Jenkins

## What is Jenkins?

Jenkins is a self-contained, open source automation server which can be used to automate all sorts of tasks related to building, testing, and delivering or deploying software.

Jenkins can be installed through native system packages, Docker, or even run standalone by any machine with a Java Runtime Environment (JRE) installed.

**Installation**

Create Jenkins Folder in C:\Program Files

Click on this link to download Jenkins war file <https://get.jenkins.io/war-stable/2.387.3/jenkins.war>

Place it inside C:\Program Files\Jenkins

Open command prompt and navigate to Jenkins folder. C:\Program Files\Jenkins

Type “java -jar Jenkins.war” and click enter to start Jenkins. Copy the password displayed

<http://localhost:8080> is the url to open Jenkins in your local.

Enter password to unlock Jenkins

Create first admin user entering details

Enter admin, admin, admin, email id , click on save and continue.

After Login, we need to configurations inside Jenkins.

Dashboard-Manage Jenkins- global tool configuration- Click on JDK

Enter name “ JDK” and enter java\_home path. Example: C:\Program Files\Java\jdk-17.0.4.1

Dashboard-Manage Jenkins- global tool configuration- Click on Maven

Enter name “MAVEN\_HOME” and enter MAVEN\_HOME path. Example: C:\Program Files (x86)\apache-maven-3.6.3-bin\apache-maven-3.6.3

Apply and save

Go to Dashboard- Manage Jenkins-Manage plugins-enter “maven” in search .

Select all options related to maven- maven integration, maven invoker etc and click on Install without restart( adding only maven integration is also fine)

Restart and login to Jenkins

Test Execution can be done in different ways. 7008137561- narayan

**Run the test from local pom.xml**

# Click on Create New Item

# Enter project name and select Maven project

# Go to “Build” section- give complete path of pom.xml of your project, enter ‘clean install’ in “goals”.

# Apply and save

# Now click on Build now

**Run the tests from github**

# Login to Jenkins

# Create new item maven project

# Click on project-configure-source code section

# Select git in source code and enter githuburl

# Change master to main

# Under build- pom.xml and clean install

# Manage Jenkins- global tool configuration- enter the path of git bin folder. Ensure that git bin path is added in path variable in system “environment variables”.

# Click on Apply, Save.

**Auto build a job in Jenkins when there is any change in code on GitHub repository**

# Open Jenkins dashboard. Click on manage Jenkins

# Click on Configure system and under github configuration click advanced tab.

# Check 'Specify another hook url' for GitHub configuration.

# Now you will get a url in the textbox. Copy this url as it is required in the next steps.

# Now open your github repository. Go to settings -> webhooks -> add webhooks.

# Now paste the url from step 4 in the payload url section. Next click on just push the event Now you should be able to see the added webhook in the list of webhooks.

# Now go to jenkins dashboard. Go to your project configuration. In the build triggers section select github hook trigger for git scm polling. Save the changes.

Html publisher to be installed in Jenkins

Post build actions

Publish html reports



In end to end approach – we need new webdriver all the time.

Parameterization support in Jenkins

Below command needs to typed in Jenkins to add select the browser for test execution

Java -Dhudson.model.DirectoryBrowserSupport.CSP=””

A screenshot of a computer

Description automatically generated with medium confidence

A blue and white sign

Description automatically generated with low confidence

A screenshot of a computer

Description automatically generated

Similarly we can add parameter for environments

First add it in Jenkins and then ensure that below code is present in your base class

If Browser not null and not empty then get browser name mentioned in Jenkins, else get default browser present in configuration file.

A picture containing text, font, screenshot

Description automatically generated