Phase 5 :

Day 1 : 05-11-2022

Testing, Devops and deployments

Testing using TestNG Framework

Selenium Automation testing tool

Docker

CI and CD tool with Jenkin

Overview of Kubernetes

Cloud computing using AWS : S3, EBS and EC2 instance

In phase 3 : jUnit testing (jUnit 5)

Testing : Testing is use to find the defect or error or bugs in the application.

Running the application using any technologies means indirectly we are testing the application.

Read a, b

Compute sum = b+b

Write b

Testing the function functionality without main method is known as testing.

TestNG TestNG is an open source testing framework inspired from Junit and NUnit. In TestNG Ng means next generation. It is light weighted framework.

Compared to Junit testing framework

1. They provided more meaning full annotation in TestNG
2. It support parallel testing.
3. It provided default testing report in the form of HTML.
4. It support group of test.
5. It support priority base test.

Junit testing mainly use for unit testing.

TestNG framework mainly use unit testing as well as integration testing with one of the automation tools ie Selenium.

Test case : It is type of class which contains more than one test function which help to test function functionality. Inside that class we can write more than one method with @Test Annotation.

Test suite : test suite is use to execute more than one test case class. In Junit we need to make normal java class with annotation to run the test suite. In TestNG provide test suite using xml file.

Assertion : inside test we can use more than one assertion method which help to check actual and expected output.

TestNG hook methods or Life cycle methods

Day 2 : 06-11-2022

TestNG framework execute more than one test function by default ascending order of function name.

If we want to execute base upon our requirement then we have to use priority attribute.

TestSuite is use to execute more than one test case. Every test case class contains more than one test function.

Mock testing

We can do mock testing using jUnit as well as TestNG framework.

Controller or Rest Controller

Http Layer testing using Spring boot.

Html, css, js

Angular : Jasmine and Karma.

React JS : JEST

Jasmine, Mocha with Chai, JEST these framework help to do the unit testing for UI component.

Java : jUnit and TestNG

Selenium : Selenium is an open source Web UI Automation testing tool. Which is use to do the Testing for UI component develop in any language. Selenium is use to integration testing.

Selenium support other programming language like Java, Python, C#, Perl etc.

Selenium support to integrate with any testing framework like junit or TestNG.

Selenium provided web driver which help to load the web page base on browser.

And it provided lot of API which help to read the dom elements. But Selenium doesn’t provide assert methods so we need to take the help of TestNG or junit which provided set of assert method which help check actual and expected output.

Type 4 driver for MySQL database

Class.forName(“com.mysql.cj.jdbc.Driver”);

This class is a part of mysql connector jar file.

Day 3 : 12-11-2022

We can do the Selenium Testing using selenium IDE.

Docker :

Docker is an advanced version of OS virtualization software platform makes it easier to create, deploy and run the application in Docker container environment.

VM ware software.

Virtualization : it is use to create virtual version of a resource such as server, data storage, or application or tool.

Using VM ware software we can run Guest OS

Base machine 16 GM RAM

HD : 1 TB

GOS 🡪 4GM RAM , 50 HD

If we are planning to run more than one GOS

10 GOS

Using Docker we can create Containerization application.

Virtualization Vs Containerization

Virtualization is use to create abstract version of physical machine or OS.

Containerization is use to create abstract version of the application.

Docker container is responsible to run this application.

Dockerfile :Docker file is a blue print or it contains set of instruction which help to create or build the image. This file contains application details which we want to run with that application configuration details. With help of Docker file we build or create the image.

Ex: Normal Java program.

Docker Image : Docker image are the source code for our container.

Ex: Docker image is like a Jar or war file in Java.

Docker container : Docker container is responsible to run the docker image. Docker container also known as instance of image. Once image run actual application become up.

Container : it is a run time environment or engine which help to run the application.

Web Container

EJB Container

**Web Server** : Tomcat : it is light weighted sever which contains only one type of container ie web container. This container is responsible to run the servlet and jsp programs.

