31-01-2022

Phase 4 :

Day 1

**Testing and Deploying an application**

**Testing**

**Testing with JavaScript that may be client side JavaScript and Server Side JavaScript.**

**Testing with Angular Application**

**GraphQL : self learning**

**Grunt : JavaScript test runner**

**Docker**

**CI and CD tools with Docker**

**Introduction to Cloud computing with AWS**

**S3, EC2 and EBS Volume**

Testing : finding defects or error or bugs in application is knows as testing.

Testing : are divided into two types

Black box : Input ---------------🡪 Process ----------------🡪 Output

Behaviour

White box : Input ---------------🡪 Process --------------------🡪Output

Structural

Functional testing : checking actual functionality of the application. Online shopping

1. View all product details
2. Add the product to cart
3. Place the order

Non functional testing

: it use to support to run the application.

Environment to run application or tool to run application or software to run the application.

01-02-2022

Day 2

Unit testing : It is a type of white box testing where developer is use to check the smallest testable module which written inside may be function or method or module or class. Which help to do achieve specific task.

To do the Unit testing in UI Technologies. We can use different type of framework.

Jasmine : Jasmine is open source DOM – Less simple Java Script testing framework which help to do the Testing for Client side as well as Server side JavaScript code.

Jasmine also known as BDD testing framework. Behaviour Driven Development Testing framework.

Create the Folder as Testing

Then create the sub folder as

**Client Side scripting testing**

Testing without Node JS

Jasmine provide lot of pre-defined function which help to do the testing.

Suite : Suite is use to write more than one spec as well as another suite.

To write the suite we have to pre-defined function it

describe(“Messsage”, ()=> {

});

Spec : spec is a type of function which help to do the testing for function functionality. To write the spec jasmine provided pre-defined function ie

It(“Message”,()=> {

})

expect : Jasmine provide lot of pre-defined expect function which help to check actual and expected output.

expect(result).toXXX(actualOutput);

Karma : Karma is known as Test runner which help to do the testing for JavaScript file using Jasmine framework.

Jasmine Old Version depends upon the Karma runner to do the testing.

But new version of jasmine provided own test runner to do the testing.

If we want to do the Angular testing. Angular Framework internally use Jasmine framework which provide pre-defined function which help to testing for angular application. Angular use Karma as test runner to do the testing.

Jasmine look or life cycle functions.

beforeEach() : this function call automatically before each it functions.

beforeAll() : this function call automatically only once before all it function.

afterEach() : this function call automatically after each it function.

afterAll() : this function call automatically only once after all it functions

if Employee object depends upon the project object. But project object task under process or need more than to complete.

Employee object create the fake or mock object of project.

Jasmine provided SpyOn function to do mock testing.

Mocha

JEST

Day 3

02-02-2022

Testing for Angular Application

Angular Framework internally use Jasmine Open source framework which provide set of function

describe, it and more than expect which help to do the testing.

Angular framework using Karma as a test runner to the run the jasmine testing with angular.

Angular provide own testing utilities API which is a part of @angular/core/testing module. Which contains TestBed is a class which provide set of method or function which help to do the testing for angular component, service, pipe and more.

Create one folder ie angular-testing-app

ng new angular-testing-app

package.json file : we can see devDependencies ie jasmine and karma.

karma-config.json file

To run the angular test we have to use the command as

ng test

03-02-2022

when we what to do the Testing for Rest API call using Angular we have make sure all REST API develop any language like Java Or python or Express JS must be ready.

Fake or mock rest API to test.

Create Server Side Scripting with Node JS

Create the package.json file using command as

npm init

testing dependencies must only in development mode not in production mode.

npm install jasmine -D

npm install jasmine –g

npm install jasmine-node –D

jasmine init : This command is use to create the spec folder which contains jasmine.json file.

in node js testing using jasmine

we have to use the command as

jasmine

connect the database using

mongo db

mongoose

node js use another library to check rest api call.

Supertest : Supertest is a external library which help to test express or rest api.

npm install supertest –D

npm install cors

npm install mongoose

npm install express

Day 5

07-02-2022

Docker :

Docker is an advanced OS Virtualization software platform that makes it easier to create, deploy and run the application in Docker container.

Virtualization : Virtualization is the means of employing software or application (such as Hypervisor) to create a virtual version of resources such as a server, data storage, or application.

