TestNG

Configure testNG with project

① Add testNG dependency in pom.xml [add dependency]
it will download automatically & add in maven project

② download plugin → to run code

Steps to download ⇒ help → install new S/W → google (testng eclipse url) eclipse plugin in testng

(dependency is used to build the project but)
which depency added in pom.xml take latest version link &
paste on work with [here] https://testng.org/testng-update-site/6.14.3/
→ press enter → next → install it then restart eclipse

Dependency → help to write code
plugin → help to execute code

testNG → Java unit testing framework → used by java dev/automation QA
python → pyTest, Unittest
PHP → phpunit
JS → mocha, jasmine
C# → Nunit

test cases
decorate tcs
maintain tcs
priority tcs
depends on
data provider - test data
parameterization
html report
xml report
listeners - extent/allure report
annotations - pre cond^n / post steps / tcs
assertions → validation point
↳ ① hard ② soft

test Runner → testng.xml → can run testng file
In testNG → No main method, it has its own mechanism to execute

src/main/java → lib, util, page class
src/test/java → test classes → testing code

public class AppTest {

global pre condn → system is up running, user id, db connect

pre condn → user is logged in, user is blocked in or not

test steps → steps to reproduce (test script)
+
Assertions → Actual VS Expected

post steps → delete cookies or cache, logout, close browser

@ BeforeSuit #import it

every annotation associated with method.

```
public void createUser(){
    S.O.P("create User - BS");
}

@BeforeTest
public void connectWithDB(){
    S.O.P("BT - connect with DB");
}

@BeforeClass
public void launchBrowser(){
    S.O.P("BC - Launch Browser");

@Before Method
    public void login(){
        S.O.P("BM - Login to App");
}

@Test
public void homePageTest(){
    S.O.P("Home Page Test");
}

@Test
public void searchTest(){
    S.O.P("SearchTest");
}

@Test
public void priceTest(){
    S.O.P("priceTest");
}
```

④ AfterMethod
public void logout(){
S.O.P("AM — logout");
}

⑤ AfterClass
public void closeBrowser(){
S.O.P("AC — close Browser");
}

⑥ AfterTest
public void disconnectWithDB(){
S.O.P("AT — disconnectwithDB");
}

⑦ AfterSuite
public void deleteUser(){
S.O.P("AS — delete User");
}
}

ctrl + shift + O
↓
Automatic generate all imports
↑
Jar files

① what is annotation & its sequence

execute once
BS
BT
BC

executed before each n every method

BM
homePageTest
AM

BM
priceTest
AM

BM
SearchTest
AM

AC
AT
AS

Test run alphabetically

* if you are writing any test method then name of test method always ends with Test. eg. LoginTest (it is good practise)

* before and after are only pre & post condn not a test

if you don't want to alphabetical order then can use priority → inbuilt testng variable it takes integer

priority = 1 ⇒ execute 1st

priority = 0 ⇒ execute 1st

We can have negative priority then it will execute first.

⇒ what if all method have priority = 1
then it will execute alphabitically order.
alphabetic order

⇒ if there are some test are with non priority & some test are having priority then preference will be given to test which is not assign any priority then priority based test case

⇒ If we don't define any priority all the test cases will be define with 0 priority & order will be alphabitically.
priority निश्चित है तो उसी क्रम में चलेगा।

Create Class ⇒

⇒ testNG method should not return any thing only test methods will be there.

Before method is good than before test

↓

each time login to Applicn

↑

what if suddenly driver stop working at test 5 & again 5 test are remaining

DependsOnMethodsConcept ⇒

```
@Test
Public void loginTest(){
        S.O.P("login to app");
}
@Test(dependsOnMethods = "loginTest")
   public void homeTest(){
        S.O.P("home test");
}
@Test(dependsOnMethods = "loginTest")
   public void searchTest(){
        S.O.P("search test");
   }
}
```

login test fail then rest 2 methods will not execute

↳ Test should be independent not to be dependent

↳ try to avoid priority

CRUD ⇒ how will you update

① create user ② take the user id from the new user (userId=3) ③ pick the same user id to update the user ④ delete the user

