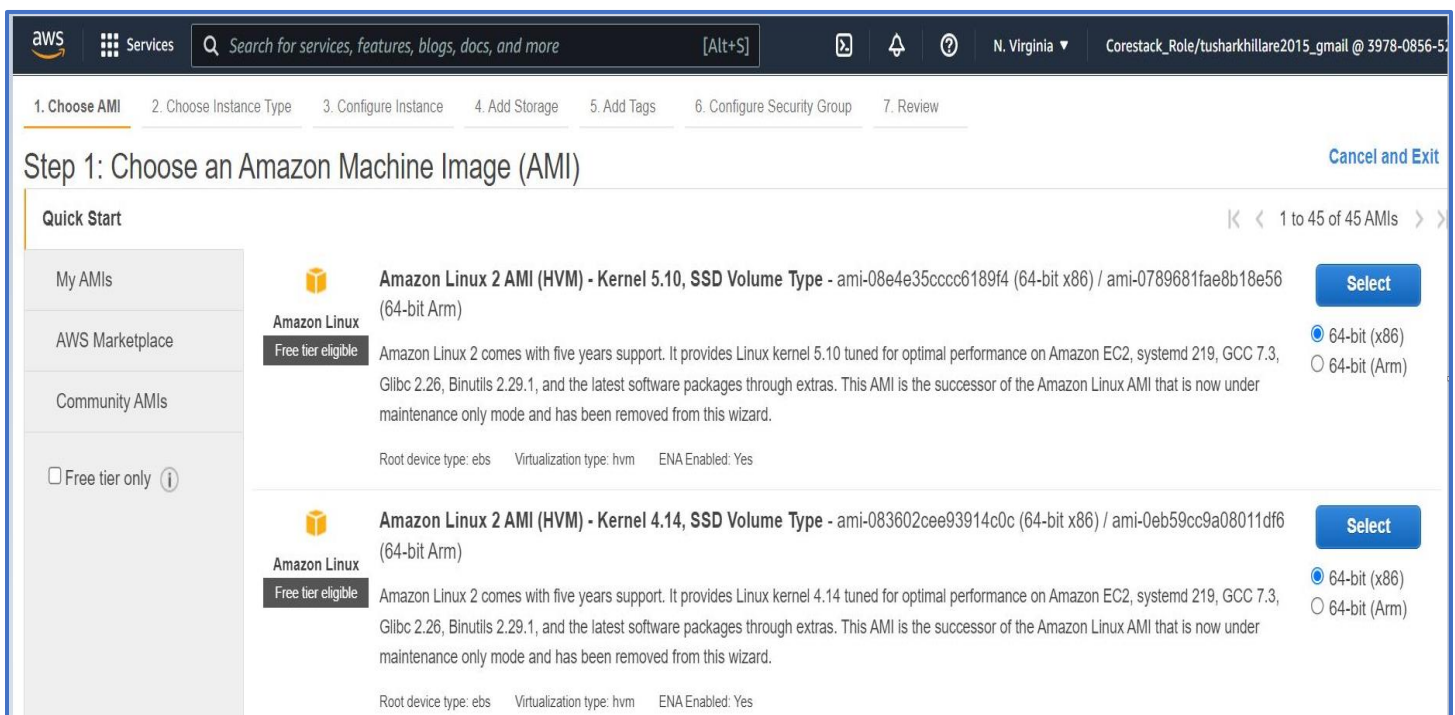
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 17.913 s - in com.example.demo.CicdDeploymenntApplicationTests
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO]
[INFO] --- maven-war-plugin:3.3.2:war (default-war) @ CICDDeployenmt ---
[INFO] Packaging webapp
[INFO] Assembling webapp [CICDDeployenmt] in [C:\ProgramData\Jenkins\.jenkins\workspace\cicdddeployment\target\CICDDeployenmt-0.0.1-SNAPSHOT]
[INFO] Processing war project
[INFO] Building war: C:\ProgramData\Jenkins\.jenkins\workspace\cicdddeployment\target\CICDDeployenmt-0.0.1-SNAPSHOT.war
[INFO]
[INFO] --- spring-boot-maven-plugin:2.6.3:repackage (repackage) @ CICDDeployenmt ---
[INFO] Replacing main artifact with repackaged archive
[INFO]
[INFO] --- maven-install-plugin:2.5.2:install (default-install) @ CICDDeployenmt ---
[INFO] Installing C:\ProgramData\Jenkins\.jenkins\workspace\cicdddeployment\target\CICDDeployenmt-0.0.1-SNAPSHOT.war to
C:\Windows\system32\Config\SystemProfile\m2\Repository\com\example\CICDDeploymennt\0.0.1-SNAPSHOT\CICDDeployenmt-0.0.1-SNAPSHOT.war
[INFO] Installing C:\ProgramData\Jenkins\.jenkins\workspace\cicdddeployment\pom.xml to
C:\Windows\system32\Config\SystemProfile\m2\Repository\com\example\CICDDeployenmt\0.0.1-SNAPSHOT\CICDDeployenmt-0.0.1-SNAPSHOT.pom
[INFO] -----
[INFO] BUILD SUCCESS
```

- Once the Build is completed login to Docker Hub (Desktop/URL) and see if the image has been created and pushed to the Docker hub as in the below image:



C. AWS EC2 instance and Deploy

- Login to Amazon AWS and choose to launch a new instance by selecting the Amazon Machine Image (AMI)



- After selecting the AMI do the configuration as in the below images:

Services

[Alt+S]

N. Virginia

Corestack_Role/tusharkhillare2015_gmail @ 3978-0856-5269

1. Choose AMI
2. Choose Instance Type
3. Configure Instance
4. Add Storage
5. Add Tags
6. Configure Security Group
7. Review

Step 2: Choose an Instance Type

Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. [Learn more](#) about instance types and how they can meet your computing needs.