**Application Server** : WebLogic, JBoss, WebSphere, Glashfish etc.

This server contains different types of container ie Web Container, EJB Container, JMS container

Spring container

Angular Container

docker --version : This command is use to find the version of docker

docker images : this command is use to find all images present in local machine.

docker pull imageName : This command is use to pull the image

docker pull hello-world

docker run imageName/imageId : this command is use to run the image

Docker hub : it is remote repository which help to publish or push and pull the images. Docker hub is like a git hub but git hub can store any type of files. But Docker hub hold more than one images.

Please create Docker hub account

busybox is a small tiny OS between size 1 to 5mb

docker pull busybox

docker run -it ubuntu bash

docker pull alpine

**docker run -it alpine**

We want to create our own custom image to display echo message.

Dockerfile

FROM busybox

CMD ["echo","Welcome to Docker Custom image created by Akash"]

docker build –t my-busybox1 . –f Dockerfile

docker run my-busybox1

We will create another image to display date information.

FROM alpine:latest

CMD ["date"]

docker build -t myalpine1 . -f Dockerfile

docker run myalpine1

Creating image to run the Java Program

class Demo {

    public static void main(String[] args) {

        System.out.println("Welcome to Java program running through docker");

    }

}

**Dockerfile**

FROM openjdk:8

COPY Demo.java .

RUN javac Demo.java

CMD ["java","Demo"]

docker build -t myjava1 . -f Dockerfile

docker run myjava1

Day 4 : 13-11-2022

Creating image to run the spring boot application

mvn clean : to clean the project

mvn compile : to compile the project

mvn test : to test the project

mvn package : it is use to create jar or war file

first create the spring boot project with one or more rest api

then create the executable jar file using mvn or through eclipse IDE

then create the docker image with help of docker file

**FROM** openjdk:11

**COPY** ./target/spring-with-docker.jar .

**CMD** ["java","-jar","spring-with-docker.jar"]

docker build –t myspring1 . –f Dockerfile

docker run –p 9090:9090 myspring1

docker run -d -p 9090:9090 my-spring1

docker ps : this command is use to display all running container

or

docker container ls : this command is use to display all running container

docker stop containerId/containerName

docker start containerId/containerName

docker rm containerId/containerName

docker rmi imageName/imageName

creating image to run angular application

ng new angular-docker

routing 🡪 no

styling 🡪 css

app.component.html

<h2>Welcome to angular project created by Akash Kale</h2>

After created the project now we need to build the project

ng build

Now create the Dockerfile

Dockerfile

FROM nginx

COPY ./dist/angular-docker/ /usr/share/nginx/html

docker build –t my-angular1 . –f Dockerfile

By default nginx server running on port number 80.

**docker login** : this command is use to connect local terminal with docker hub account

docker tag imageName dockerHubAccountId/imageName:version

docker tag imageName dockerHubAccountId/imageName:1.0

docker tag my-angular1 akashkale/my-angular1:1.0

after tag created then we can push the image to docker hub

docker push akashkale/my-angular1:1.0

more than one image container want to communicate with each others.

Agular Image want to communicate with Spring boot image

Front end -----------------------🡪 Backend technologies

Spring boot image want to communicate with mysql image container.

Frontend

Browser

Backend

Backend : spring boot

Db 🡪 MySQL

Docker – Compose

Docker swarm

Kubernetes

These above technologies which help to maintain the more than one containers.

Docker compose is tool which help to run more than one container at time it may be interact with each other or working independently. In Docker compose we will use docker-compose command which help to run the docker-compose.yml ( which contains all images details which we want to run).

With help of docker-compose command we can run more than one container at time like build, up, down etc.

**Day 5 : 19-11-2022**

Creating docker compose file responsible to run spring boot and mysql containers.

mvn compile

mvn test

mvn package : this command is use to create jar or war file

so please remove testing dependencies from pom.xml file

then create the jar file using mvn package command.

This command is use to pull and run the image

**docker run --name some-mysql -e MYSQL\_ROOT\_PASSWORD=my-secret-pw -d mysql:8**

this command is use to check container status

**docker ps**

This command is use to open mysql image os terminal

**docker exec -it some-mysql bash**

to connect mysql in image

**mysql –u root –p**

**🡪my-secret-pw**

Network environment between two container to communicate to each others.