public class ExpectedExceptionConcept {

@Test (expectedExceptions = {NullPointerException.class, ArithmeticException.class})

class array

public void searchTest() {
S.O.P ("search test");
int i = 9/0;
}

use for negative testing

InvocationCount

public class InvocationCount {

@Test (invocationCount = 10)
public void getUser() {
S.O.P ("get user");
}
}

UN | PW

excel sheet

it will supply the data

@DataProvider
Public Object[][] loginTestData() {
return new Object[][] {
{"admin@gmail.com", "admin123"},
{"cust@gmail.com", "cust123"},
{"Vendor@gmail.com", "vendor123"},
};
}

@Test (dataProvider = "loginTestData")
public void loginTest(String userName, String password) {
Assert.assertTrue(doLogin(userName, password));
}

We need Runner to execute 100 test cases

for this ⇒ ① create same folder
src/test/resources

② right click → folder → testRunners.

③ right click → file → testng.xml
↑ name can be anything

Create multiple test block for individual class

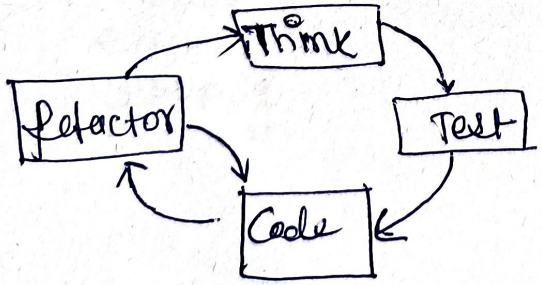
```
<test name="GoogleTest">
  <classes>
    <class name="AppTests.GoogleTest" />

  </classes>
</test>
```

after run this testNg.xml then refresh project one test-output folder is created.

Can publish report by using Jenkins.

TDD ⇒ used for unit testing point of view. whenever you design test cases either in java test cases

Test centric approach
→ Req → test cases → failed → refactor code → passed
→ test cases → failed → refactor → passed



Base Test ⇒ before test, after test

before each and every test it will execute

if GoogleTest extends BaseTest

↑ it will execute first

What if you want to execute your test cases on some other browsers ?

use parameterization → add code in testng.xml

```
<test name="GoogleTest">
    <parameter name="browser" value="chrome"/>
    <parameter name="url" value="https://www.google.com"/>
    <classes>
```

2 parameters

Sequence of execution ⇒ test will execute from testNG.xml then go to the base class

↓ write cross browser testing code

we need to handle parameters ⇒

```
@Parameters ({"browser", "url"})
@BeforeTest
public void setUp (String browserName, String url) {
    if (browserName.equalsIgnoreCase("chrome")){
        WebDriverManager.chromedriver().setup();
        driver = new ChromeDriver();
    }
    else
```

2 parameters

2 parameters

Verbos means label of logs

Verbos=10

give so much internal information about execution

max val of verbos = 10

min — 11 — verbos = 11

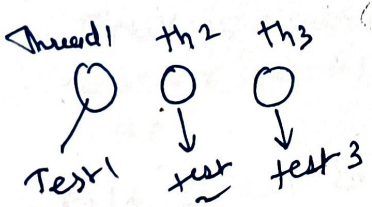
Verbos ⇒ gives more logs (get better console log)
individual test Reports

<Suite name = "TestNG Sanity Suite" verbose = "10" >

(parallel & thread count)

thread count = "3" # Run test in parallel mode

parallel = "tests"



⇒ @Test(enabled = false) ⇒ this test will not participate

⇒ <classes>

<class name = "AppTests.GoogleTest" />

include

exclude

grouping

CI ⇒ CI is the process of automating the build & testing of code every time a team member commits changes to version control.

CD ⇒ It is the process to build, test, configure & deploy from a build to a production environment.

deploy code into server, so it will reflect in browser

Application server ⇒

An application/web server is a server specifically designed to run application (or) it is server to host applications.
It handles Http requests & send response calls over http protocol.