Filter by: All instance families Current generation [Show/Hide Columns](#)

Currently selected: t2.micro (- ECUs, 1 vCPUs, 2.5 GHz, -, 1 GiB memory, EBS only)

| | Family | Type | vCPUs | Memory (GiB) | Instance Storage (GB) | EBS-Optimized Available | Network Performance | IPv6 Support |
|-------------------------------------|--------|---|-------|--------------|-----------------------|-------------------------|---------------------|--------------|
| <input type="checkbox"/> | t2 | t2.nano | 1 | 0.5 | EBS only | - | Low to Moderate | Yes |
| <input checked="" type="checkbox"/> | t2 | t2.micro Free tier eligible | 1 | 1 | EBS only | - | Low to Moderate | Yes |
| <input type="checkbox"/> | t2 | t2.small | 1 | 2 | EBS only | - | Low to Moderate | Yes |

1. Choose AMI
2. Choose Instance Type
3. Configure Instance
4. Add Storage
5. Add Tags
6. Configure Security Group
7. Review

Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage of the lower pricing, assign an access management role to the instance, and more.

Number of instances
[Launch into Auto Scaling Group](#)

Purchasing option
☐ Request Spot instances

Network
[Create new VPC](#)

Subnet
[Create new subnet](#)

Auto-assign Public IP

Hostname type

DNS Hostname
☒ Enable IP name IPv4 (A record) DNS requests
☒ Enable resource-based IPv4 (A record) DNS requests
☐ Enable resource-based IPv6 (AAAA record) DNS requests

Placement group
☐ Add instance to placement group

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

Step 4: Add Storage

Your instance will be launched with the following storage device settings. You can attach additional EBS volumes and instance store volumes to your instance, or edit the settings of the root volume. You can also attach additional EBS volumes after launching an instance, but not instance store volumes. [Learn more](#) about storage options in Amazon EC2.

| Volume Type <small>i</small> | Device <small>i</small> | Snapshot <small>i</small> | Size (GiB) <small>i</small> | Volume Type <small>i</small> | IOPS <small>i</small> | Throughput (MB/s) <small>i</small> | Delete on Termination <small>i</small> | Encryption <small>i</small> |
|---------------------------------|-------------------------|---------------------------|-----------------------------|------------------------------|-----------------------|------------------------------------|--|------------------------------|
| Root | /dev/xvda | snap-06705e15d64acb59a | 30 | General Purpose SSD (gp2) | 100 / 3000 | N/A | <input checked="" type="checkbox"/> | Not Encrypt <small>v</small> |
| <button>Add New Volume</button> | | | | | | | | |

Ensure that you have configure Security group so that we are able to do the inbounds and outbounds rules to access our spring boot application using the public IP address as in the below image:

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

Step 6: Configure Security Group

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. [Learn more](#) about Amazon EC2 security groups.

Assign a security group: ☒ Create a new security group
☐ Select an existing security group

Security group name:
Description:

| Type <small>i</small> | Protocol <small>i</small> | Port Range <small>i</small> | Source <small>i</small> | Description <small>i</small> |
|------------------------------|---------------------------|-----------------------------|---|---|
| SSH <small>v</small> | TCP | 22 | Custom <small>v</small> 0.0.0.0/0 | e.g. SSH for Admin Desktop <small>x</small> |
| All traffic <small>v</small> | All | 0 - 65535 | Anywhere <small>v</small> 0.0.0.0/0, ::/0 | e.g. SSH for Admin Desktop <small>x</small> |
| <button>Add Rule</button> | | | | |

Once done with the configuration review and launch the instance

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

Step 7: Review Instance Launch

Please review your instance launch details. You can go back to edit changes for each section. Click **Launch** to assign a key pair to your instance and complete the launch process.