Spring boot image pull mysql image we can run mysql image

These two container are running in different OS.

Using Docker compose we can configure more than one image details which help to communicate to each others.

**Steps to run docker compose**

First open the project in eclipse IDE (spring-with-mysq-in-docker)

Update maven project

Then create the jar file using **mvn package** command

Then create docker image using command as

**docker build –d employee-spring-boot –d . Dockerfile**

docker-compose up –d

then open another command prompt and check both container are running or not using **docker ps**

then git both rest api one for store ie post and another for get

<http://localhost:9999/storeEmployee>

<http://localhost:9999/retrieveEmployee>

then if you want to store both container run the command as docker-compose down

CI and CD : Continues Integration and Continues Delivery or Continues Deployment

Team1 push

Team2 push GitHub (build the project) CI / CID

Team3 push

Testing server production

Server

Whenever any team push the code to remote repository then CI and CD tool pull the code from Github and build it if it build successfully then it will pass this code to anther team.

Try to avoid System.out.println(“”);

log.info

log.error

log.warning

Jenkin : Jenkin is a type of CI and CD tools created using Java language. Jenkin is open source plugin base CI and CD tools. It is also known integration server which help to configure with sub server tool like git and as well as we can add plugin for devops tools like maven, ant, gradle, docker etc.

Jenkin GUI base CI and CD tools.

**Jenkin Architecture**



**We need install the Jenkin**

1. **Install Exe file (install the software base upon OS).**
2. **We need to download war file and run that war file using tomcat server.**
3. **We can take the help of Docker Jenkin image**

**Jenkin job stored in base machine.**

**This below command is use to pull Jenkin Server and run server on default port number 8080.**

**docker run -p 8080:8080 -p 50000:50000 --restart=on-failure jenkins/jenkins:lts-jdk11**

**When we run this command. It will display random password in terminal. Please copy that password**

**After Jenkin server started successfully.**

**Open the browser and type** [**http://localhost:8080**](http://localhost:8080)

**Day 6 : 20-11-2022**

**firstJob : it is use to display echo message.**

**SecondJob : it execute this job again and again using trigger.**

**ThirdJob : we pull the Java project from Git.**

**Fourth Job : we pull Java project from git and run the program.**

**Using**

**docker ps (checking running container).**

**Docker stop containerId**

**Docker rm containerId**

[**https://www.jenkins.io/download/**](https://www.jenkins.io/download/)

**download the war and paste in tomcat webapp folder.**

**make sure local machine jdk must be 11+ version**

**then open the command prompt in tomcat webapp folder.**

**C:\Program Files\apache-tomcat-9.0.63\webapps**

**java –jar jenkins.war**

**or**

**java –jar Jenkins.war –httpPort 8181**

**after started**

**then open** [**http://localhost:8080**](http://localhost:8080)

**then copy and paste random password**

**then install the suggested plugin.**

**We will create job to run java program, maven program and**

**Jenkin pipeline.**

**Creating job to run the Java program**

[**https://github.com/Kaleakash/jenkinjava.git**](https://github.com/Kaleakash/jenkinjava.git)

**creating a job to execute spring boot project in Jenkin Environment**

[**https://github.com/Kaleakash/jenking\_with\_spring\_boot.git**](https://github.com/Kaleakash/jenking_with_spring_boot.git)

**mvn clean**

**mvn compile**

**mvn test**

**mvn package**

**mvn install**

**Jenkin Pipe line**

**Jenkin Pipe line is collection of more than one event or jobs which are interlinked with one another in a sequence manner.**

**Check all version of the software**

**Compile the project**

**Test the project**

**Package the project mean create jar or war file.**

**Jenkin pipe line syntax written using Groovy DSL (Domain Specific Language).**

**Jenkin pipe line we can execute two ways**

1. **Declarative manner**

**We need to create the file with name Jenkinsfile and in that file we need to write script code to run more than one job interconnected to each others. File must be store in sub version.**

1. **Scripted pipe line : in this ways we use sample file must be present in Jenkin environment.**

