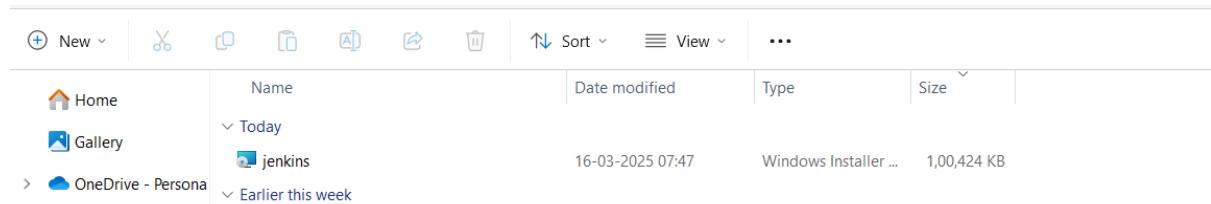


## JENKINS NOTES

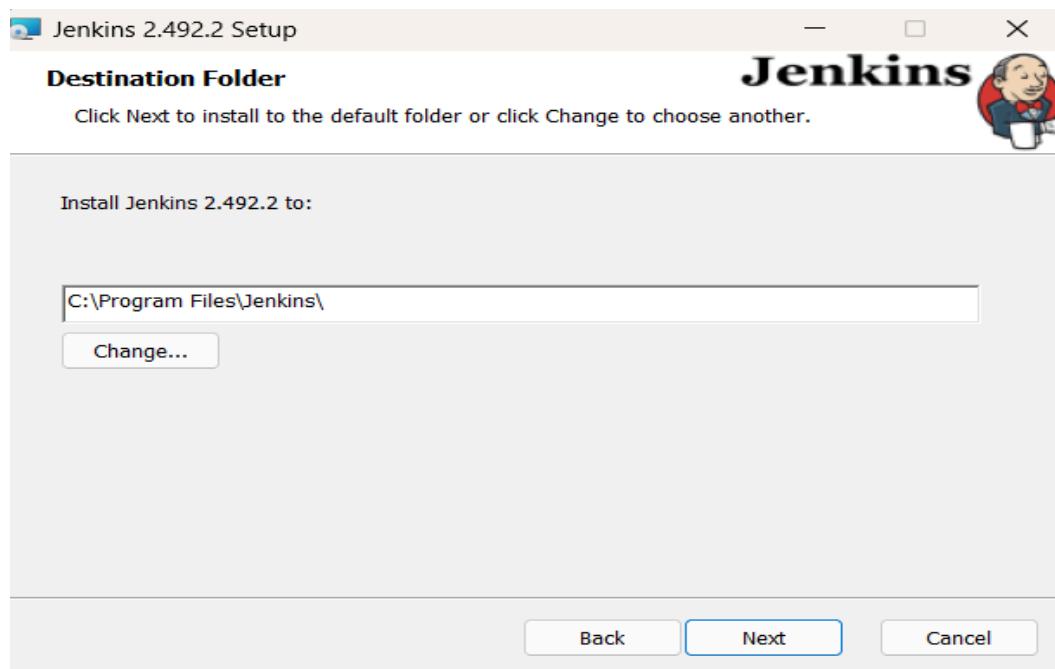
### JENKINS INSTALLATION

Step 1 : Download Jenkins from <https://www.jenkins.io/download/> - Windows Users should download the ".msi" file and not ".war" file

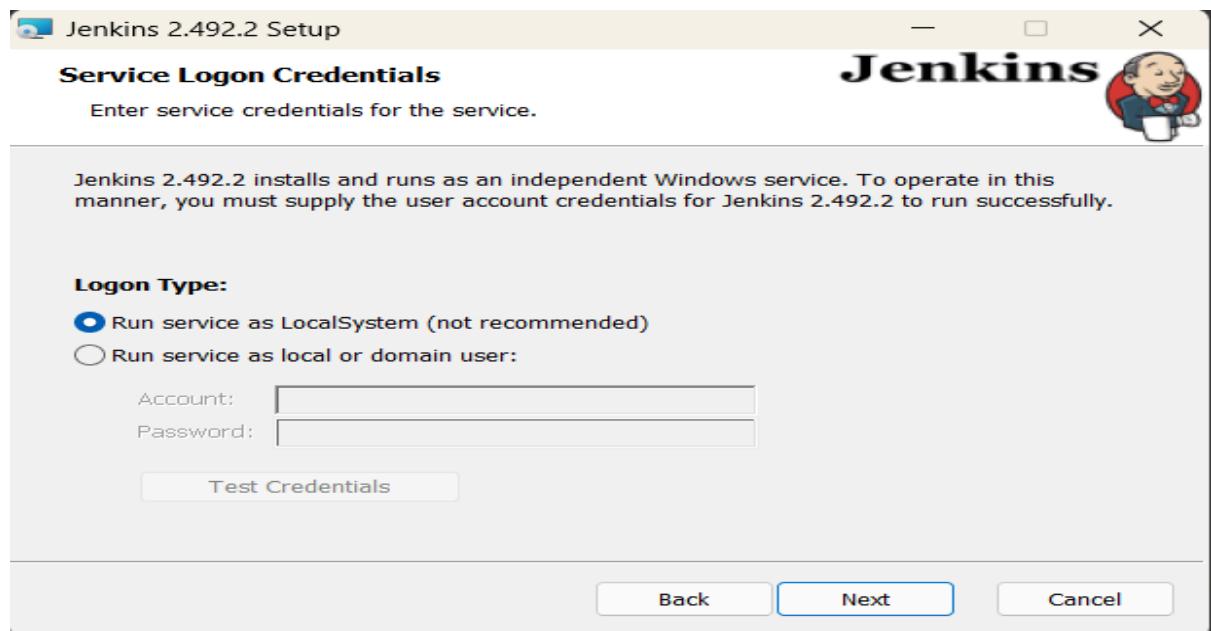
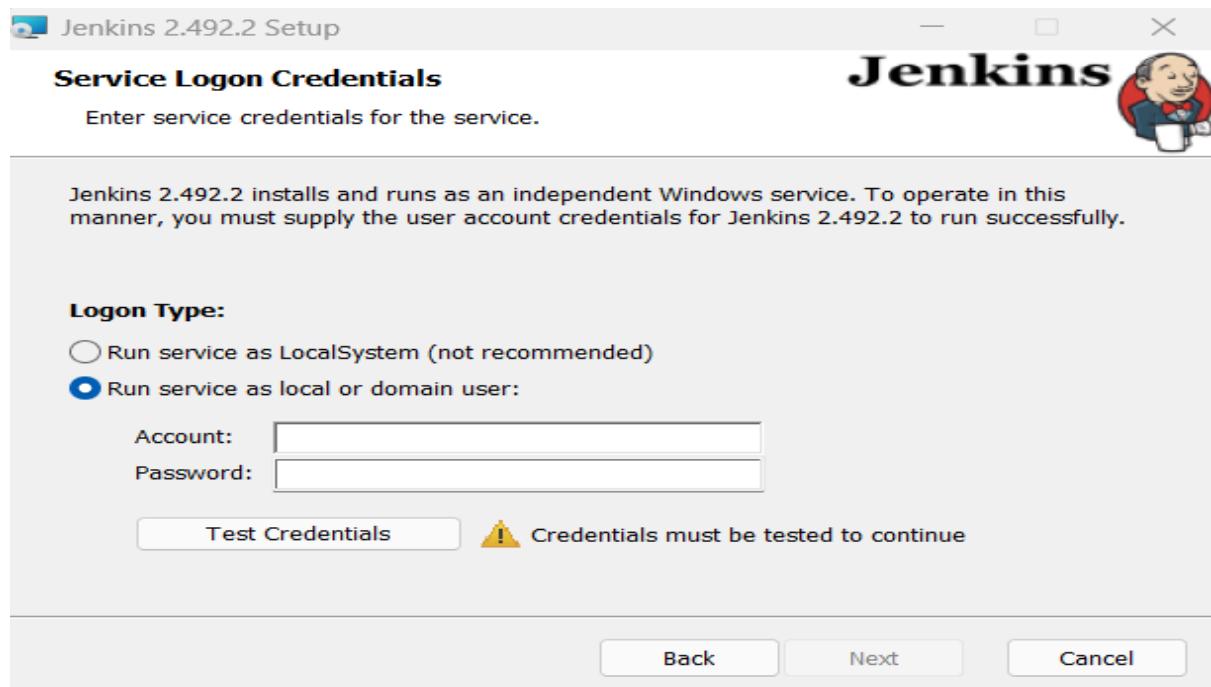
Step 2: Go to download and open Jenkins



Step 3: Click Next ---> Next



Step 4: In “Logon Type” select “Run Services as local system” and click Next



Step 5: Here we have to test the port , Click on the “Test Port” - If after clicking you get a green colour tick then your port is passed , otherwise if not getting the green colour tick then try changing the port and click TEST PORT again and then click next

Note: Remember the port number for future use

Jenkins 2.492.2 Setup

**Port Selection**

Choose a port for the service.

Please choose a port.

**Port Number (1-65535):**

 Click 'Test Port' button to proceed

It is recommended that you accept the selected default port.



Jenkins 2.492.2 Setup

**Port Selection**

Choose a port for the service.

Please choose a port.

**Port Number (1-65535):**

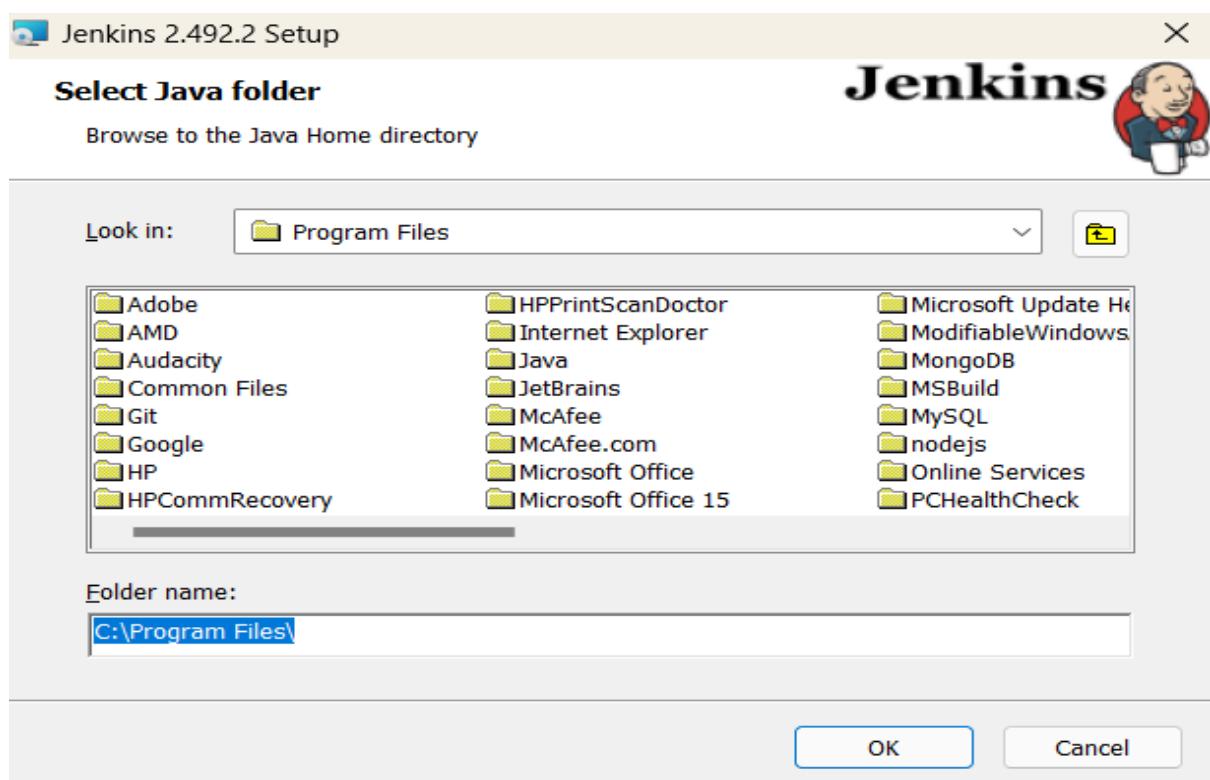
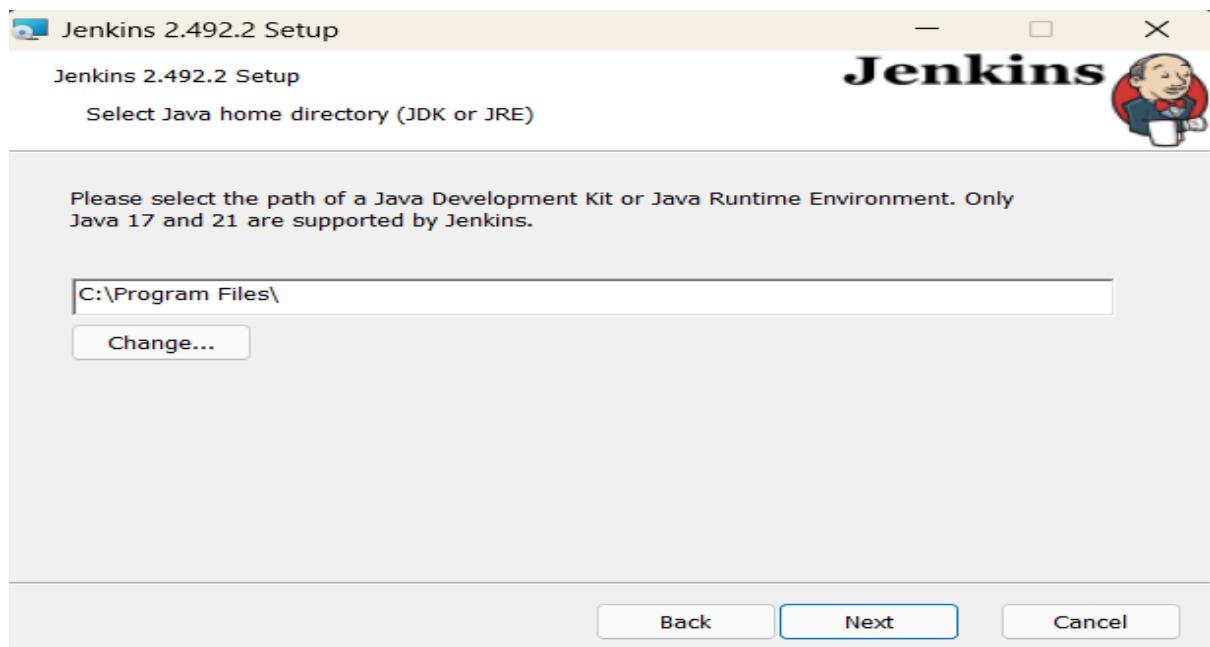