Virtual machine : A VM (Virtual Machine) is a computing environment or software that aids developers to access an operating system via physical machine.

Base Machine ; Window or Mac or Unix

VM Ware software

Oracle VM

Citrix

Etc

Base Machine : Window 10 OS and 16 GB RAM

Run OS images like Cent OS or Unix OS etc

Cent OS : 4GB RAM

Unix OS : 4 GB RAM

10 to 20 Guest OS.

Docker mainly use to achieve containerization.

Virtualization Vs containerization

Virtualization is an abstract version of physical machine. While containerization is the abstract version of an application or software.

Container : Container is also known as engine or run time environment for running the application or software.

Docker Container : Running the instances of Docker Images container turn or run the actual application. A container includes an application and all its dependencies.

Docker Images : It is a file system and configuration of our application which are used to create the container. Using image we can run the container.

Docker file : A Docker file is a blueprint or set of instruction that defines how our images is built.

Docker hub : Docker hub is like a github which help to publish and store the Docker images. Using Docker hub we share the images between two team or groups so we can push and pull the image from the Docker hub.

**Docker commands**

**docker –version : This command is use to check the docker version**

**docker images : own machine**

**LMS labs**

**sudo docker images**

**syntax to pull the images in local machine**

**docker pull imageName**

**docker pull hello-world**

**command to run the image**

**docker run image-name/image-id**

**docker run hello-world : This image is use to run the C program**

**docker pull busybox**

**docker run –it busybox :This command is use to run the os in iterative mode.**

**Create folder Docker images**

**Then create sub folder as**

**Image to run date command**

**To create image we have to create the Dockerfile**

**Dockerfile**

**docker build –t imageName . –f Dockerfile**

**Example**

**docker build –t mydate-mern . –f Dockerfile**

**To create the image to run node js program**

**app.js**

let a = 10;

let b = 20;

let sum = a+b;

console.log("Sum of two number is "+sum);

function sayHello(name){

    return "Welcome "+name;

}

console.log(sayHello("Ravi"));

**Dockerfile**

FROM node:latest

COPY app.js .

CMD ["node","app.js"]

**docker build -t mynode-mern . -f Dockerfile**

**Creating Docker image to run Express JS program**

**Create the package.json file using npm init command**

**npm init**

**npm install express**

**app.js**

let express = require("express");

let app = express();

app.get("/",(req,res)=> {

    res.send("Welcome to Simple Express JS with Docker")

});

app.get("/user/:name",(req,res)=> {

    let name = req.params.name;

    res.send("Welcome user "+name+" To express with docker");

})

app.listen(9090,()=>console.log("Server running on port number 9090"));

**Dockerfile**

FROM node:latest

RUN mkdir /usr/app/

WORKDIR /usr/app/

COPY package.json /usr/app/

RUN npm install

COPY app.js /usr/app/

CMD [ "node" ,"app.js"]

**docker build -t myexpress-mern . -f Dockerfile**

**docker run myexpress-mern : We can’t see any output**

**This image contains container so we have to expose port number.**

**docker run –p 9090:9090 imageName/imageid**

**9090 : the port number in red colour in right side must be actual port number.**

**9090 : The port number in green colour in left side expose port number it can be same actual port number or different port number.**

**docker run –d –p 9090:9090 imageName/imageid**

**This command is use to run the image container in background**

**To check all images running in our machine**

**docker images**

**To check all running container we can use the command as**

**docker ps**

**To stop the container we can use the command as**

**docker stop containerid**

**To start the container we can use the commands as**

**docker start containerid**

**docker ps : This command display only running container**

**docker ps –a : This command display all container which present in our machine it may be running or stopped.**

**Creating the image to run the html, css and js program**

**After created web application using html, css and js we have to deploy on server.**

**Tomcat**

**Web logic**

**Jboss**

**Iis**

**Apache**

**Nginx : it is type of open source server which help to deploy the application.**

**Create the html, css and js file**

**Then create the Dockerfile**

FROM nginx:latest

COPY index.html /usr/share/nginx/html

**Build the image**

**docker build -t myweb-mern . -f Dockerfile**

**nginx server by default port number is 80**

**docker run -d -p 9292:80 myweb-mern**

**Then open the browser and run the application using the command as**

[**http://localhost:9292**](http://localhost:9292)