Hosted server ⇒ where you placed your s/w
जहाँ exe application download करते हैं।
eg. Linux is a hosted server where i downloaded application server eg. apache tomcat where i will push my application code.
mic will be uniquely identified using ip address
every ~~post~~ application will have diff. port No.

Connect b/n hosted server, application & url

254:188:156:40 # 8080 ← tomcat server
hosted mic ip address ↳ listning all http responses

↳ with the help of this can access my website/(port no) (hosted on) mic.

Build the code ⇒ build java code using maven get war file ↳ artifact
copy this file to tomcat server (then our application will be rendered)
dist - ng - build # dist folder will be created (build angular code)
mvn clean install # build Java code

copy build files to server

To keep our developed app running on server & access it from anywhere, we should first build the code so that build files will be generated. these build files should be placed into app/webserver.
& then we can access the developed app directly on the browser with Hosted Server IP Address : port number or with domain name if mapped.

Deploying Hello World App in Tomcat Server (manually)

step① ⇒ build project and create war file
→ Apache tomcat used to deploy all our front end apps.
→ download apache tomcat → zip file → extract it
→ Build from cmd prompt → mvn clean install ⇒ build command
→ after every build → all files store in target folder
code → target → webapp → .war file
↳ copy this into tomcat

step②→ copy war into tomcat
→ verify server is start or not
↳ for this → go to bin folder of tomcat → startup → cmd prompt
error may occur like port is already in use then provide
2 command →

netstat -ano | findstr :8080 /
o/p ⇒ pid निकालो
taskkill / PID 37401 F # delete करो running task

How access server from browser?
url ⇒ localhost :8080/webapp → name of war file
↑ → where tomcat is there
given IP address

CI/CD can be achieve by Jenkins
↳ to implement this also require docker, kubernetes

using Jenkins, can automate this deployment process

~~connect github with~~

⑧ Sign up Jenkins instance in Windows for deployment ⇒ Automate deployment

↳ install Jenkins

everyone will use linux m/c to deploy

Step ① ⇒ build project & create war file

Jenkins → new item → OK → source code management → git → paste Git repo

→ Build → maven goals → test install → window batch command

↳ build

Jenkins ← github acc. code ~~shift~~

add build steps → Invoke

copy war file to tomcat ⇒ through cmd prompt this can be done automatically in Jenkins

copy project path from right click on project (copy path)

paste on cmd prompt add .war file path too (cd "webapp/target)

project path | copy | tomcat webapp path ↵

Jenkins

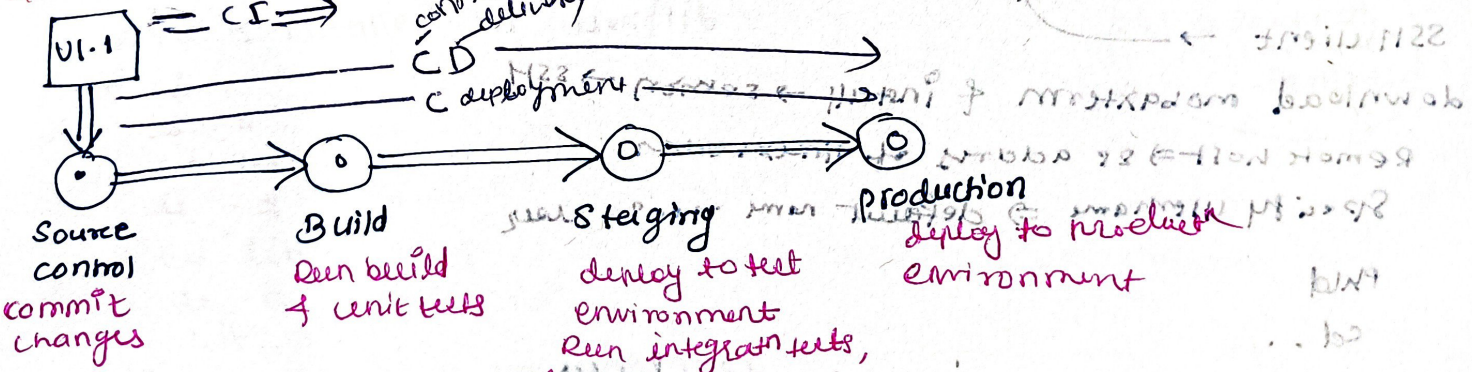
Code management

Build

add build step → execute windows batch cmd

cmd ⇒ cd webapp/target

actual cmd



(*) Java, Maven and Jenkins setup on AWS linux EC2 Instance

create 2 linux m/c

1 m/c ⇒ Jenkins → to build & deploy code

2 m/c ⇒ deploy war file in another m/c (tomcat)

