Day1 : 22-01-2022

Phase 4 : Testing and deployment

Software testing

Grunt task runner for JavaScript

Docker

CI and CD : Jenkins

Cloud computing

AWS

Ec2, S3 and EBS modules

Graph QL : Self learning

Elastic Stack : Self learning

Testing : Testing is use to find the defects or errors or bugs.

Read value a, b,

Process compute sum = a+b

Display value write b

Layer architecture

MVC

function operation() {

}

function add(a,b) {

// coding

var sum = a+b;

return sum;

}

Function sub() {

}

Function mul() {

}

Testing mainly divided into 2 types

Black box testing

Input -------------------🡪Process ---------------------🡪Output

Raj,123 success

Failure

Ravi, 123,

Age

21 60

Age between 21 to 60

>=21 <=60

<21 >60

Special symbol

-ve number

White box testing

Input --------🡪Process -----------🡪Output

Unit Testing :

Test suite and test case

describe(“”,()=> {

it(“”,()=> {

})

})

Unit testing : Unit testing is a kind of software testing method in which each individual and independent part of the source code or functions tested developed by developer.

Unit can be function or method or module or class.

If we do unit testing for UI side then we can call front end unit testing.

We can do backend side unit testing with respective framework.

Front end unit testing : Jasmine :

Jasmine is open source library which help to do the unit testing for Client side as a well as Server scripting language.

Jasmine provide pre-defined function which help to write test suite, test cases and expect functionality to check actual and expected output.

Old version of jasmine to generate the report or output we were depends upon Another ie Karma.

Karma is test runner for the Jasmine testing framework.

But new version Jasmine provide pre-defined plugin which help to generate the report or result without depending upon the Karma test runner.

Client Side JavaScript testing : Jasmine framework + Karma with old version Jasmine tool

Jasmine Framework with inbuilt plugin to run the tool.

Server Side JavaScript testing (Node JS): Node JS provided pre-defined external module jasmine which help to do the Testing for Server Side JavaScript (Node JS). In Server side JavaScript karma not required.

Mocha : Mocha is light weighted library which upon Jasmine. Using Mocha we can testing for Client side as well as server side testing.

Jasmine Vs Mocha

Mocha with Chai

Server Side JavaScript testing we can do with Jasmine or Mocha with Chai.

Client side testing Using Jasmine Plugin

Test suite : Test suite is use to write more than one test cases as well as another test suite.

To write the test suite using testing framework.

describe(“message”,callbackfunction);

Example

describe(“Operation Testing ”,()=> {

});

Test case : Test case is a type of test function which help to check the function functionality with the help of more than one except functions.

Syntax

it(“message”,callback);

describe(“Operation Testing”,()=> {

it(“Additing Testing”,()=> {

// write the code to call the function and

// using expect function we will check actual and expected output

})

})

Assert function : Jasmine framework provided lot of pre-defined assert function in the form of expectXXX() syntax to check actual and expected output.

Day 2 : 23-01-2022

.toBe(expected)

expect().toBeCloseTo(expected, precisionopt)

expect().toBeDefined()

expect().toBeFalse()

expect().toBeFalsy()

expect().toBeGreaterThan(expected)

expect().toBeGreaterThanOrEqual(expected)

expect().toBeInstanceOf(expected)

expect().toBeLessThan(expected)

expect().toBeLessThanOrEqual(expected)

expect().toBeNaN()

expect().toBeNegativeInfinity()

expect().toBeNull()

expect().toBePositiveInfinity()

expect().toBeTrue()

expect().toBeTruthy()

expect().toBeUndefined()

expect().toContain(expected)

expect().toEqual(expected)

expect().toHaveBeenCalled()

expect().toHaveBeenCalledBefore(expected)

expect().toHaveBeenCalledOnceWith()

expect().toHaveBeenCalledTimes(expected)

expect().toHaveBeenCalledWith()

expect().toHaveClass(expected)

expect().toHaveSize(expected)

expect().toMatch(expected)

expect().toThrow(expectedopt)

expect().toThrowError(expectedopt, messageopt)

expect().toThrowMatching(predicate)

expect().withContext(message)

Jasmine hook functions.

beforeEach() :This is life cycle or hook function it will call before each it function.

afterEach() : it will call after each it function

beforeAll() : it will call only once before all it function

afterAll() : it will call only once after all it function.