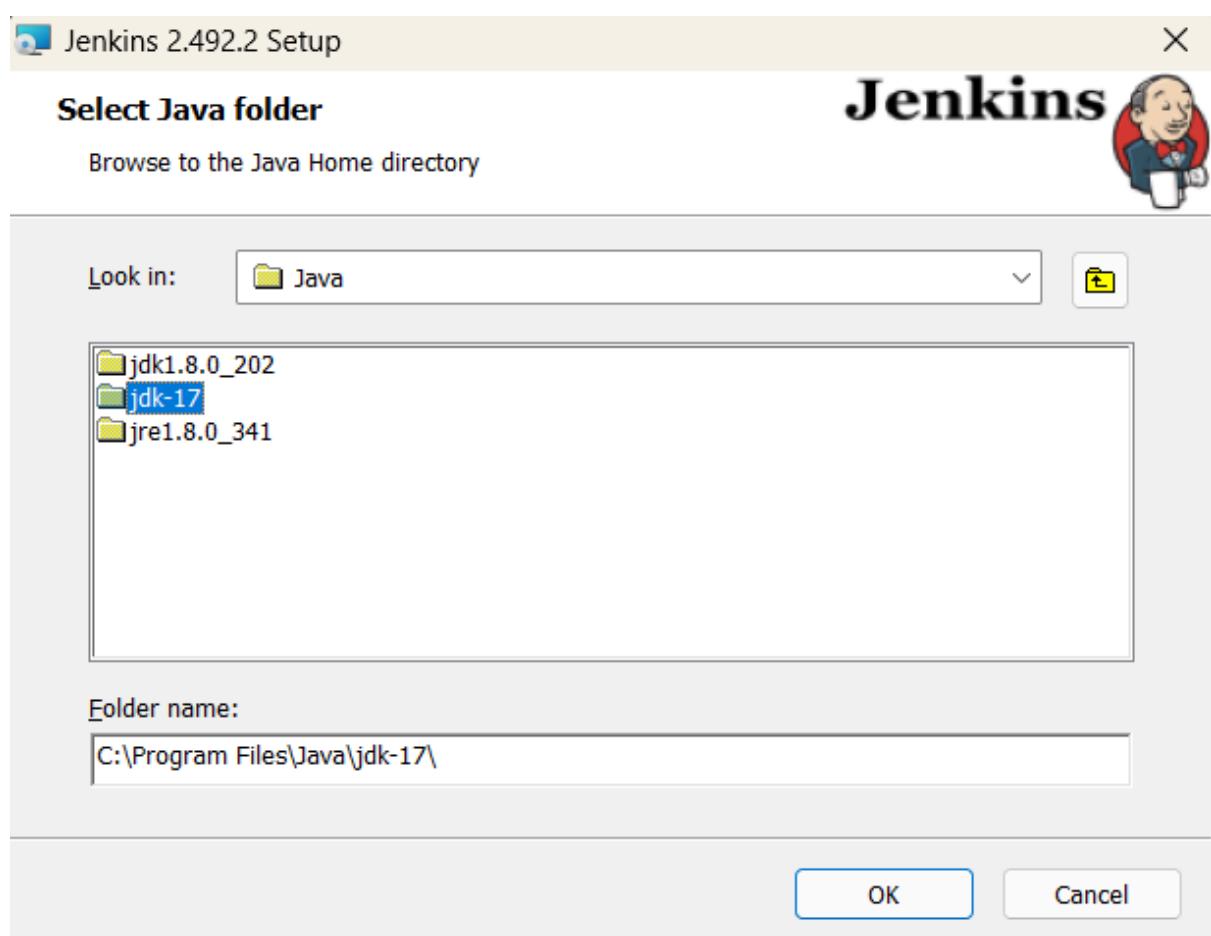
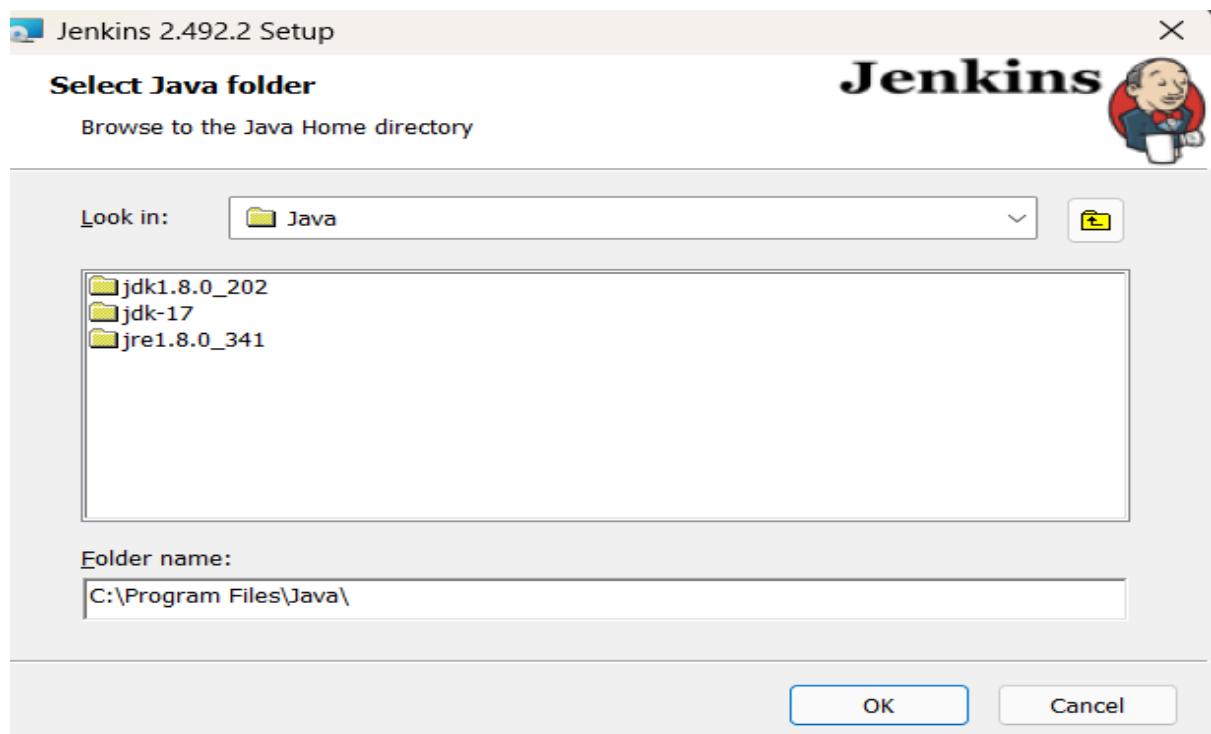
It is recommended that you accept the selected default port.



Step 6: Next you have to select the JDK path. Click on Change button ----->select java -----> Select JDK17 -----> click Next

Note: Only JDK 17 and 21 are supported

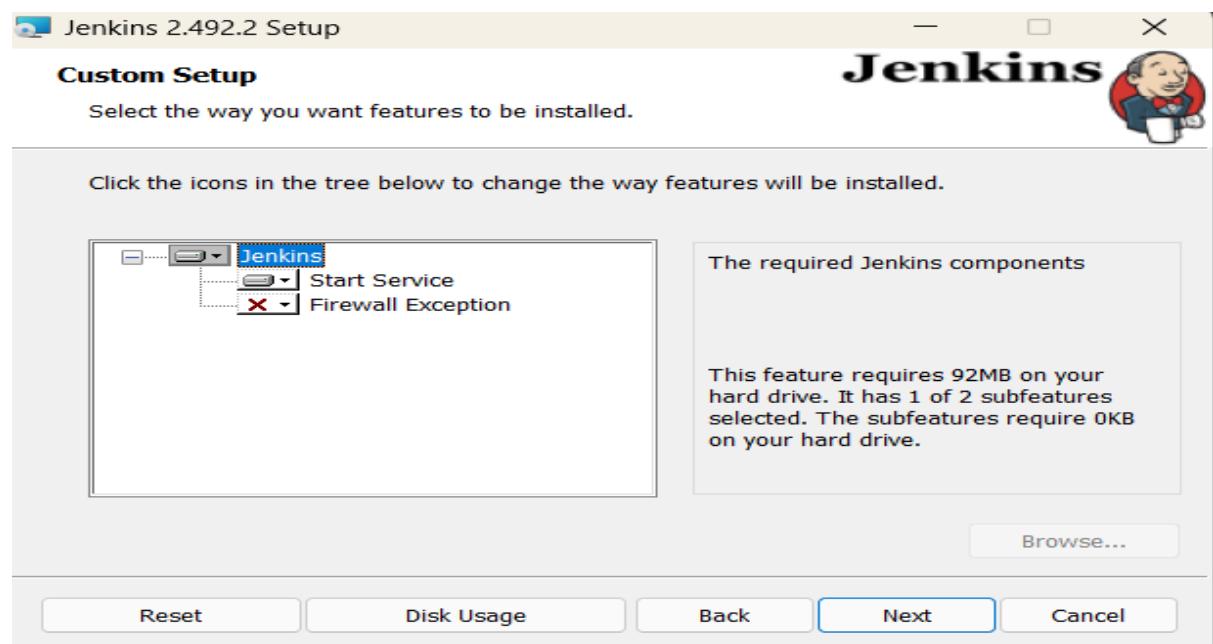


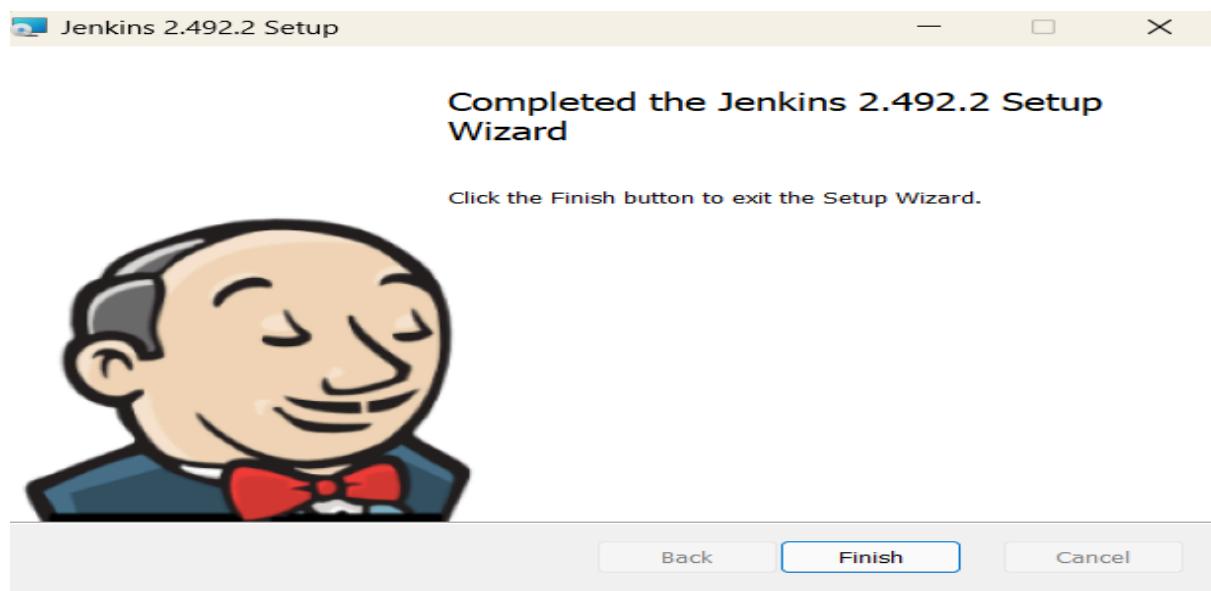
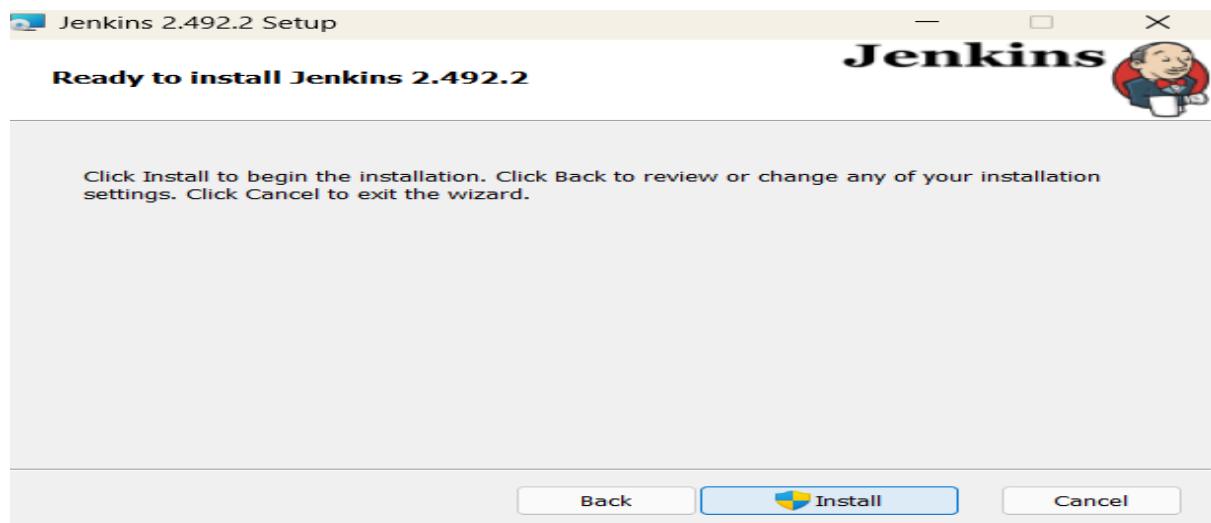