⚠ Improve your instances' security. Your security group, launch-wizard-1, is open to the world.
Your instances may be accessible from any IP address. We recommend that you update your security group rules to allow access from known IP addresses only. You can also open additional ports in your security group to facilitate access to the application or service you're running, e.g., HTTP (80) for web servers. [Edit security groups](#)

AMI Details

[Edit AMI](#)

Free tier eligible **Amazon Linux 2 AMI (HVM) - Kernel 4.14, SSD Volume Type - ami-083602cee93914c0c**
Amazon Linux 2 comes with five years support. It provides Linux kernel 4.14 tuned for optimal performance on Amazon EC2, systemd 219, GCC 7.3, Glibc 2.26, Binutils 2.29.1, and the latest software packages through extras. This AMI is the successor of the Amazon Linux AMI that is n...
Root Device Type: ebs Virtualization type: hvm

Instance Type

[Edit instance type](#)

| Instance Type | ECUs | vCPUs | Memory (GiB) | Instance Storage (GB) | EBS-Optimized Available | Network Performance |
|---------------|------|-------|--------------|-----------------------|-------------------------|---------------------|
| t2.micro | - | 1 | 1 | EBS only | - | Low to Moderate |

Instances (1/1) [Info](#)

Connect

Instance state ▼

Actions ▼

Launch instances



< 1 >

| <input checked="" type="checkbox"/> | Name ▼ | Instance ID | Instance state ▼ | Instance type ▼ | Status check | Alarm status | Availability Zone |
|-------------------------------------|--------|---------------------|------------------|-----------------|--------------|--------------|-------------------|
| <input checked="" type="checkbox"/> | - | i-0543bccd23b851484 | Running | t2.micro | Initializing | No alarms + | us-east-1c |

Instance: i-0543bccd23b851484 ×

Details

Security

Networking

Storage

Status checks

Monitoring

Tags

▼ Instance summary [Info](#)

Instance ID

i-0543bccd23b851484

IPv6 address

-

 Public IPv4 address
copied 3.91.182.20 | [open address](#)

Instance state

Running

Private IPv4 addresses

172.31.84.149

Public IPv4 DNS

 ec2-3-91-182-20.compute-1.amazonaws.com |
[open address](#) Connect to instance [Info](#)

Connect to your instance i-0543bccd23b851484 using any of these options

EC2 Instance Connect

Session Manager

SSH client

EC2 Serial Console

Instance ID

i-0543bccd23b851484

Public IP address

3.91.182.20

User name

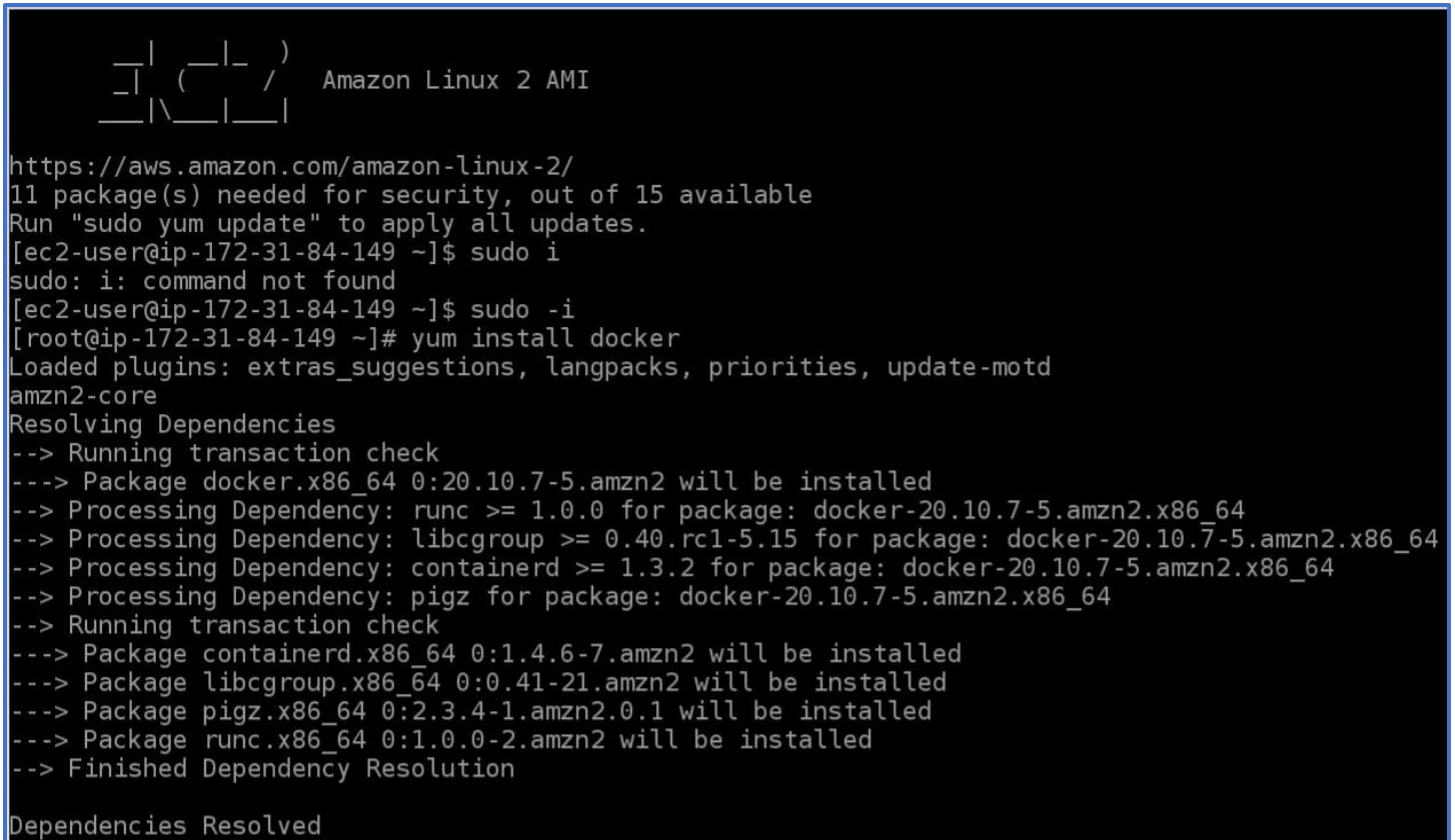