**Day 6**

**08-02-2022**

**Creating image for Angular Application**

**When we create the angular project google provide internally web server. Which run on default port number ie 4200.**

**Angular application we are running on development mode.**

**First we will create the Angular project**

**ng new angular-docker-app**

**after ng build you can see dist folder insider angular project folder and inside dist folder you can see another folder with projectname and that folder contains all our angular build files.**

**Build the project using command as**

**ng build if you get the error you have open the file as**

**.browserslistrc**

**And comment the code as**

**#last 2 Safari major versions**

**Then build using the command as**

**ng build**

**Create the Docker file inside the angular project**

**Dockerfile**

FROM nginx:latest

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

**Then build the image**

**docker build -t my-angular-mern . -f Dockerfile**

**nginx default port number 80**

**docker run –d –p 80:80 my-angular-mern**

**So before push we have give tag for that image. Tag is just like identity or version**

**Syntax**

**docker login**

**it ask username and password**

**please write username of docker hub and password (plz type it take those password but doesn’t display on console).**

**docker tag imageName:latest dockerhubaccount/imageName**

**Example**

**docker tag my-angular-mern:latest akashkale/my-angular-mern**

**docker push dockerhubaccount/imageName**

**Example**

**docker push akashkale/my-angular-mern**

**docker pull akashkale/my-angular-mern**

**docker run –d –p 81:80 akashkale/my-angular-mern**

**CI and CD Continuous integration and Continuous delivery or deployment**

**Manager ----🡪 Create the project structure or skeleton**

**Push this code in remote repository.**

**Git hub**

**Dev1 git clone url**

**Login**

**Dev2 git clone url Remote repository**

**Product info**

**Dev3 git clone url**

**Customer details**

**Code changes made by individual team members are merged together into shared repository (git repository). This phase is known as integration phase.**

**After merge another person code we have to build the project means**

**Run the application, deploy the application.**

**CI and CD : Continuous integration and Continuous delivery or deployment.**

Jenkins is an open-source automation server in which the central build and continuous integration process take place. It is a self-contained Java-based program with packages for Windows, macOS, and other Unix-like operating systems. With hundreds of plugins available,



**First pull the Jenkin software**

**docker pull jenkins/Jenkins**

**Run the Jenkin (default port number for Jenkin server is 8080)**

**docker run –p 8080:8080 jenkins/jenkins**

**after few second you can see alpha numerical password in terminal**

**Then open the browser and type as**

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

**it ask the password so type that password in textfield and install the suggested plugin**

**Jenkin pipe line : Jenkin pipe line is use to execute more than task which internally connected to each others.**

**Build Phase**

**Environment setup**

**Verify the version**

**Run the application**

**Test the application**

**Deploy the application**

**Day 7**

**09-02-2022**

**Grunt :**

**Grunt is a JavaScript task runner.**

**What is Task runner :**

**We can say task runner is one word automation tool do more than one task automatically.**

**Task runner help us to automates these task and perform these task synchronously and asynchronously.**

**About the Task runner**

**CSS file compression**

**Unit testing**

**Build deployment**

**Converting ts to js**

**Etc**

**To do the above tasks we need so many different reliable tools and it is tedious task to find such types of reliable tools. One tool can’t do all types of task.**

**Grunt take help of Node js to do the all types of task. Grunt is command line tool which run on Node JS.**

**Create the Grunt projects**

**Inside this folder create another sub folder as**

**Simple Task**

**Install the grunt**

**npm install grunt-cli –g**

**now create the package.json file using the command as**

**npm init**

**To run any type of task we have to create the Gruntfile.js**

**This file provide the all task configuration details.**

**npm install grunt –D (only development mode)**

**grunt**

**Create the folder as GrountFrontEndProject**

**Create package.json file using npm init command.**

**Then install the grunt module**

**npm install grunt –D**

**Grunt provide lot of dependencies for different type of task runner.**

**In Grunt we call as Minification. What is minification**

**Minification also known as minimization, is the process of removing all unnecessary character from source code without altering its functionality.**

1. **Html minification**
2. **JavaScript file minification**
3. **Css file minification**
4. **A Task to create final build project.**

**Grunt pre-defined task**

1. **Clean task.**

**npm install grunt-contrib-clean –D**

1. **Copy task**

**npm install grunt-contrib-copy –D**

1. **JavaScript task**