It()

deposit amount

It()

without amount

Jasmine client side testing using node js

First create the folder and install two dependencies

**Npm install –D jasmine-browser-runner jasmine-core**

**npm install --save-dev jasmine-browser-runner jasmine-core**

or

npm install –D **jasmine-browser-runner jasmine-core**

Next run the command as

npx jasmine-browser-runner init

This command is use to create the spec folder and jasmine.browser.json file.

npx jasmine-browser-runner serve

The application run on default port number ie 8888

<http://localhost:8888>

Day 3 : 29-01-2022

Angular Framework testing using Jasmine and Karma

Ionic framework for hybrid mobile application

React JS test using JEST library

With Redux or Flux

React native

Component angular as well as react js

@Input : parent to child

@Output or @ViewChild : child to parent

Shared service or sessionStorage or rxjs using observable : sibling component

In React JS using props we can share the data between one component to another component depending upon the their relationship.

Redux is state management tool which provide centralized tool to share the data between more than one component doesn’t matter what is the relationship between two components.

Create new angular project using ng command as

ng new angular-testing

routing no :

styling css

Jasmine is testing framework which provide pre-defined function ie describe, it and more than one expect.

It may be jasmine or jest or mocha.

Karma is a test runner which help to generate testing reports.

Angular Utilities testing classes.

TypeScript

We use special type of class using decorator @Component, @Service, etc

Angular providing pre-defined module ie @angular/core/testing

TestBed is pre-defined class which provide set of method which help to do the testing for Angular component and service.

To start the angular testing we have to use the command as

ng test

ng g class employee this command is use to create model or normal type script class.

Angular service testing

ng g s fake

Day 4 : 30-01-2022

Docker :

OS :

Window, Linux , Unix or Mac etc

VM : Virtual Machine

Base Machine : 16 GB RAM

Unix Guest OS or Virtual Machine

4 gb

12 GB

10 Guest OS

1gb

Docker : Docker is a advanced os virtualization software platform that makes it easier to create, deploy and run the application in a Docker Container.

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

Virtualization let you divide a system into series of separate section, each one acting as distinct individual system. The virtual environment is known as virtual machine.

MQ or

Python

Table

View

Docker is use to create containerization application.

Virtualization Vs Containerization

Virtualization is an abstract version of physical machine.

While containerization is the abstract version of an application or tool or application software/system software.

Docker providing images which help to run the application that application internally run on os.

Containerization is use create application, deploy the application and run the application on Docker engine.

Container : Container is a run time environment to the run the application.

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

Docker Image : It is file system and configuration of our application which is use to create the container. Docker images are the source code of our container.

Docker file : A Docker file is blue print or set of instruction that defined how our images are built.

Docker registry Docker provide Docker hub it is like a git hub which help to publish the our own images as well as we can pull other open source images and run in our machine.

Docker hub account create.

docker --version

docker images : this command display all images present in local machine.

docker pull image-name : This command is use to pull the image from hocker hub to local machine.

docker run image-name/image-id : This command is use to run the image when we run the image the application present in image run.

Virtual Machine

1. It is not a container based model. They use user space along with kernel space on OS. Here the information store permanently.
2. It doesn’t share the host kernel.
3. Virtual machine don’t start quickly and lead to poor performance.
4. It can run only a limited number of VMs on a system depends upon the Base OS.

Docker Container

1. In Docker, the container share the host OS kernel space. Here the information store or hold temporary.
2. They share kernel space through images.
3. Docker container can start up quickly and result in less boot-up time.
4. With Docker container, user or programmer or developer can create the application and store it into a container images for temporary purpose.
5. Using Docker we can run multiple container at a time.

Create image to run the data information using busybox image

**Dockerfile**

#pull the image from local or docker hub

FROM busybox:latest

#run the date command.

CMD ["date"]

To build the image we have to execute the command as

Docker build –t imageName . –f Dockerfile

docker build –t my-busy222 . –f Dockerfile

Create image to run the node js application

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 to Node js with docker"+name

}

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

**Dockerfile**

# pull the node js image base upon alpine image run time environemnt is ready

FROM node:latest

# copy app.js file into image in current path.

COPY app.js .

# open the command and run app.js file in node js image

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

docker –t my-node222 . –f Dockerfile

. : current location of file

-f : file