Connect using a custom user name, or use the default user name ec2-user for the AMI used to launch the instance.

**Note:** In most cases, the guessed user name is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI user name.

Cancel

Connect

- It will Open the AWS instance as in the below image:



```

_ | _ | )
_ | ( _ | /   Amazon Linux 2 AMI
_ | \ _ | _ |

https://aws.amazon.com/amazon-linux-2/
11 package(s) needed for security, out of 15 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-84-149 ~]$ sudo i
sudo: i: command not found
[ec2-user@ip-172-31-84-149 ~]$ sudo -i
[root@ip-172-31-84-149 ~]# yum install docker
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core
Resolving Dependencies
--> Running transaction check
--> Package docker.x86_64 0:20.10.7-5.amzn2 will be installed
--> Processing Dependency: runc >= 1.0.0 for package: docker-20.10.7-5.amzn2.x86_64
--> Processing Dependency: libcgroupp >= 0.40.rc1-5.15 for package: docker-20.10.7-5.amzn2.x86_64
--> Processing Dependency: containerd >= 1.3.2 for package: docker-20.10.7-5.amzn2.x86_64
--> Processing Dependency: pigz for package: docker-20.10.7-5.amzn2.x86_64
--> Running transaction check
--> Package containerd.x86_64 0:1.4.6-7.amzn2 will be installed
--> Package libcgroupp.x86_64 0:0.41-21.amzn2 will be installed
--> Package pigz.x86_64 0:2.3.4-1.amzn2.0.1 will be installed
--> Package runc.x86_64 0:1.0.0-2.amzn2 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

```

D. Project Appearance

Here we have gone through the flow by the following steps:

- Programming Spring boot application with a docker file.
- Pushing the project into the GitHub.
- Configure Jenkins to have GitHub project as input and Docker Hub as output and by building the WAR file and converting it into the Docker image and pushing into the Docker Hub.
- Configure the AWS ECS and connecting it through Putty.
- Deploying and hosting the docker file on the AWS EC2 instance using the demon command.
- We can access our Spring boot application from anywhere using the Public Ip address in my case which is: <http://3.91.182.20:7000/>
- Appearance of the website is in the below images:



Not secure

3.91.182.20:7000

Public IP Address



Apps



git hub - megnand



GitHub - Niks4u2/KitchenStory



Firebase



Normal English vs Advanced E...

CI-CD Deployment using Jenkins+Docker into AWS EC2.Developed by Tushar Khillare

Thank You.....