You can see the JDK path here and then click Next



Step 8 :Click Next -----> Click Install -----> Finish



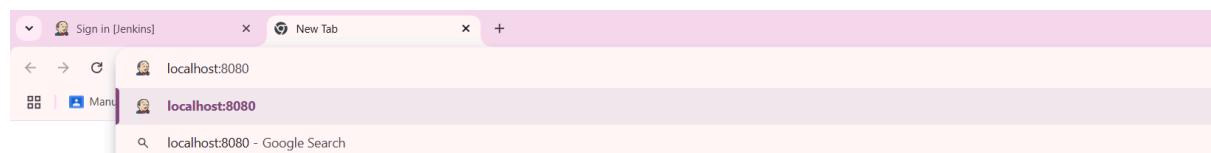


Your have Successfully INSTALLED JENKINS

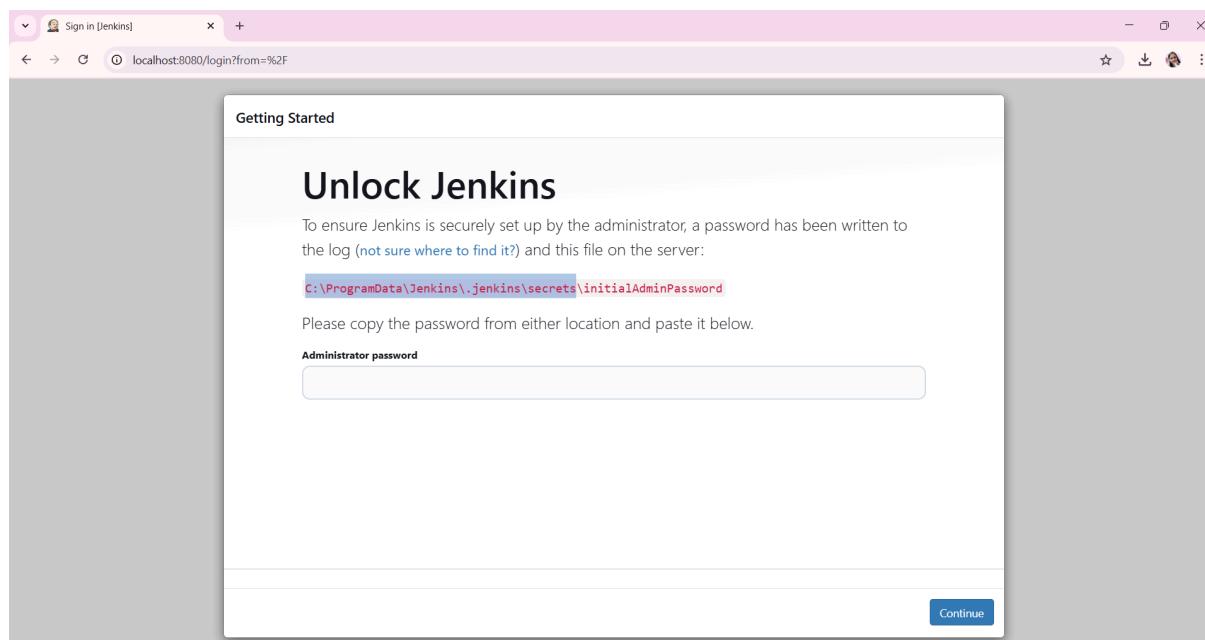
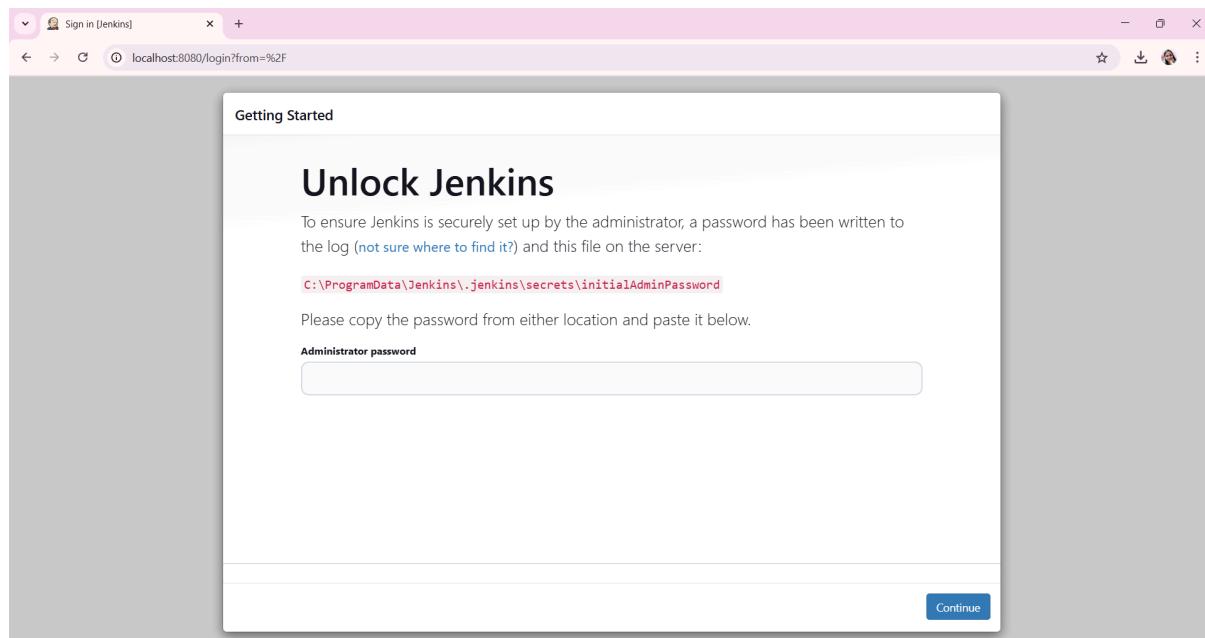
---

Step 1: Open a new tab in Chrome -----> in SEARCH BAR type **localhost:8080**  
[If you have changed the port number then replace 8080 with you port number]

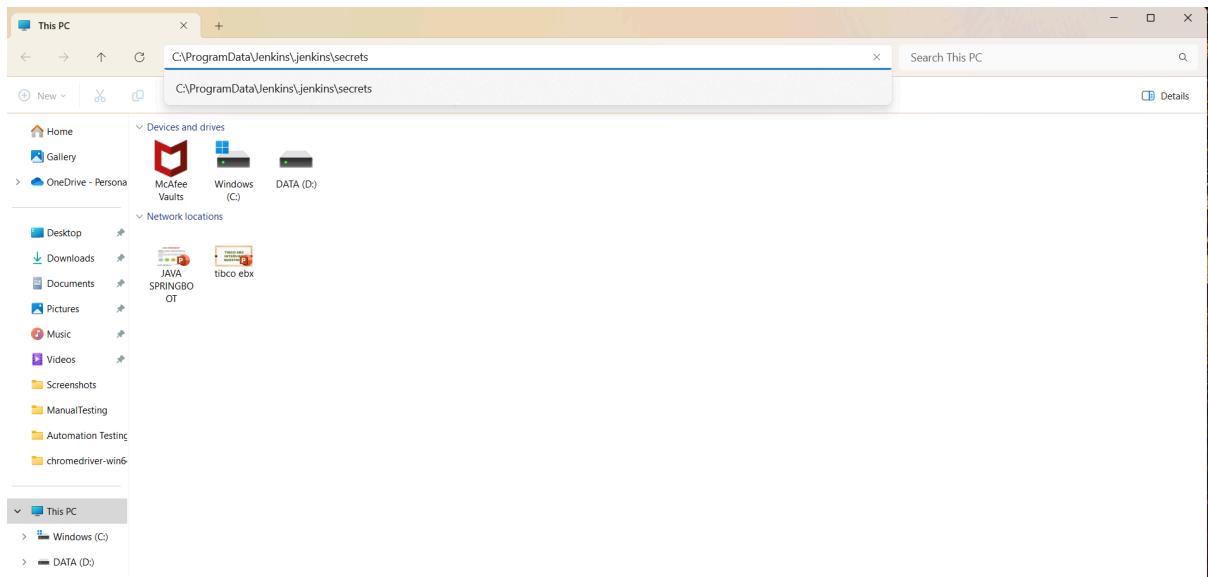
Press ENTER



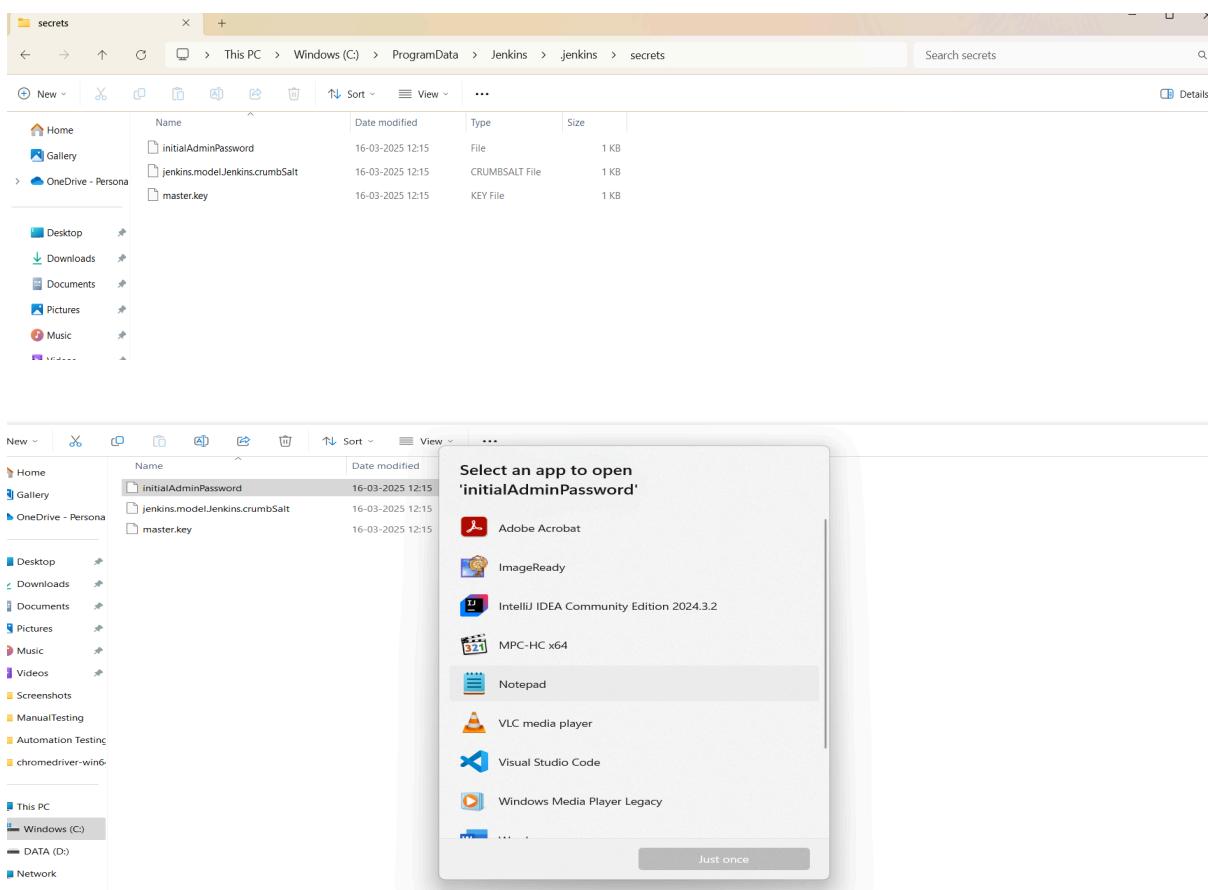
Step 2: Getting Started screen will appear. Here you will see a path, copy that path till SECRETS



Step 3: Go to my computer and paste the path on the top as shown in the image after pasting hit enter



Step 4: In the next screen open initialAdminPassword [Right click and open it in notepad]



Step 5: Copy whatever you got in the notepad and paste it in the Administrative password [In the getting started chrome page] as shown below

And click continue

Getting Started

## Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log ([not sure where to find it?](#)) and this file on the server:

`C:\ProgramData\Jenkins\.jenkins\secrets\initialAdminPassword`

Please copy the password from either location and paste it below.

Administrator password

.....

Continue

Step 6: In the next page “Customize Jenkins” simply click on install suggested plugins

Getting Started

## Customize Jenkins

Plugins extend Jenkins with additional features to support many different needs.

**Install suggested plugins**

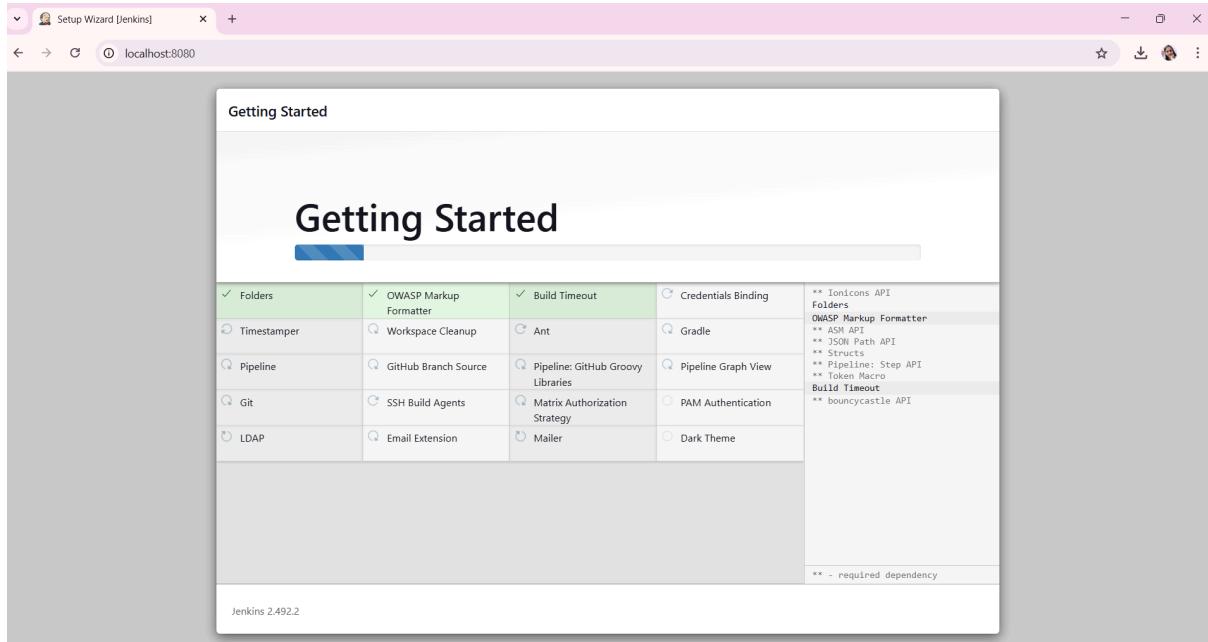
Install plugins the Jenkins community finds most useful.

**Select plugins to install**

Select and install plugins most suitable for your needs.

Jenkins 2.492.2

Now it will automatically install required plugins



After Successful installation of the plugins, you will be automatically redirected to the “CREATE FIRST ADMIN USER”

\*If your plugins fails by chance just click retry and after successful installation if not directed to next page directly then, click CONTINUE to go to “CREATE FIRST ADMIN USER”

Step 7: In you “CREATE FIRST ADMIN USER” give any username, password and other details and Click SAVE AND CONTINUE

A screenshot of the Jenkins "CREATE FIRST ADMIN USER" form. It has fields for Username (nikita), Password (redacted), Confirm password (redacted), Full name (Nikita Verma), and E-mail address (nikitaverma7006@gmail.com). At the bottom are two buttons: "Skip and continue as admin" and "Save and Continue".

Step 8: Next Page is the INSTANCE CONFIGURATION

Click on Save and Finish

Getting Started

## Instance Configuration

Jenkins URL:

http://localhost:8080/

The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the BUILD\_URL environment variable provided to build steps.

The proposed default value shown is **not saved yet** and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.

Jenkins 2.492.2

Not now

Save and Finish

Click on Start Using Jenkins

Getting Started

## Jenkins is ready!

Your Jenkins setup is complete.

[Start using Jenkins](#)

After successfully completing the above steps you will see your JENKINS Dashboard

The screenshot shows the Jenkins dashboard at [localhost:8080](http://localhost:8080). The top navigation bar includes links for 'Dashboard [Jenkins]', 'Build History', 'Manage Jenkins', and 'My Views'. The main content area features a 'Welcome to Jenkins!' message, a 'Start building your software project' section with links for 'Create a job', 'Set up a distributed build', 'Set up an agent', 'Configure a cloud', and 'Learn more about distributed builds'. On the left, there are sections for 'Build Queue' (empty) and 'Build Executor Status' (0/2). The bottom right corner shows 'REST API' and 'Jenkins 2.492.2'.

Step 9: Click On manage jenkins -----> Click On “SYSTEM”

The screenshot shows the Jenkins dashboard at [localhost:8080](http://localhost:8080). The 'Manage Jenkins' link in the top navigation bar is highlighted. The main content area displays the 'System Configuration' section, which includes tabs for 'System', 'Tools', 'Plugins', 'Nodes', 'Clouds', and 'Appearance'. Each tab has a brief description below it. The bottom left corner shows a 'Security' link.

On the System Page don't change anything, just Click SAVE

The screenshot shows the Jenkins System configuration page. At the top, there is a navigation bar with links for Dashboard, Manage Jenkins, and System. The main section is titled "System". It contains fields for "Home directory" (set to C:\ProgramData\Jenkins\jenkins), a "System Message" box (empty), "# of executors" (set to 2), and a "Labels" section (empty). At the bottom of the page are "Save" and "Apply" buttons.

## HOW TO ADD JDK, GIT AND MAVEN TO JENKINS

Open GITHUB -----> Open NIT Project

The screenshot shows a GitHub repository page for "NIT-9AM-Jan2025". The repository is private and was created by "TheQAGuy007" on 16-Mar-2025. It has 291 commits. The repository contains branches like main, .idea, ManualTesting, src, .gitignore, README.md, and pom.xml. The "About" section notes "No description, website, or topics provided." The "Releases" section says "No releases published" and "Create a new release". The "Contributors" section lists 23 contributors with small profile icons.