All devops happen in linux m/c

① Create AWS EC2 Instance

Can create n no. of linux/windows m/c using aws cloud can create virtual m/c

AWS console Login(a) → sign in to the console → EC2 → launch Instance

Launch EC2 (linux) instance from AWS console with necessary configurations

SSH client & how to connect to AWS linux Servers

Putty, mobaxterm → to connect & communicate with linux

SSH client →

download mobaxterm & install → session → SSH

Remote host ⇒ ip address of linux server

Specify username ⇒ default name ec2-user

Pwd

cd ..

ls → show all directory in your current folder

(*) To download Java, maven we need to use login as root

cmd ⇒ sudo su - ↵ root user

ls -a

cd root

~ → option should be there to recognize as root

Basic linux commands

yum ⇒ cmd line tool to download s/w from internet

yum is the primary tool for getting, installing, deleting, quering & managing s/w packages.

```
yum install links
yum install java-1.8.0-openjdk-devel  #write for root user in nobaterm
yum remove javel
java -version
yum install java-1.8.0-openjdk-devel
java version
ls -a  # .bash profile → it contains environment variable
cat .bash profile  # display content of file
pwd  # /root
cd ..  # home
ls
cd root # go into root folder
```

Setting java & maven path in Bash profile in linux

java is installed in usr/lib

```
cd ..
ls -a
cd usr
ls -a
cd lib
ls -a
exit
sudo su -  # log with root
```

(*) vi .bash_profile # can edit file

add #
JAVA_HOME = path of jdk
PATH = $PATH : $HOME/bin : $JAVA_HOME # path

Click on escape
:wq # save & then quit
↳ cat .bash profile

:q! ⇒ quit save without saving

Install maven in Linux
Configure maven in Linux from Apache HTTP server

easy learning → install maven on linux

maven.apache.org

download → Binary tar.gz → get this link

maven is not in yum pkg

wget ⇒ wget solely lets you download files from HTTP/HTTPS or FTP server.

You give it a link & it automatically downloads the file where the link points to. it helps to download binaries

linux terminal ⇒ wget paste copy link # download from official website

tar xzvf apache-maven-3.6.3-bin.tar.gz # unzip file come from maven official website

set maven path

where maven install ?

ls -a

move file from root home dir to / [~ → root to / → origin] cmd → cd ..

ls -a

cd root

cp -r apache-maven-3.6.3 /opt

(apache-maven-3.6.3: current location; /opt: this directory)

cd /opt

ls -a

set path in bash profile

open bash file in edit mode as

MAVEN_HOME = /opt/apache-maven-3.6.3

M2 = /opt/apache-maven-3.6.3/bin

PATH = $MAVEN_HOME : $M2

press Escape

:wq # exit with saving

Install Jenkins on Linux

wiki.jenkins.io → installation of a newer version ⇒ fire all commands on Linux

Rpm ⇒ default open source

service jenkins start/stop/restart

chkconfig jenkins on

jenkins start on id/post no. → url
→ get from AWS

generate password from linux enter on website & start jenkins

Create new jenkin job for deploying the apps:-

create job on jenkins → freestyle project OK → give project repository details

① create war file

manage jenkins → Global tool configuration → add JDK
→ save all configuration
take path from linux & store in jenkins

linux cmd → echo $JAVA_HOME (paste in jenkins JAVA_HOME)
echo

Set Git path → which git.
yum install git ← install in root user

add maven
echo $MAVEN_HOME

configure → compile project & create war file (CI)

Deploying Apps into Linux via Jenkin plugins

How to deploy the App into another linux server from Jenkins

War file/Artifact → it is responsible to renders your app in server

Send war file to another server.