**Jenkinsfile**

|  |
| --- |
| pipeline { |
|  | agent any |
|  | tools {nodejs "MyNode"} |
|  | stages { |
|  | stage("Check Node Version"){ |
|  | steps { |
|  | sh "node --version" |
|  | } |
|  | } |
|  | stage("install dependencies"){ |
|  | steps { |
|  | sh "npm --version" |
|  | sh "npm install" |
|  | } |
|  | } |
|  | stage("Test"){ |
|  | steps { |
|  | sh "node app.js" |
|  | } |
|  | } |
|  | stage("Release the version"){ |
|  | steps { |
|  | echo "Release the Version" |
|  | } |
|  | } |
|  | } |
|  | } |

Cloud Computing

The team cloud refer to a network or the internet. It is technology that users remote server on the internet or server machine to store data, manage and access data online rather than storing in local device.

There are certain service and model working in cloud computing

2

Deployment model : deployment model defines the types of access to the cloud.

Public cloud : Amazon EC2 , google cloud , sun cloud, Azure

Private cloud : Amazon virtual private cloud

Hybrid cloud : combination of private and public

Community cloud : lot of organization support to develop the cloud.

Service model

IaaS : Infrastructure as a Service : software and hardware

PaaS : Platform as a Service : In OS we need open source server or tool to deploy the application

Tomcat, weblogic, WAS etc. IDE, Database

SaaS : Software as a Service : Salesforce etc, Google App, Cisco Webex, GoToMeeting

AWS (Amazon Web Service)

Azure

Google cloud

Oracle cloud

IBM Cloud

AWS : S3 and EC2

AWS as well Azure

In AWS we can create two types of account

1. Root user
2. IAM user (Identity Access Management) user.

AWS S3 (Simple Storage Service) : it is like a google drive which help to share the any type of file data within a AWS account as well as outside base type security apply.

EC2 instance : AWS EC2 : Elastic Compute cloud : Using AWS EC2 instance which help to create virtual server machine while creating that machine we can configure according to our requirement

Like OS : Linux, Unix, Window, Mac etc

CPU details configure

RAM

Hard disk for virtual sever

Number instance at same time we can run

For that machine we can connect through using SSH client from our machine and base upon our requirement we will install all software like Java, Git, Maven, Jenkin, Docker and deploy the application In EC2 instance.

We need to install node JS

We install docker and we create image push this image in Docker hub and pull in EC2 instance we can run.

We can share the data to ES2 instance using git or S3.

We create war file for servlet and jsp

We create jar file for spring boot and upload in S3 and in EC2 instance we will download that jar or war and we run the application in EC2 instance so EC2 instance will provide one IPAddress so any one can access our application.

In ES2 instance if we want to download any file like text file, jar file , pdf file from S3.

In EC2 terminal we need to write the command as

wget URL of object part of bucket

Create Spring boot project in local machine with one or more rest api.

Then push this project in git hub

In Maven project create

git init

git add .

git commit –m “done”

connect to remote repository

git push –u origin HEAD

In In AWS create ES2 instance

Then using GIT bash connect to ES2 instance

Then install Java software (11)

Then install git software

Then install maven software.

In EC2 instance using git please clone the from Github

Using Docker we already publish angular project in Docker hub account.

Phase 5 end project

First we need to create spring boot project

With one or more rest api must created

This project we need to push in git hub account.

We need to open EC2 instance

Then connect ES2 instance using SSH client

Install required software

Git

Java (OpenJDK 11)