Step 1: From JENKINS dashboard Click Manage Jenkins -----> Click TOOLS

The screenshot shows the Jenkins Manage Jenkins interface. At the top, there are links for 'New Item', 'Build History', 'Manage Jenkins' (which is currently selected), and 'My Views'. Below this, there's a 'System Configuration' section with 'Build Queue' (0 builds in the queue) and 'Build Executor Status' (0/2). The 'Tools' section is highlighted, showing options like 'System' (Configure global settings and paths), 'Nodes' (Add, remove, control and monitor nodes), 'Tools' (Configure tools, their locations and automatic installers), 'Clouds' (Add, remove, and configure cloud instances), 'Plugins' (Add, remove, disable or enable plugins), and 'Appearance' (Configure the look and feel of Jenkins). A note at the top right says: 'Building on the built-in node can be a security issue. You should set up distributed builds. See the documentation.' There are also 'Set up agent', 'Set up cloud', and 'Dismiss' buttons. A search bar and a help icon are also present.

Step 2: In tools page Click ADD JDK

The screenshot shows the Jenkins Tools configuration page. It includes sections for 'Maven Configuration' (Default settings provider: 'Use default maven settings') and 'JDK installations' (with a 'Add JDK' button). Below these is a 'Git installations' section. At the bottom of the page are 'Save' and 'Apply' buttons.

## In NAME write- JAVA

JDK installations

Add JDK

Name

Required

JAVA\_HOME

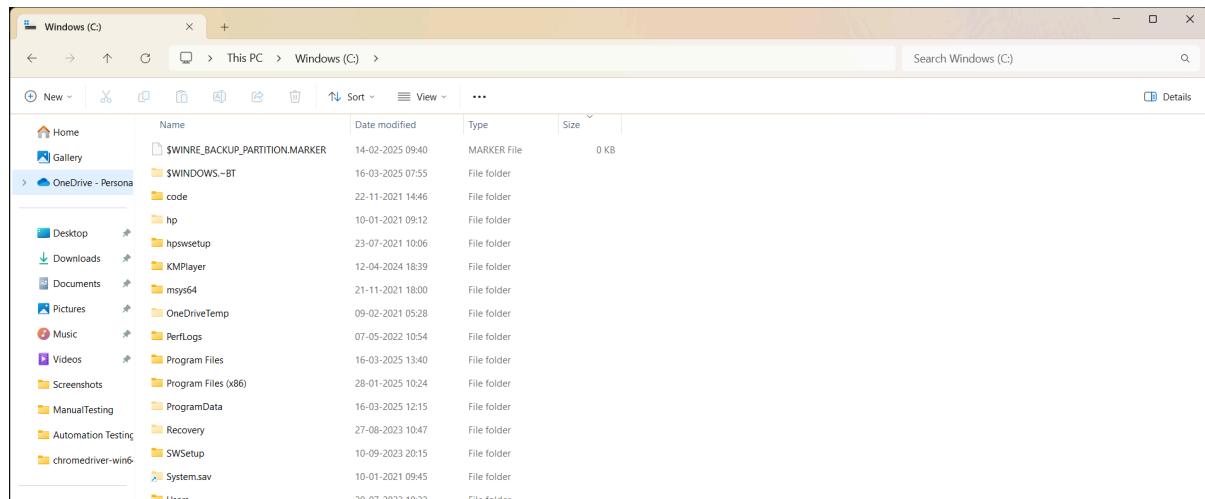
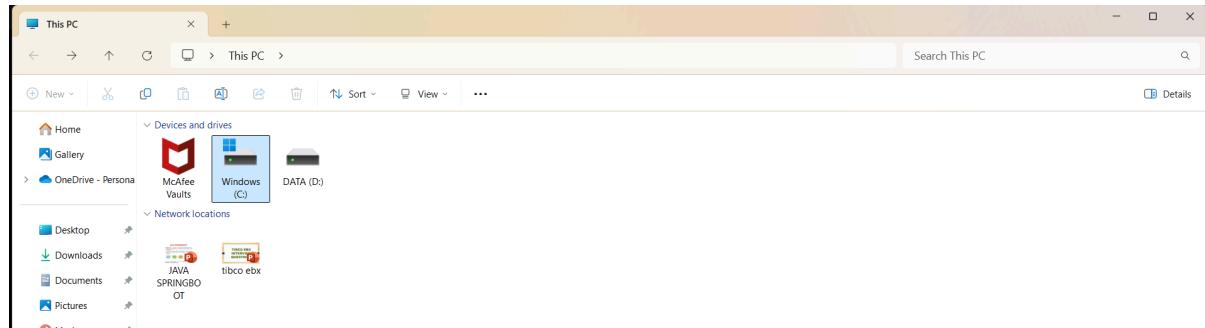
Install automatically ?

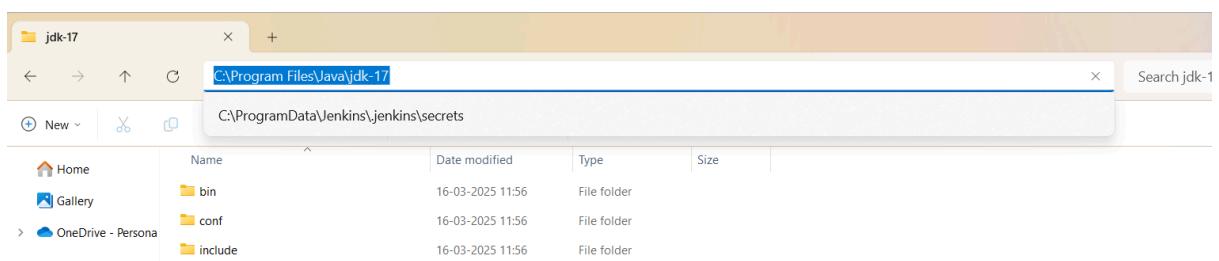
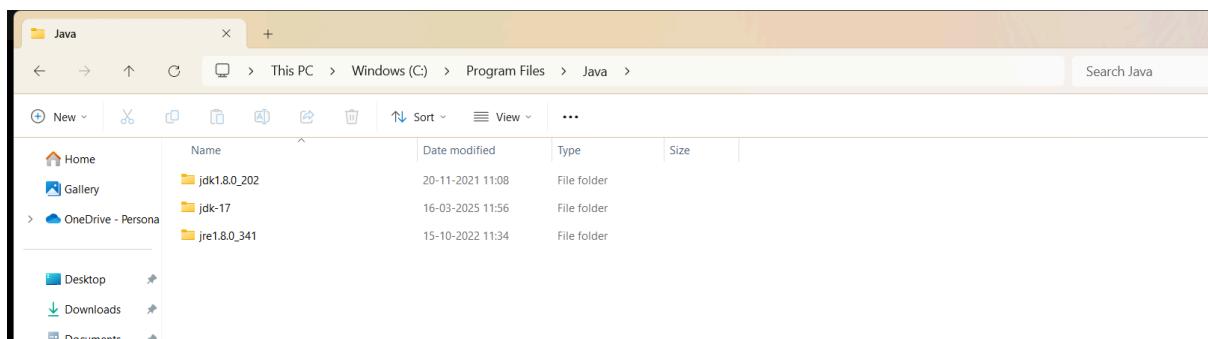
Add JDK

Git installations

Save Apply

Step 3: Go to MY COMPUTER -----> CDRIVE -----> Open Program Files ----->  
Open JAVA -----> Open JDK folder and then copy the path





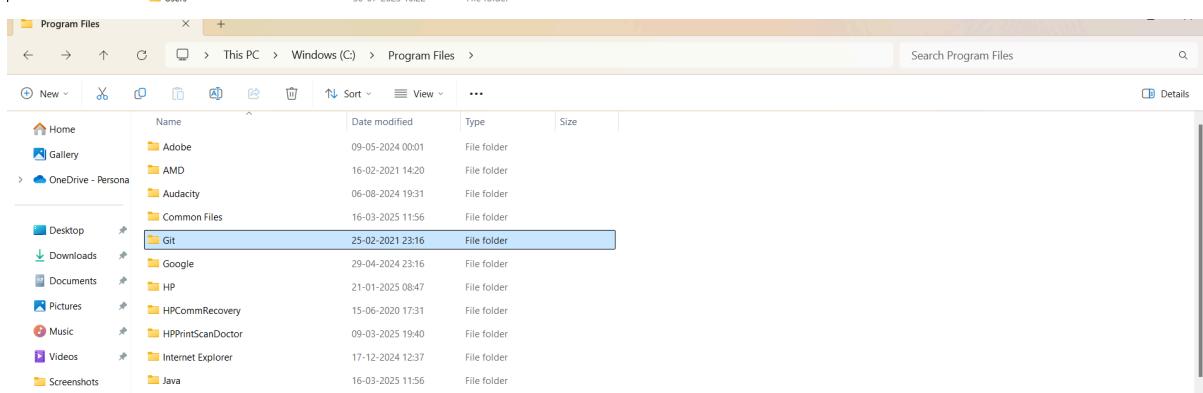
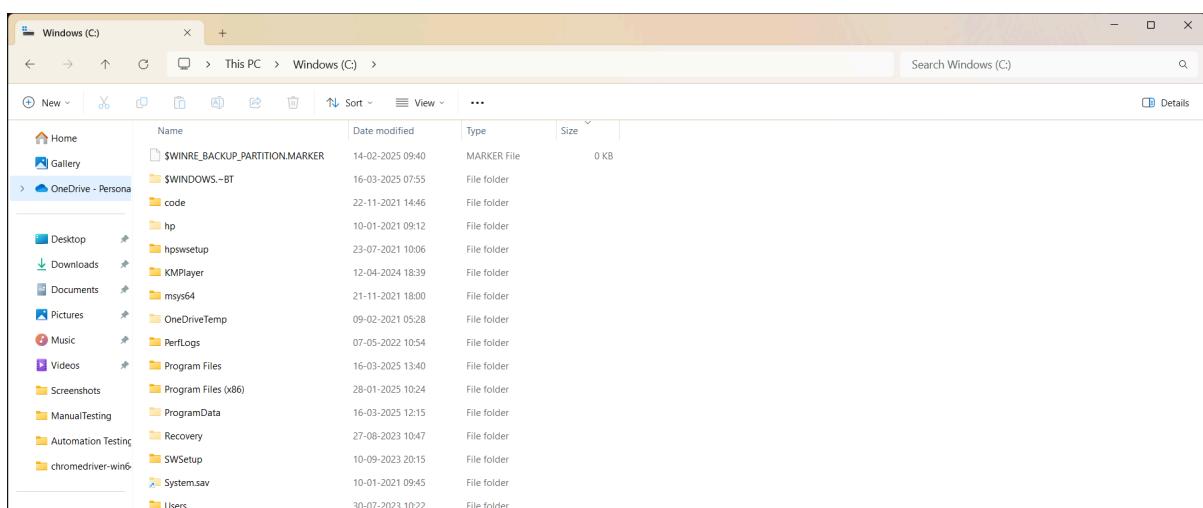
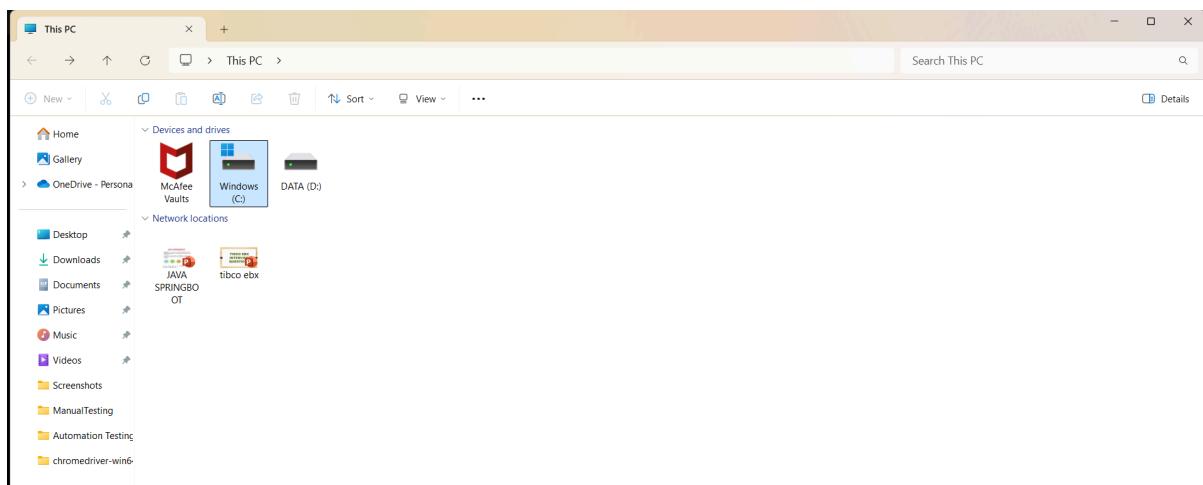
Step 4: Paste this path in JAVA\_HOME section in jenkins

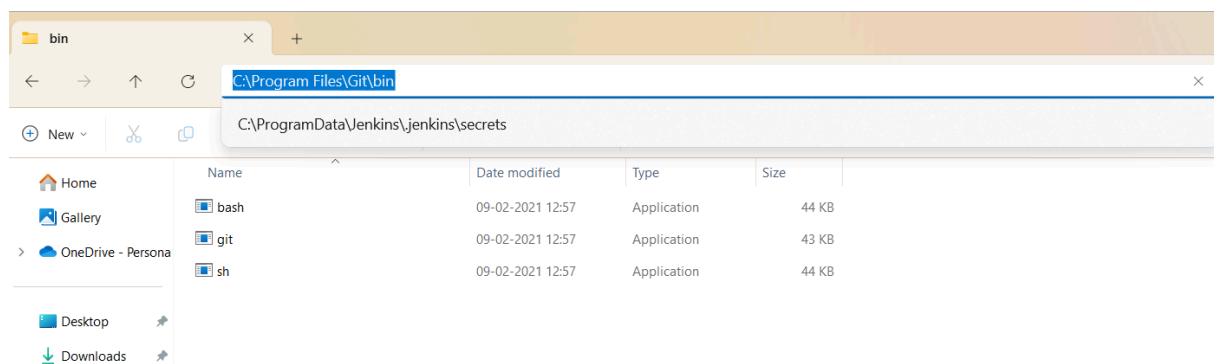
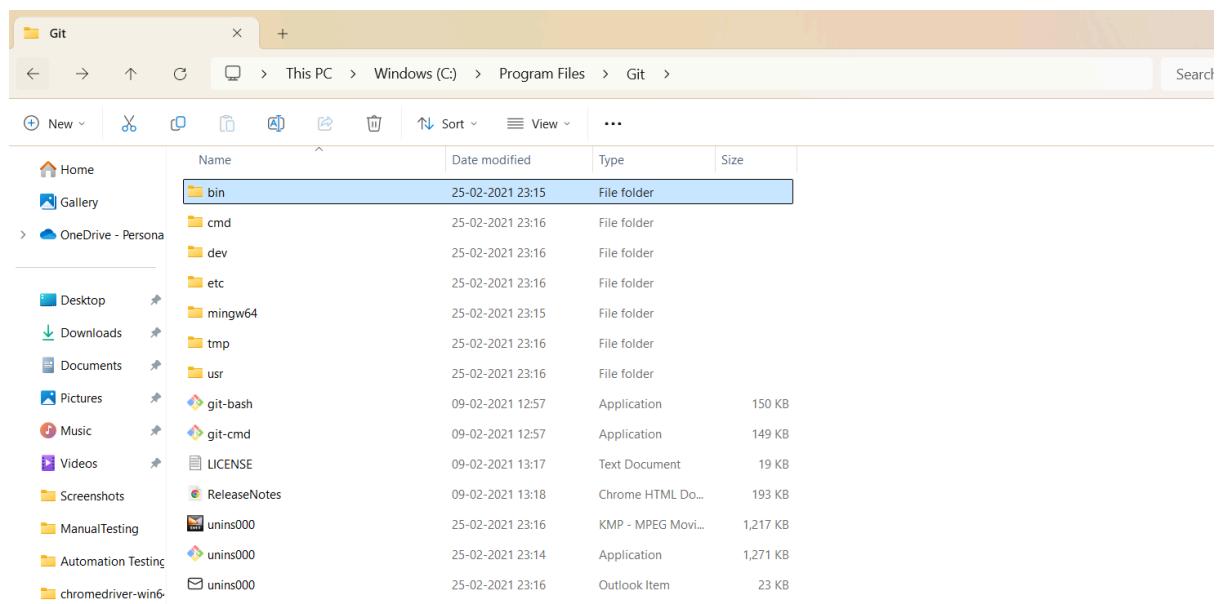
A screenshot of the Jenkins 'Manage Jenkins &gt; Tools' page under 'JDK installations'. A new 'JDK' entry is being added. The 'Name' field is empty and has a red asterisk indicating it is required. The 'JAVA\_HOME' field is also empty. There is an unchecked checkbox for 'Install automatically'. At the bottom, there are 'Save' and 'Apply' buttons.