M/C A → linux Jenkins server

at present real time application hosted in a seperate m/c/server

Hosting m/c

Linux A Jenkins → create WAR file → push into linux m/c B

Linux B → war file → tomcat

Authenticate server with user

ec2-user → Authenticate using user not by using root level

Manage Jenkins → manage plugins → Available → search for publish over SSH (install it) send build artifacts over SSH to another linux server.

manage Jenkins → configure system → .

Publish over SSH → add → linux B server, user name, pass

Jenkins - configuration

SSH servers ⇒ Name ⇒ Application_server (from aws)

Host name ⇒ IP from AWS

Username ⇒ ec2-user

Remote directory ⇒ . (whatever installed will store in home directory)

advanced ⇒ give password

(create pwd in linux user m/c ⇒)

↳ password ec2-user

↳ Sudo su -

↳ ls -a

↳ cd ..

↳ ls -a

↳ cd home

↳ ls -a

↳ password ec2-user

↳ ec2-user

↳ ec2-user

Manually enable pwd Authentification

make sure you are in root

vi (cmd) → from internet

file will open search for pwd authentication

type i (insert mode will open)

esc + :wq ← # updated that file changes should be reflected

↳ service sshd reload

Jenkins → test configuration →

↳ sudo su ec2-user

ls -a
clear
cd ..
ls
cd home
ls -a
cd ec2-user/

~ ↳ home page of user

Jenkins will push in home path of linux.

Create Job in Jenkins

Jenkins → New Item → (deploy to App server) →

goal → post build Actions → send build Artifacts over SSH

linux → cd ec2-user/ # build को Jenkins पे build हुआ file को linux folder में

ls -a

Jenkins connect with 2nd user (ssh plugin)

↳ rm webapp.war # delete file

Jenkins → build now

↳ ls -a

(phase 1) → ① Spin up New AWS instance for deploying application

Create New user with password

password ec2-user (make sure you are in root)

② enable pwd Authenticatn in below file path

vi /etc/ssh/sshd config

③ Reload service with service sshd reload

④ Add server into Jenkins from manage jenkins

⑤ Download publish over SSH plugin in Jenkins

⑥ Add Docker server SSH to Jenkins in the ~~Jen~~ job

⑦ Run the deployment job to deploy the artifacts into ~~docker~~ application Server from Jenkins server

Docker

Create container in Os. this container is fresh installed os.

vm os os
chrome firefox

install docker in own os → create container → schedule test

it is computer program that performs OS level virtulization.
it is tool designed to make it easier to create, deploy & run applicat by using containers.

In simple words, docker is a containerizat platform, meaning you can build your applicat, pkg them along with their dependencies into a container & then these containers can be easily shipped to run on other mic

Containers

deploy img which will have all the dependencies

An image is a lightweight, standalone, executable package of sw that includes everything needed to run an applicat: code, runtime, system tools, system libraries & settings.

container is the runtime instance of an img.

A dockerfile is where you write the instructions to build a docker img.

docker ps ↵ # to see any container is running in mic or not
(give container status)

docker pull Selenium/Standalone-chrome # whether img present in docker hub download into your mic then deploy this img into container then trigger test
↳ image name

docker pull img:latest
docker images

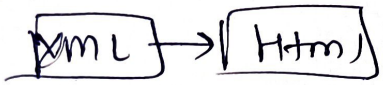
container means plane OS which will share host os resources like network etc. to deploy img so container

docker run # start container in my mic with the help of docker sw

docker run -d -p4444:4444 -v /dev/shm:/dev/shm img name:latest
- -d: run container in background so save memory allocation time
- -p4444:4444: local : container port
- deploy

Listener

- Listener is a ~~selenium~~ testNG feature.
- It will check at each & every annotation level.
- keep listening from 1 test-case to n test case
- it will ~~run~~ run in background



format

- If you apply Main listener it will give you plain report
- extent report listener - 3rd party library
  allure report listener

step ① ⇒ pom.xml ⇒ add dependency

~~java class~~