Maven

Docker

Jenkin

Then open the port for Jenkin 8080

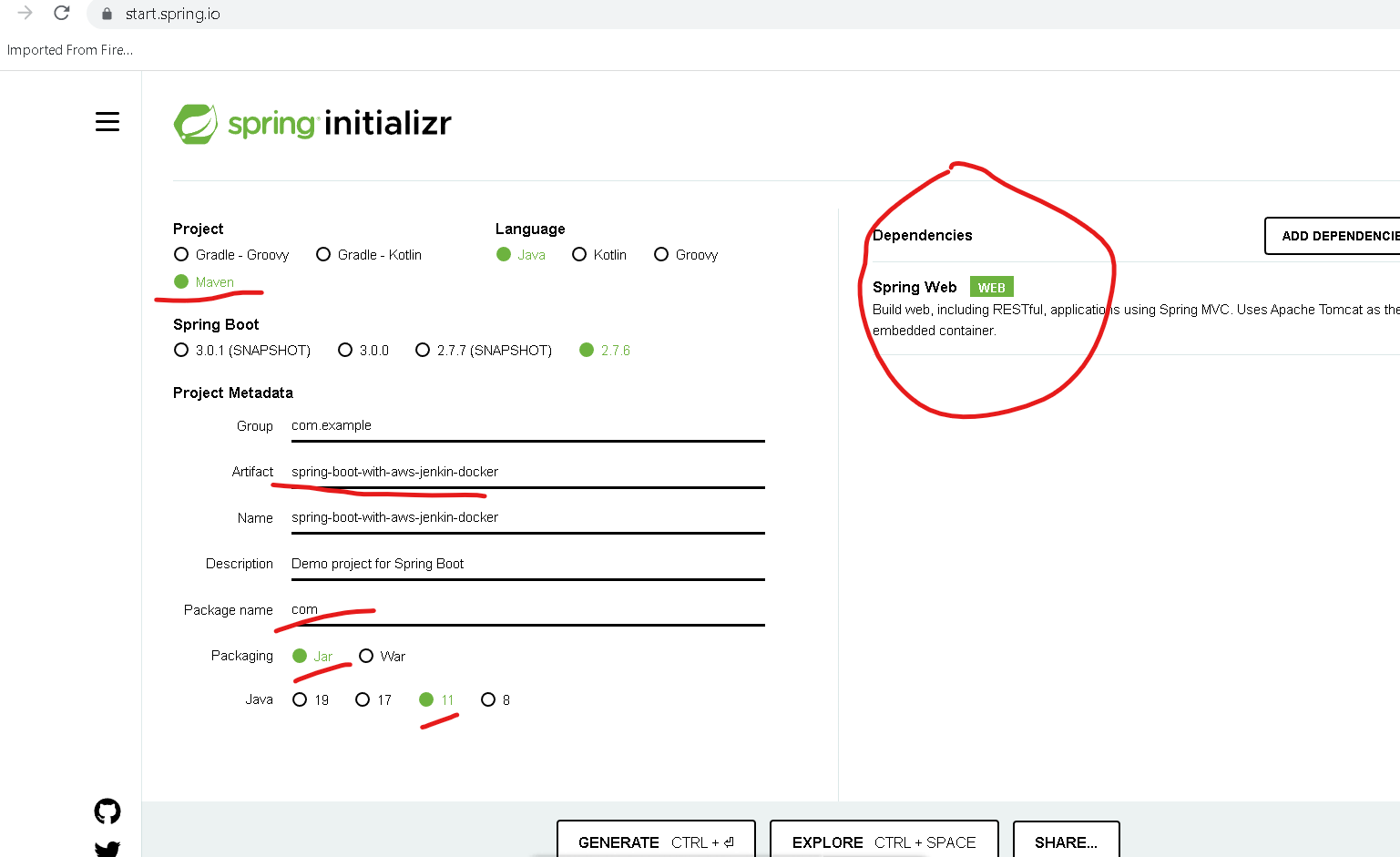
<http://198.12.436.19:8080>

In Jenkin we need to create Pipeline JOB that job is responsible to pull the project from git hub account then using Docker we will deploy this application in ES2 instance so we can we see that application.

**Steps to create the application do deploy using Jenkin Pipe with in AWS with Docker**

First open the eclipse IDE and create spring boot project with spring initializer.

<https://start.spring.io/>



In Spring boot please create one or more rest api

In application.properties file port number

In pom.xml file write finalName tag with custom name for jar file

Then create the jar file usng mvn package

Then create Dockerfile

Then create Jenkinsfile

Then push this project o github account.

Now please login to AWS account

And start EC2 instance

In ES2 instance (please refer EC2 instance plugin notepad file

1. Please update ec instance using command
2. Then install Open JDK 11
3. Then install git
4. Then install maven
5. Then install docker
6. Then install Jenkin