Step 5: Scroll Down and you will see GIT INSTALLATION just below JDK INSTALLATION

**Now to get GIT path :**

Go to my computer -----> open Program files -----> Open Git folder -----> Open Bin -----> Copy Path

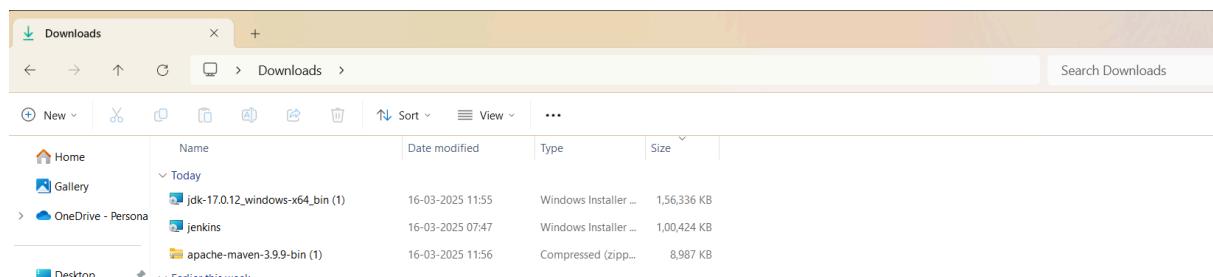




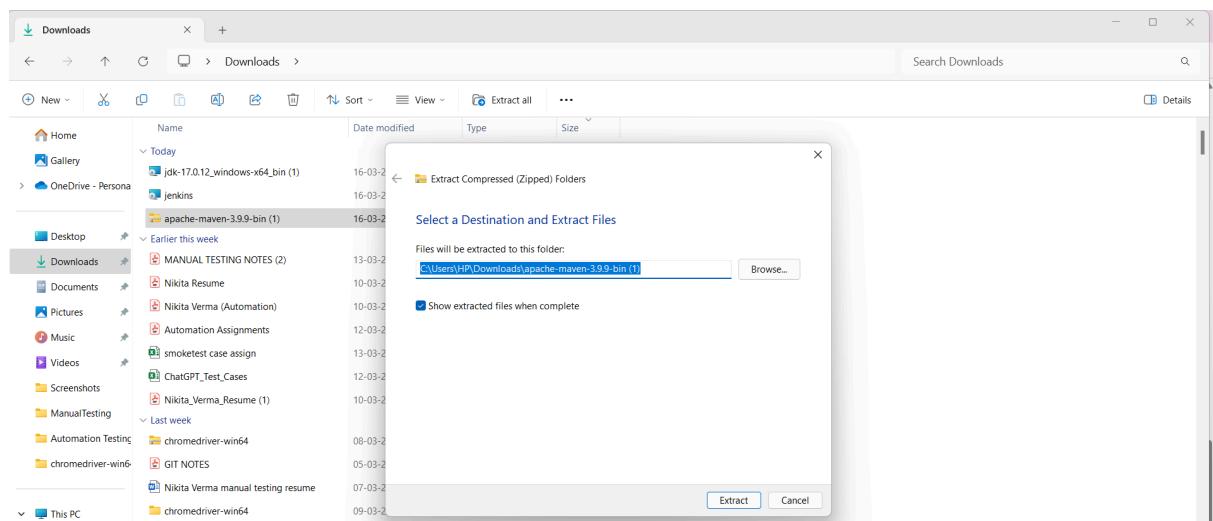
And paste it in the GIT path in jenkins

The screenshot shows the Jenkins 'Tools' configuration page under 'Manage Jenkins'. It displays settings for 'Git' installations. A single entry named 'Default' is shown, with its path to the Git executable set to 'C:\Program Files\Git\bin\git.exe'. There are buttons for 'Save' and 'Apply' at the bottom.

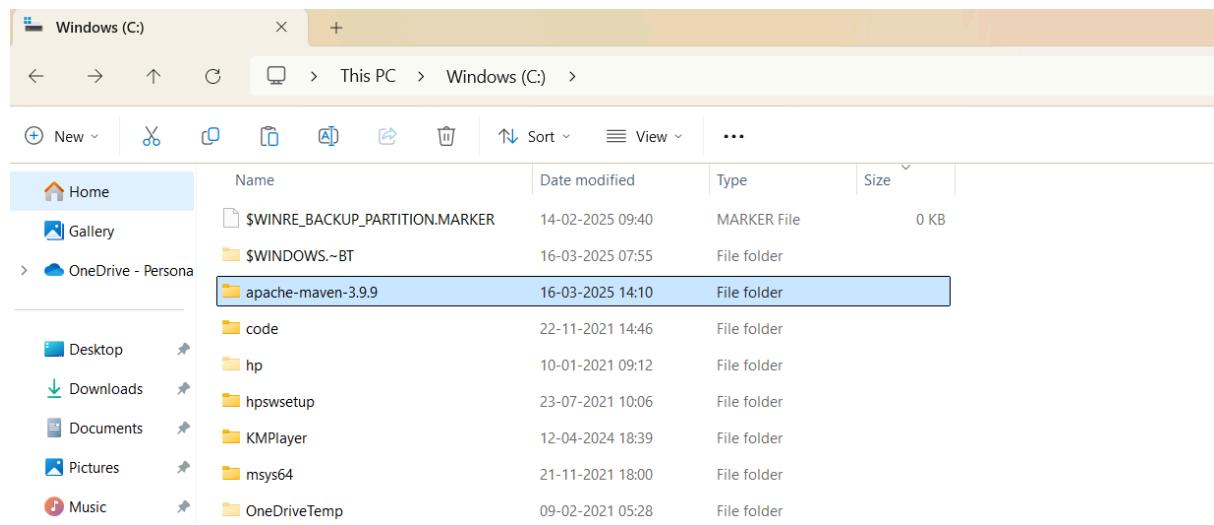
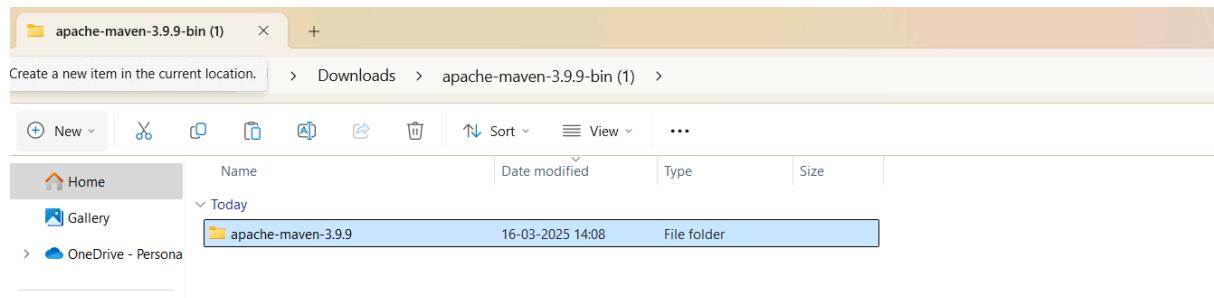
**Step 6 : Now Go to MAVEN that you have downloaded**  
[Maven download link : - <https://maven.apache.org/download.cgi> - Windows Users should download the "apache-maven-3.9.9-bin.zip" file.]



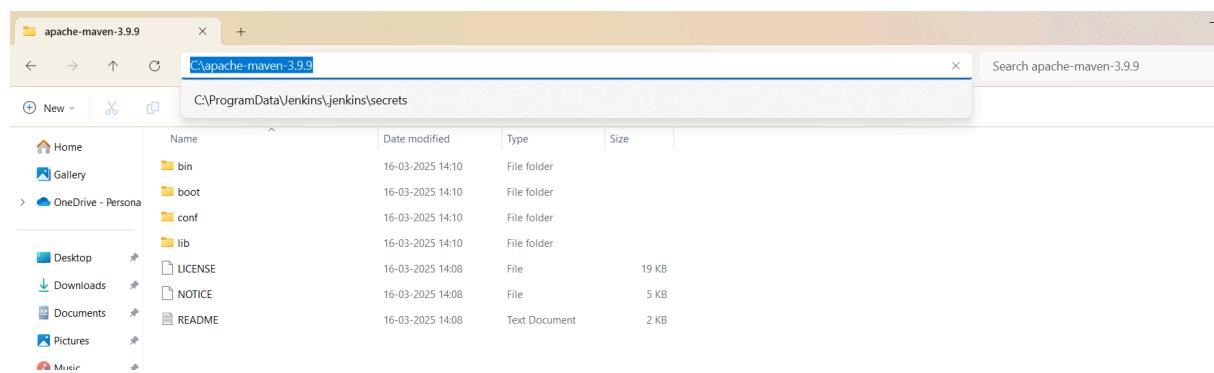
**Right Click and extract it**



Copy the extracted folder and paste it in the CDRIVE



Step 7: Go inside that folder and copy the path



Step 8: Go back to the same page where you have pasted git nd jdk path, in the same page scroll down you will see MAVEN INSTALLATION, Click ADD MAVEN

The screenshot shows the Jenkins management interface for tools. At the top, there are sections for 'Gradle installations' (with 'Add Gradle' button), 'Ant installations' (with 'Add Ant' button), and 'Maven installations' (with 'Add Maven' button). Below these is a 'Save' button in a blue box and an 'Apply' button.

In name section write maven  
And DESELECT INSTALL AUTOMATICALLY

This screenshot shows the 'Maven' configuration dialog. It has a 'Name' field (empty) with a red 'Required' error icon. Below it is a checked checkbox for 'Install automatically'. A nested panel titled 'Install from Apache' contains a 'Version' dropdown set to '3.9.9'. There is also an 'Add Installer' button. At the bottom are 'Add Maven', 'Save' (blue), and 'Apply' buttons.

This screenshot shows the 'Maven installations' section. It has an 'Add Maven' button. Below it is a 'Maven' configuration dialog with a 'Name' field containing 'maven'. Under 'MAVEN\_HOME', there is an empty text input field. A checkbox for 'Install automatically' is unchecked. At the bottom are 'Add Maven', 'Save' (blue), and 'Apply' buttons.

Then paste the copied path here in MAVEN\_HOME

CLICK SAVE AND YOU ARE DONE

The screenshot shows the 'Maven installations' section of the Jenkins 'Manage Jenkins' interface. A new Maven configuration is being added with the name 'maven'. The 'MAVEN\_HOME' field is set to 'C:\apache-maven-3.9.9'. The 'Install automatically' checkbox is unchecked. At the bottom, there are 'Save' and 'Apply' buttons.

## SECURITY CONFIGURATION

Step 1 : Now Again go back to Dashboard -----> Manage Jenkins -----> And Open Security

The screenshot shows the Jenkins 'Manage Jenkins' dashboard with the 'Security' section highlighted. It includes links for 'Security', 'Credentials', 'Users', and 'Nodes'. Other sections like 'System Configuration' and 'Status Information' are also visible.

Step 2 : Go to Git host Key Verification Configuration

The screenshot shows the Jenkins security configuration page. In the 'API Token' section, three options are listed: 'Generate a legacy API token for each newly created user (Not recommended)', 'Allow users to manually create a legacy API token (Not recommended)', and 'Enable API Token usage statistics' (which is checked). In the 'Git Host Key Verification Configuration' section, the 'Host Key Verification Strategy' dropdown is set to 'Known hosts file'. Below it, there's a checkbox for forcing the use of the sandbox globally. At the bottom are 'Save' and 'Apply' buttons.

Here select NO VERIFICATION from drop down

The screenshot shows the Jenkins security configuration page. In the 'Host Key Verification Strategy' dropdown, the 'No verification' option is highlighted with a blue background. A warning message below the dropdown states: '⚠️ This option is generally insecure. Host key will not be verified during SSH connection'.

And click SAVE

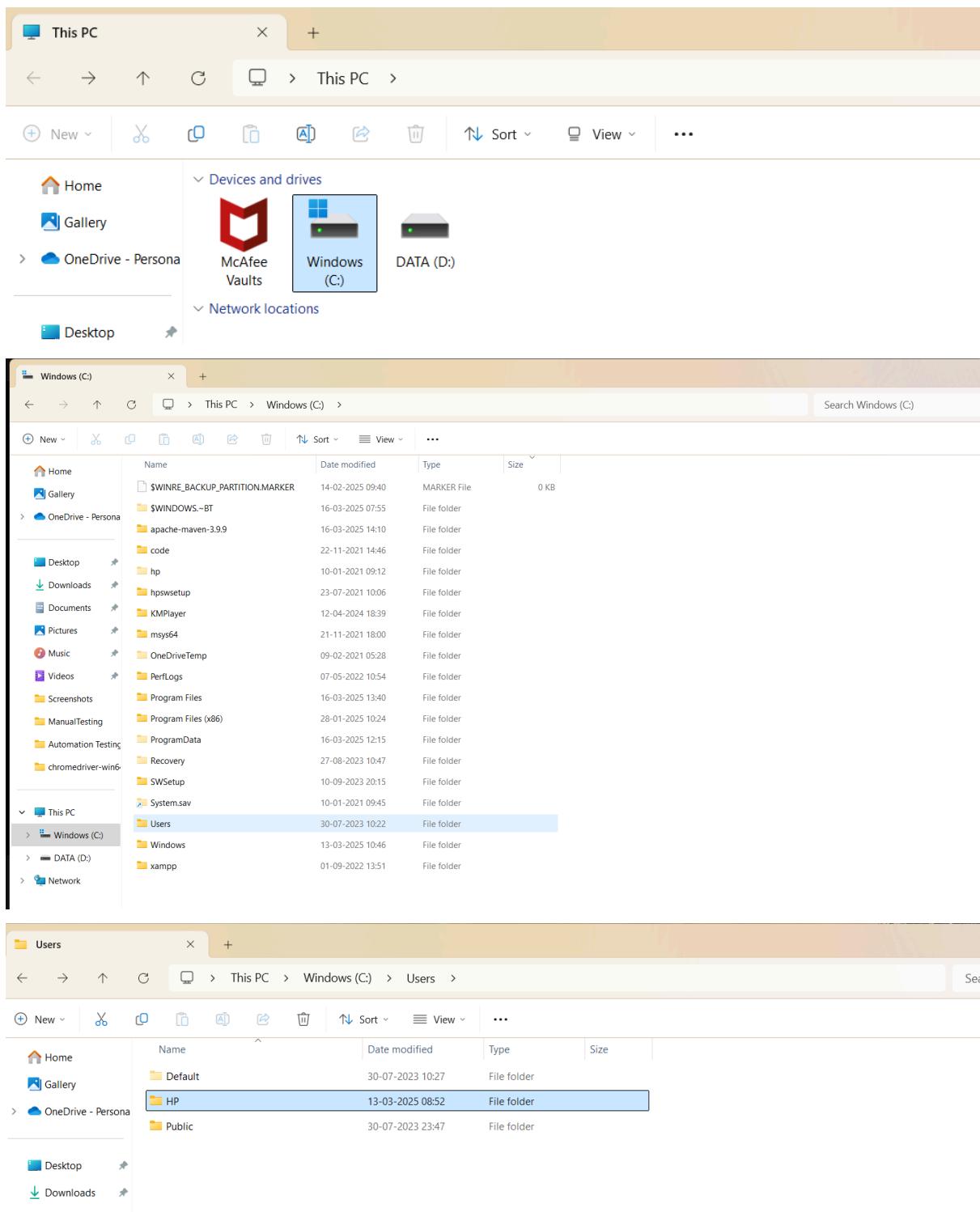
The screenshot shows the Jenkins security configuration page after saving the changes. The 'Host Key Verification Strategy' dropdown now shows 'No verification'. A warning message is displayed: '⚠️ This option is generally insecure. Host key will not be verified during SSH connection'. Below the dropdown, the 'Sandbox Configuration' section and its checkbox are visible. At the bottom are 'Save' and 'Apply' buttons.

By now all setup are completed and now, we have to CREATE JOBS

## INTEGRATING GITHUB WITH JENKINS

### SSH KEY GENERATION

Step 1: Go to Cdrive -----> Users -----> And select your Laptop User -----> Open it -----> Go to View-----> Click Show -----> Hidden Items



After unhiding the items, right click Randomly and select OPEN IN TERMINAL  
In terminal type **ssh-keygen**  
And keep on clicking till you get a triangular pattern

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

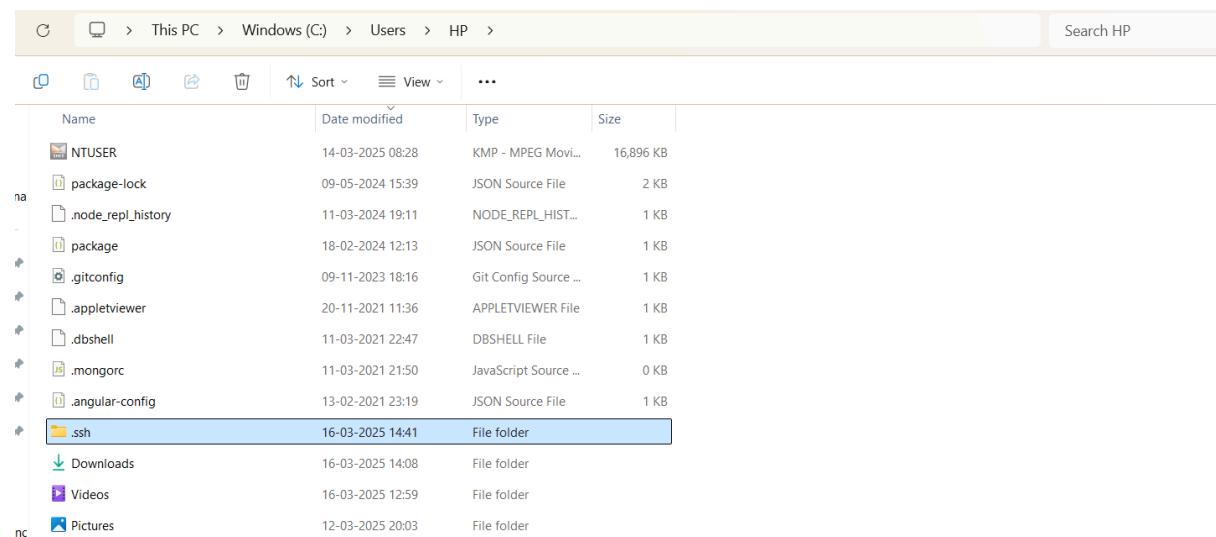
PS C:\Users\HP> ssh-keygen
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

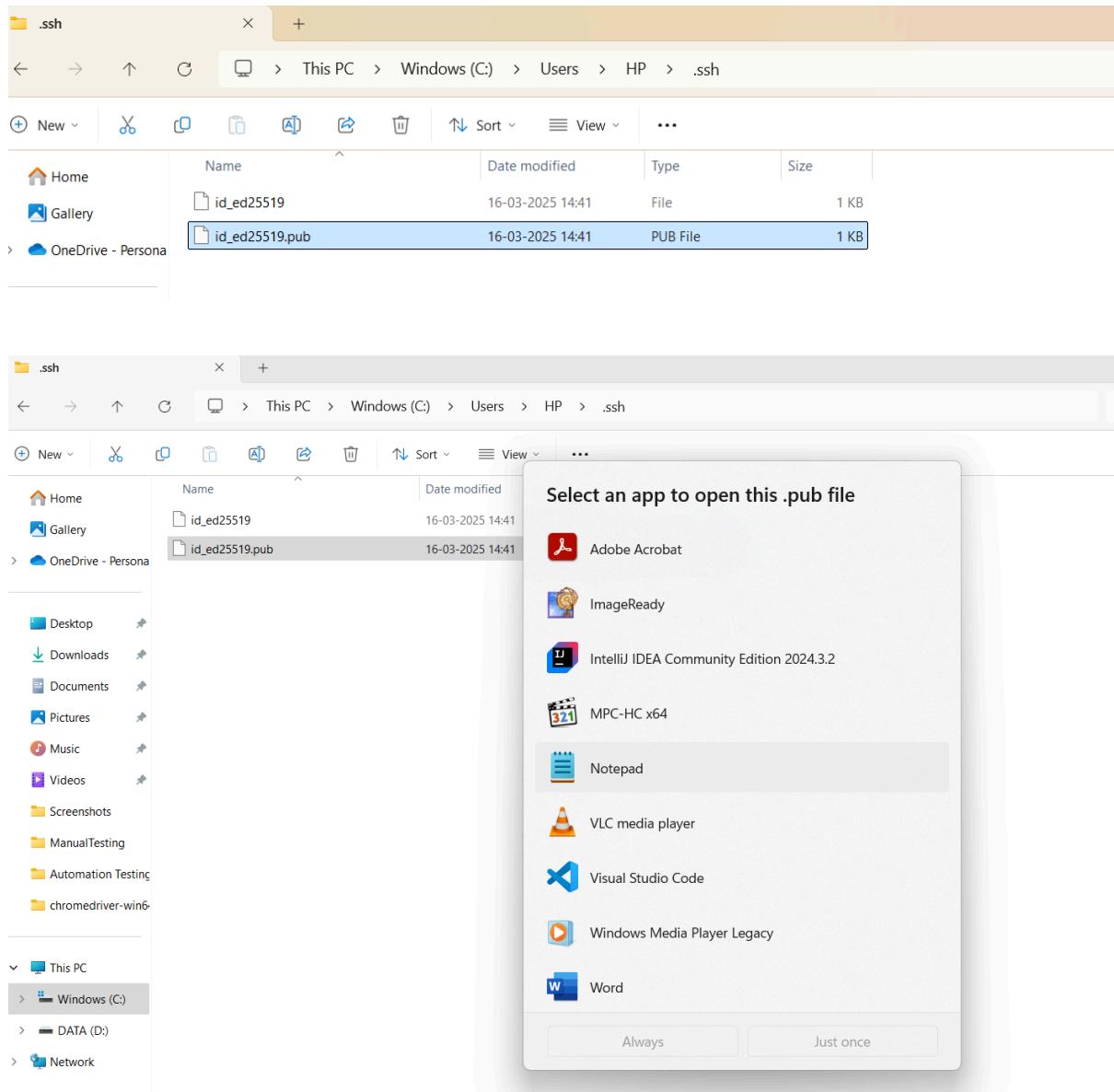
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\HP> ssh-keygen
Generating public/private ed25519 key pair.
Enter file in which to save the key (C:/Users/HP/.ssh/id_ed25519):
Created directory 'C:/Users/HP/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in C:/Users/HP/.ssh/id_ed25519
Your public key has been saved in C:/Users/HP/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:TFvdWgkpROCxjQ0FiiLrk1OqsD6EcF6y+NqrcY29UM hp@LAPTOP-QB0532VQ
The key's randomart image is:
+--[ED25519 256]--+
|.... =Bo. . .
|..o ..+o . +
|o. o o . . +
|o.+ . o o . +
|=o S . . +
|. O. E . o .
|.Oo.o .
|.*+. o
|=oo .
+---[SHA256]---+
PS C:\Users\HP> |
```

Step 2: Now again go to th User Folder and you will see .ssh folder



Step 3: Open the .ssh folder-----> Right Click on the .pub file and open it in notepad



Copy Everything that you got in your notepad

Step 4: Go to GitHub -----> On right side top click on your profile picture icon  
-----> Go to settings

The image shows two GitHub repository pages displayed side-by-side in a browser window. Both pages belong to the user 'TheQAGuy007' and are titled 'NIT-9AM-Jan2025'. The left page is a private repository, while the right page is a public repository.

**Left Repository (Private):**

- Code:** main, 35 Branches, 0 Tags
- Commits:** 291 Commits (by TheQAGuy007 on 16-Mar-2025)
- Files:** .idea, ManualTesting, src, .gitignore, README.md, pom.xml
- Activity:** 2 weeks ago (28-Feb-2025)
- Contributors:** 23

**Right Repository (Public):**

- Code:** main, 35 Branches, 0 Tags
- Commits:** 291 Commits (by TheQAGuy007 on 16-Mar-2025)
- Files:** .idea, ManualTesting, src, .gitignore, README.md, pom.xml
- Activity:** 3 hours ago (16-Mar-2025)
- Contributors:** 23

**User Profile (Right Side):**

- Profile:** nikitaverma7006, Nikita Verma
- Status:** Set status
- Profile:** Your profile
- Repositories:** Your repositories
- Copilot:** Your Copilot
- Projects:** Your projects
- Stars:** Your stars
- Gists:** Your gists
- Organizations:** Your organizations
- Enterprises:** Your enterprises
- Sponsors:** Your sponsors
- Enterprise:** Try Enterprise (Free)
- Feature:** Feature preview (New)
- Settings:** Settings
- GitHub:** GitHub Website, GitHub Docs
- Support:** GitHub Support
- Community:** GitHub Community
- Sign Out:** Sign out

The screenshot shows the GitHub profile settings page for a user named Nikita Verma. On the left, there is a sidebar with various settings categories: Public profile (selected), Account, Appearance, Accessibility, Notifications, Access (Billing and plans, Emails, Password and authentication, Sessions, SSH and GPG keys, Organizations, Enterprises, Moderation), and Code, planning, and automation. The main content area is titled "Public profile" and contains fields for Name (Nikita Verma), Public email (a dropdown menu with "Select a verified email to display"), Bio (a text input field with placeholder "Tell us a little bit about yourself"), Pronouns (a dropdown menu with "Don't specify"), and URL (an empty text input field). A search bar at the top right says "Type [ ] to se".

Step 5: On left hand side you will see SSH & GPG key , Click it

The screenshot shows the GitHub profile settings page for a user named Nikita Verma. The sidebar is identical to the previous screenshot. The main content area has three sections: "SSH keys" (with a "New SSH key" button), "GPG keys" (with a "New GPG key" button), and "Vigilant mode" (with a checkbox for "Flag unsigned commits as unverified"). The "SSH and GPG keys" category in the sidebar is also highlighted.

Step 6: Click On NEW SSH KEY -----> give some title -----> paste that thing that you copied (from the notepad by opening the pub file) in the key section -----> Click ADD SSH KEY

And it's done

The screenshot shows the GitHub Settings page for user Nikita Verma. The left sidebar is collapsed. The main area displays the 'SSH keys' section, which is currently empty. A 'New SSH key' button is visible. Below it is the 'GPG keys' section, also empty, with a 'New GPG key' button. At the bottom is the 'Vigilant mode' section, which includes an option to 'Flag unsigned commits as unverified'. The navigation bar at the top includes links for 'Go to your personal profile' and 'Settings'.

This screenshot shows the 'Add new SSH Key' form on the GitHub Settings page. The title field contains 'keypath' and the key type is set to 'Authentication Key'. The key itself is a long string of characters: 'ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIjQTFK0lbH8lYPyzpMWnnQnhI+9/wXSUhb6wPRnns5D hp@LAPTOP-QB0532VQ'. A large blue rectangular box highlights this key field. At the bottom is an 'Add SSH key' button. The left sidebar shows the collapsed navigation menu.

## CREDENTIALS Settings

Step1 : GO to the dashboard of JENKINS -----> Manage Jenkins -----> Go to Credentials

The screenshot shows the Jenkins Manage Jenkins dashboard. The left sidebar is collapsed. In the center, there is a grid of management links. The 'Credentials' link is highlighted with a red box. Other visible links include 'Nodes', 'Clouds', 'Appearance', 'Security', 'Users', 'Status Information', 'System Information', 'System Log', and 'About Jenkins'. The 'Credentials' link is located under the 'Status Information' section. The navigation bar at the top includes links for 'Dashboard' and 'Manage Jenkins'.

## Step 2 : Go to System -----> Global Credentials

The screenshot shows the Jenkins Global Credentials page. At the top, there's a navigation bar with the Jenkins logo, a search icon, a shield icon with a red dot, a user profile for 'Nikita Verma', and a 'log out' button. Below the navigation is a breadcrumb trail: Dashboard > Manage Jenkins > Credentials. The main title is 'Credentials'. Under 'Stores scoped to Jenkins', there's a table with one row: 'System' (Icon: User, Kind: (global)). At the bottom left is an 'Icon:' dropdown with options S, M, L.

The screenshot shows the Jenkins System Global Credentials page. At the top, there's a navigation bar with the Jenkins logo, a search icon, a shield icon with a red dot, a user profile for 'Nikita Verma', and a 'log out' button. Below the navigation is a breadcrumb trail: Dashboard > Manage Jenkins > Credentials > System >. The main title is 'System'. On the right, there's a blue button '+ Add domain'. Below it is a table with one row: 'Global credentials (unrestricted)' (Icon: Trophy, Kind: (global)). The description column states: 'Credentials that should be available irrespective of domain specification to requirements matching.' At the bottom left is an 'Icon:' dropdown with options S, M, L.

## Step 3: Click On ADD CREDENTIALS

The screenshot shows the Jenkins Global credentials (unrestricted) page. At the top, there's a navigation bar with the Jenkins logo, a search icon, a shield icon with a red dot, a user profile for 'Nikita Verma', and a 'log out' button. Below the navigation is a breadcrumb trail: Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted). The main title is 'Global credentials (unrestricted)'. On the right, there's a blue button '+ Add Credentials'. Below it is a note: 'Credentials that should be available irrespective of domain specification to requirements matching.' A table header row is shown with columns: ID, Name, Kind, and Description. A message in the table body says: 'This credential domain is empty. How about adding some credentials?' At the bottom left is an 'Icon:' dropdown with options S, M, L.

## Step 4 : Now in credentials Page

In KIND choose “SSH USER NAME WITH PRIVATE KEY”

In ID - give your Github user name

In Username - write the same githuhub user name

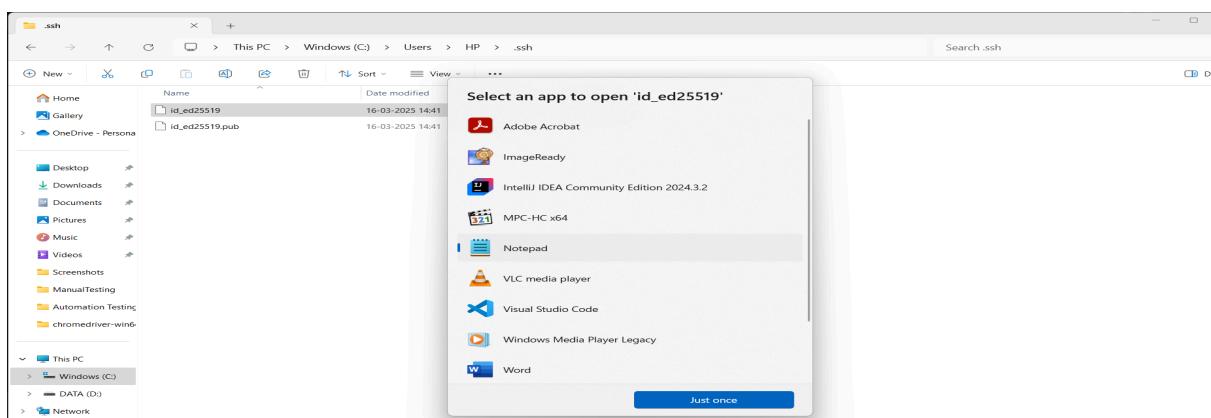
## In private key - Select Enter Directly

The screenshot shows the Jenkins 'New credentials' configuration page. The 'Kind' dropdown is set to 'Username with password'. The 'Scope' dropdown is set to 'Global (Jenkins, nodes, items, all child items, etc)'. The 'Username' field contains 'nikitaverma7006'. The 'Password' field is empty. The 'ID' field contains 'nikitaverma7006'. The 'Create' button is at the bottom.

Below this, the 'Private Key' section is shown. The 'Enter directly' radio button is selected. The 'Key' field has a placeholder 'No Stored Value' and an 'Add' button. The 'Passphrase' field is empty. The 'Create' button is at the bottom.

Step 5 : Now go back to the SSH Folder in Cdrive and copy the other file data and paste it in KEY SECTION

And Click CREATE



The screenshot shows the Jenkins 'Create New Credential' page. The 'Kind' is set to 'SSH Username with private key'. The 'Username' field contains 'nikitaverma7006'. The 'Private Key' section has 'Enter directly' selected, and the key value is a long string of characters starting with '-----BEGIN OPENSSH PRIVATE KEY-----'. A 'Passphrase' field is empty. At the bottom is a 'Create' button.

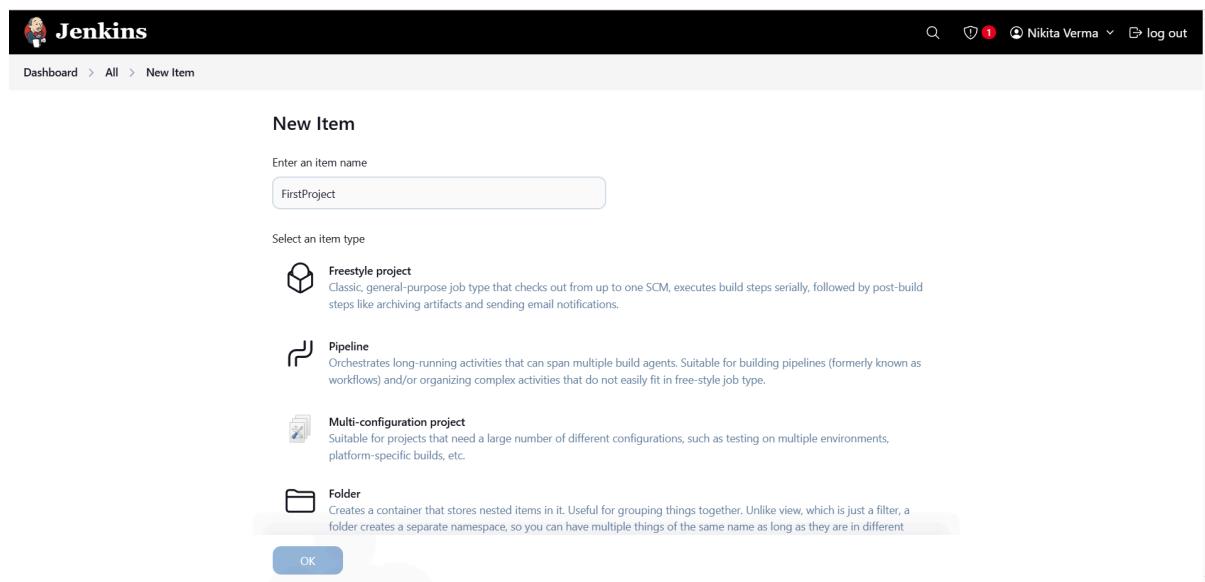
And your GLOBAL CREDENTIALS is CREATED

The screenshot shows the 'Global credentials (unrestricted)' page. It lists one credential: 'nikitaverma7006' (Kind: SSH Username with private key). There is a '+ Add Credentials' button at the top right.

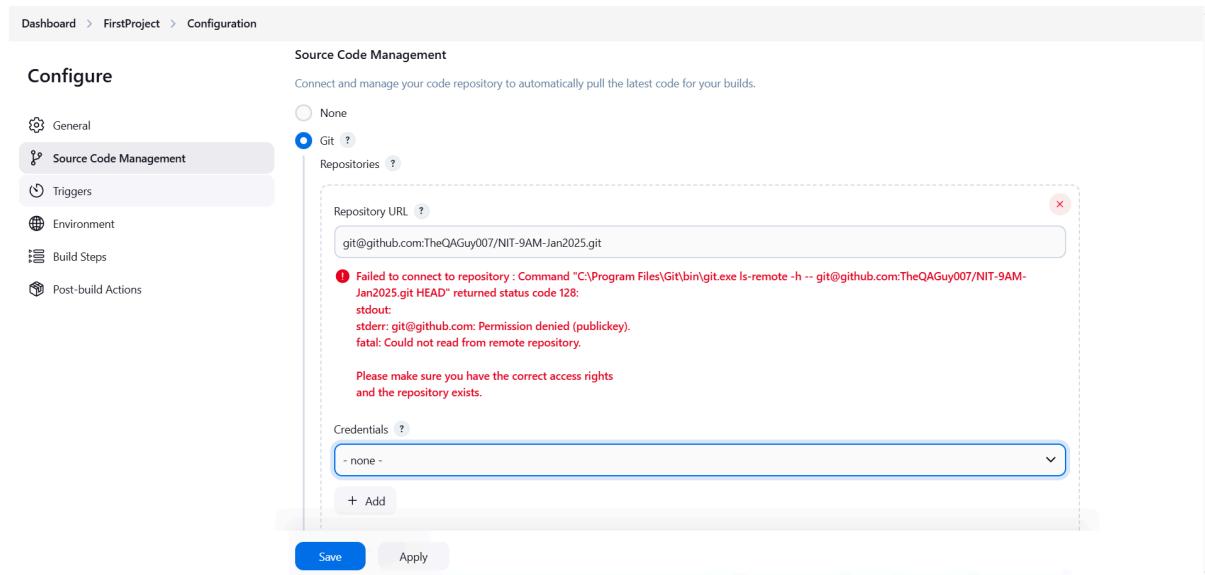
## CREATING JOBS IN JENKINS

Step 1 : GO to Jenkins DASHBOARD -----> Click New Items -----> Enter an item name -----> Click on FreeStyle Project -----> Click Ok

The screenshot shows the Jenkins dashboard. On the left, there are links for 'New Item', 'Build History', 'Manage Jenkins', and 'My Views'. The main area features a 'Welcome to Jenkins!' message and a 'Start building your software project' section with options like 'Create a job', 'Set up a distributed build', 'Set up an agent', 'Configure a cloud', and 'Learn more about distributed builds'.



Step 2: Go to your Job that you have created now , In the “Source Code Management” ----- Select Git



Step 3 : Go to Your Project in GitHub -----> Click on the Green colour Code Button present under NIT-9AM-JAN2025

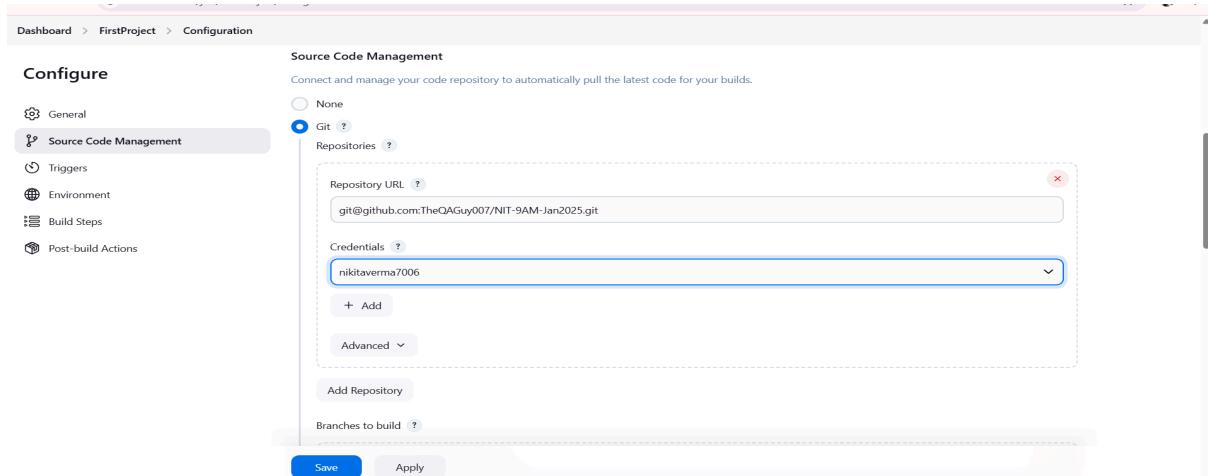
There Select “SSH” and then copy it and Paste it in The REPOSITORY URL of Jenkins

The screenshot shows a GitHub repository page for 'NIT-9AM-Jan2025'. The repository is private and has 35 branches and 0 tags. The main branch was last updated on 16-Mar-2025. The repository contains files like .idea, ManualTesting, src, .gitignore, README.md, and pom.xml. The README file is visible. The repository has 291 commits, with the most recent being a merge pull request from 'TheQAGuy007/Shantanu'. The repository has 2 watching and 0 forks. There are no releases or packages published.

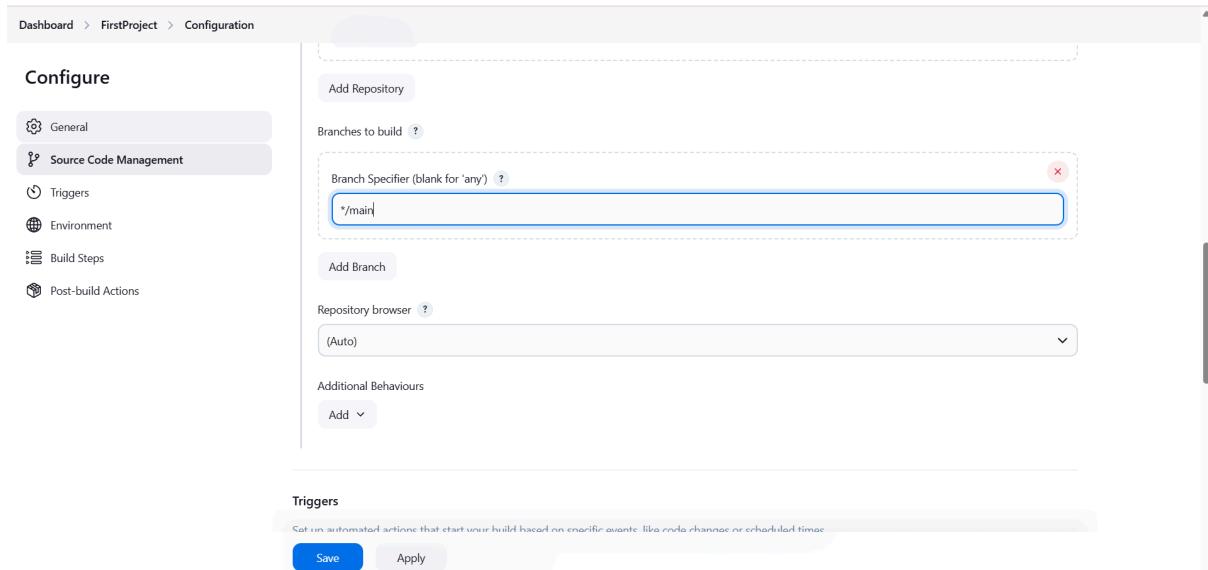
This screenshot shows the same GitHub repository page with a 'Clone' modal open. The modal provides options for cloning via Local, Codespaces, HTTPS, SSH, or GitHub CLI. It includes a field for a password-protected SSH key and links to open the repository in GitHub Desktop or download a ZIP file. The rest of the repository details and sidebar are visible.

The screenshot shows a Jenkins configuration screen for a project named 'FirstProject'. Under the 'Source Code Management' section, 'Git' is selected as the provider. The 'Repository URL' is set to 'git@github.com:TheQAGuy007/NIT-9AM-Jan2025.git'. A red error message indicates a failed connection attempt, showing the command output: 'Failed to connect to repository : Command "C:\Program Files\Git\bin\git.exe ls-remote -h --git@github.com:TheQAGuy007/NIT-9AM-Jan2025.git HEAD" returned status code 128: stdout: stderr: git@github.com: Permission denied (publickey). fatal: Could not read from remote repository.' Below this, a note says 'Please make sure you have the correct access rights and the repository exists.' Under 'Credentials', there is a dropdown menu currently set to 'none'. At the bottom are 'Save' and 'Apply' buttons.

Step 4 : Select Proper Credentials and the error will be gone (will take few seconds)



Step 5: Scroll Down and go to BRANCHES TO BUILD , write the branch name there in which you want to work



## Step 6: Click ADD BUILD STEPS -----> SELECT INVOLVE-LEVEL MAVEN TARGETS

The screenshot shows the Jenkins configuration page for a project named 'FirstProject'. The 'Build Steps' section is highlighted. A dropdown menu titled 'Add build step' is open, showing various options: 'Execute Windows batch command', 'Execute shell', 'Invoke Ant', 'Invoke Gradle script', 'Invoke top-level Maven targets' (which is selected and highlighted in blue), 'Run with timeout', and 'Set build status to "pending" on GitHub commit'. Below the dropdown, there is a note: 'Automate your build process with ordered tasks like code compilation, testing, and deployment.' and a 'Save' button.

## Step 7: In MAVEN VERSION select maven

The screenshot shows the Jenkins configuration page for a project named 'FirstProject'. The 'Build Steps' section is highlighted. A specific step named 'Invoke top-level Maven targets' is selected and expanded. It shows a 'Maven Version' dropdown set to '(Default)' and a 'Goals' input field. Below the step, there is a note: 'Define what happens after a build completes, like sending notifications, archiving artifacts, or triggering other jobs.' and a 'Save' button.

## Build Steps

Automate your build process with ordered tasks like code compilation, testing, and deployment.

The screenshot shows the Jenkins configuration interface for a job named 'FirstProject'. Under the 'Build Steps' section, there is a single step titled 'Invoke top-level Maven targets'. This step has a dropdown menu for 'Maven Version' with options '(Default)', '(Default)', and 'maven'. The 'maven' option is selected. Below the dropdown is an 'Advanced' button.

## Step 8: In GOALS type **test -DsuiteFile=packages**

The screenshot shows the Jenkins configuration interface for the 'FirstProject' job. The 'Build Steps' section contains a step titled 'Invoke top-level Maven targets'. The 'Goals' field for this step is set to 'test -DsuiteFile=packages'. Below the goals field is an 'Advanced' button. At the bottom of the configuration page, there are 'Save' and 'Apply' buttons.

Click **APPLY** and then Click **Save**

Now go to the Dashboard and you will see Your Created job there

The screenshot shows the Jenkins Dashboard. On the left, there is a sidebar with links for 'New Item', 'Build History', 'Manage Jenkins', and 'My Views'. Below the sidebar are two sections: 'Build Queue' (which says 'No builds in the queue.') and 'Build Executor Status' (which shows '0/2'). The main area of the dashboard lists the jobs. The 'FirstProject' job is listed with the following details: Status (S), Last Success (N/A), Last Failure (N/A), and Last Duration (N/A). There is also a 'More' button next to the job name. At the bottom of the dashboard, there is a footer with icons for 'S', 'M', and 'L'.

## HOW TO RUN A JOB

Step 1: Open Your Job that you want to run

The screenshot shows the Jenkins dashboard. On the left, there's a sidebar with links like 'New Item', 'Build History', 'Manage Jenkins', and 'My Views'. Below that is a 'Build Queue' section stating 'No builds in the queue.' At the bottom of the sidebar is a 'Build Executor Status' section showing '0/2'. The main area has tabs 'All' and '+' at the top. A table lists one job: 'FirstProject' with status 'S' (green), 'W' (yellow), 'Name' 'FirstProject', 'Last Success' 'N/A', 'Last Failure' 'N/A', and 'Last Duration' 'N/A'. There's also a 'Add description' button and a 'More' button.

Step 2; Click on build now present on left side

The screenshot shows the 'FirstProject' page. The top navigation bar includes 'Dashboard > FirstProject >'. On the left, a sidebar lists options: 'Status' (selected, indicated by a grey background), 'Changes', 'Workspace', 'Build Now' (with a green checkmark icon), 'Configure', 'Delete Project', and 'Rename'. The main content area is titled 'FirstProject' with a green checkmark icon. Below it is a 'Permalinks' section with a link to 'https:///job/FirstProject/lastSuccessfulBuild/'. To the right is a list of recent builds: 'Last build (#3), 2 days 8 hr ago', 'Last stable build (#3), 2 days 8 hr ago', 'Last successful build (#3), 2 days 8 hr ago', and 'Last completed build (#3), 2 days 8 hr ago'.

Step 3: Click on Build at the bottom and then in console output you can see the output

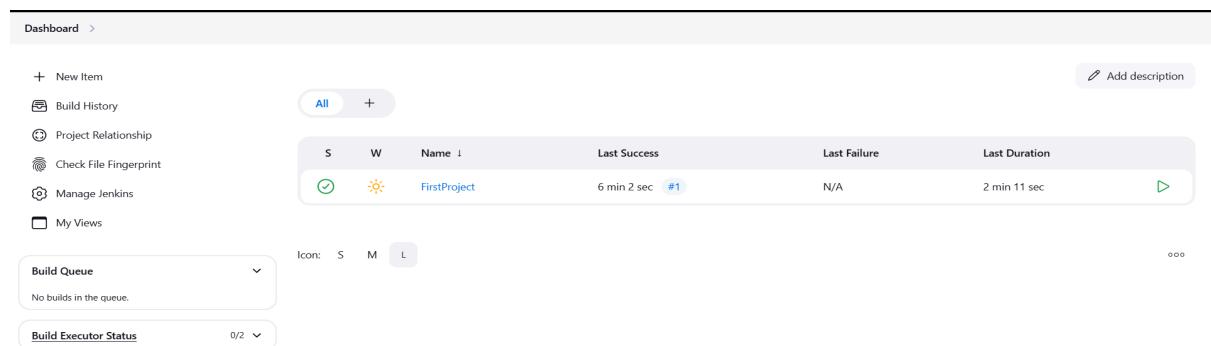
The screenshot shows the 'Console Output' for build #1 of 'FirstProject'. The top navigation bar includes 'Dashboard > FirstProject > #1 > Console Output'. The console output window displays the following log:

```
at 86 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/surefire/surefire-grouper/3.0.0-M7/surefire-grouper-3.0.0-
M7.jar (40 kB at 52 kB/s)
[INFO] -----
[INFO] T E S T S
[INFO] -----
[INFO] Running TestSuite
Sample TestNG
Void b
Void c
[INFO] Tests run: 2, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.78 s - in TestSuite
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 2, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 01:42 min
[INFO] Finished at: 2025-03-16T15:45:20+05:30
[INFO] -----
Finished: SUCCESS
```

At the bottom right of the console output window, it says 'REST API Jenkins 2.492.2'.

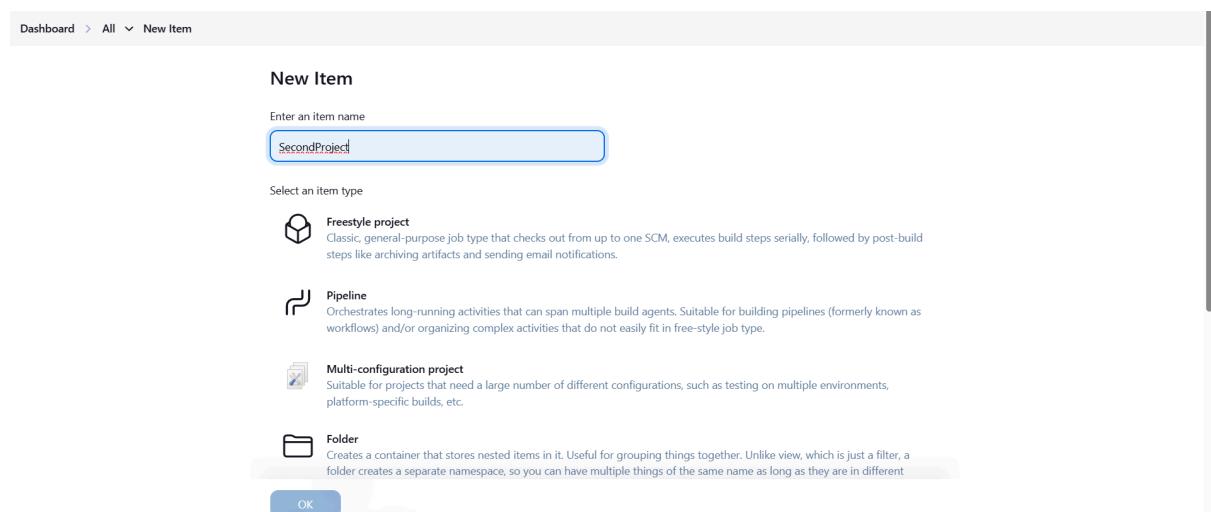
## CREATING ANOTHER JOB BY COPYING THE JOBS THAT ARE ALREADY PRESENT

Step 1: Go to DasHBoard -----> click NEW ITEM



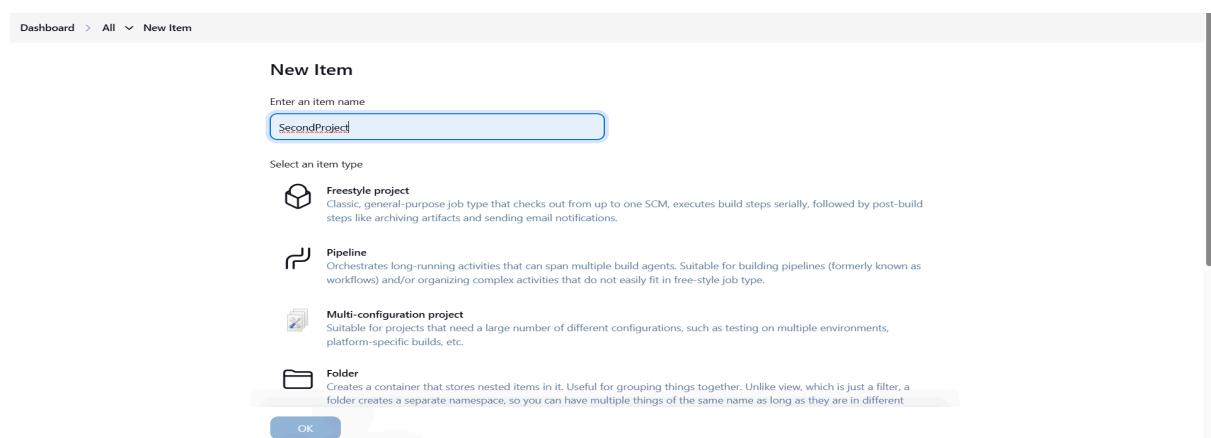
The screenshot shows the Jenkins Dashboard. On the left, there are links for 'New Item', 'Build History', 'Project Relationship', 'Check File Fingerprint', 'Manage Jenkins', and 'My Views'. In the center, there is a table with columns 'S', 'W', 'Name', 'Last Success', 'Last Failure', and 'Last Duration'. One row is visible for 'FirstProject'. Below the table, there are sections for 'Build Queue' (No builds in the queue) and 'Build Executor Status' (0/2). At the top right, there is a 'Add description' button.

Step 2: Write the name of the item (2nd job)



The screenshot shows the 'New Item' dialog. The title is 'New Item'. The 'Enter an item name' field contains 'SecondProject'. The 'Select an item type' section lists four options: 'Freestyle project' (selected), 'Pipeline', 'Multi-configuration project', and 'Folder'. Each option has a description below it. At the bottom is an 'OK' button.

Step 3 : Scroll Down & write the name of the JOB which you want to copy & click ok



The screenshot shows the 'New Item' dialog again. The 'Enter an item name' field contains 'SecondProject'. The 'Select an item type' section is identical to the previous screenshot, listing 'Freestyle project', 'Pipeline', 'Multi-configuration project', and 'Folder'. The 'OK' button is at the bottom.

 Creates a set of Pipeline projects according to detected branches in one SCM repository.

 Organization Folder  
Creates a set of multibranch project subfolders by scanning for repositories.

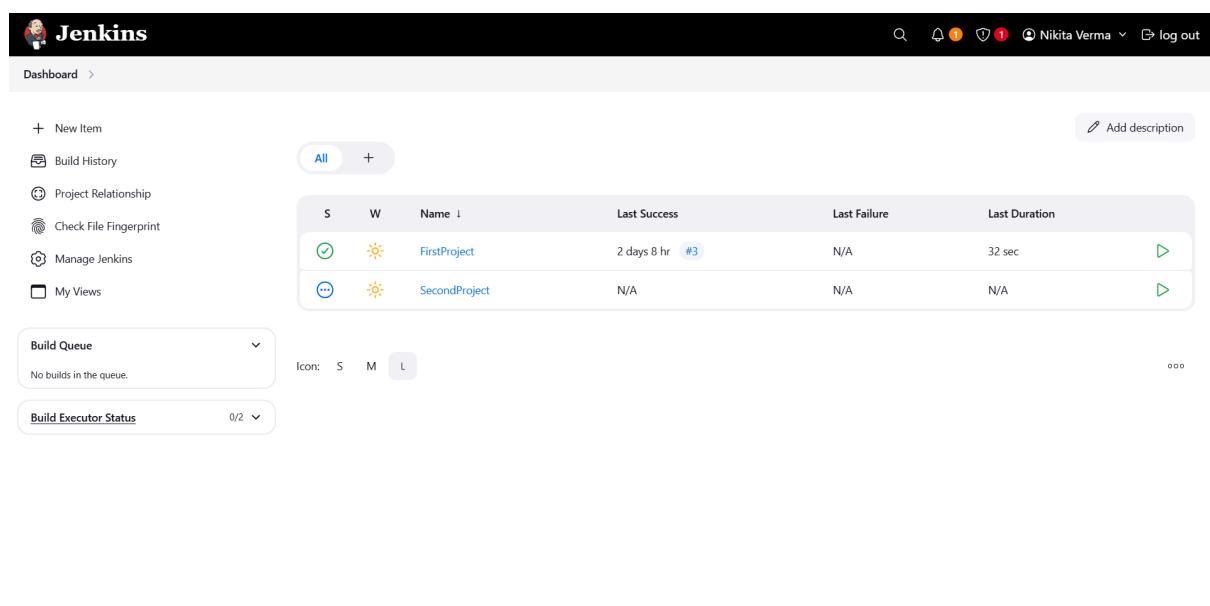
If you want to create a new item from other existing, you can use this option:

Copy from

FirstProject

OK

## Now you can see your created job in the Dashboard



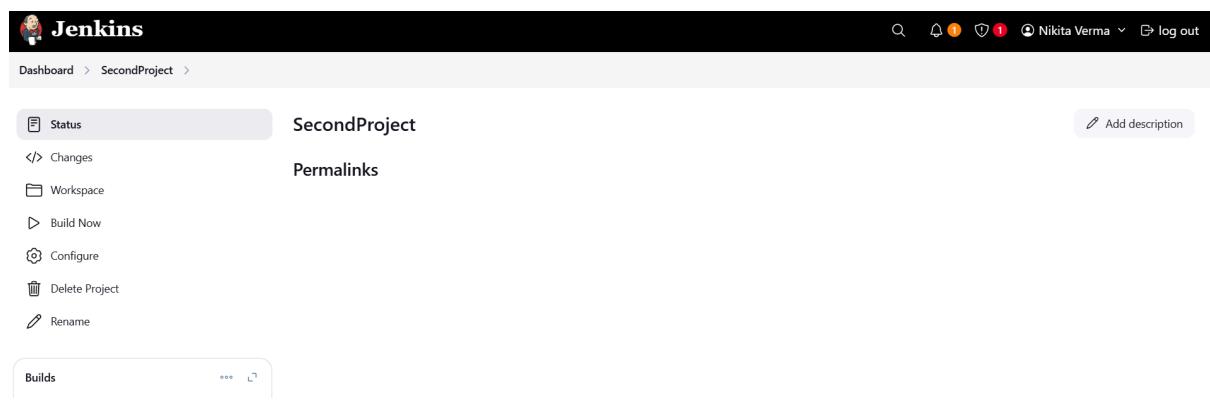
The screenshot shows the Jenkins dashboard with the following interface elements:

- Header:** Jenkins logo, search bar, notifications, user info (Nikita Verma), and log out link.
- Breadcrumbs:** Dashboard >
- Left sidebar:** New Item, Build History, Project Relationship, Check File Fingerprint, Manage Jenkins, My Views.
- Build Queue:** Shows "No builds in the queue." with icons for S, M, L.
- Build Executor Status:** Shows 0/2.
- Main Content:** A table listing projects:

S	W	Name	Last Success	Last Failure	Last Duration
		FirstProject	2 days 8 hr #3	N/A	32 sec
		SecondProject	N/A	N/A	N/A

## DELETING A JOB

Step 1: Open your Job by clicking it, on left side you can see Delete option, delete From There



The screenshot shows the SecondProject page with the following interface elements:

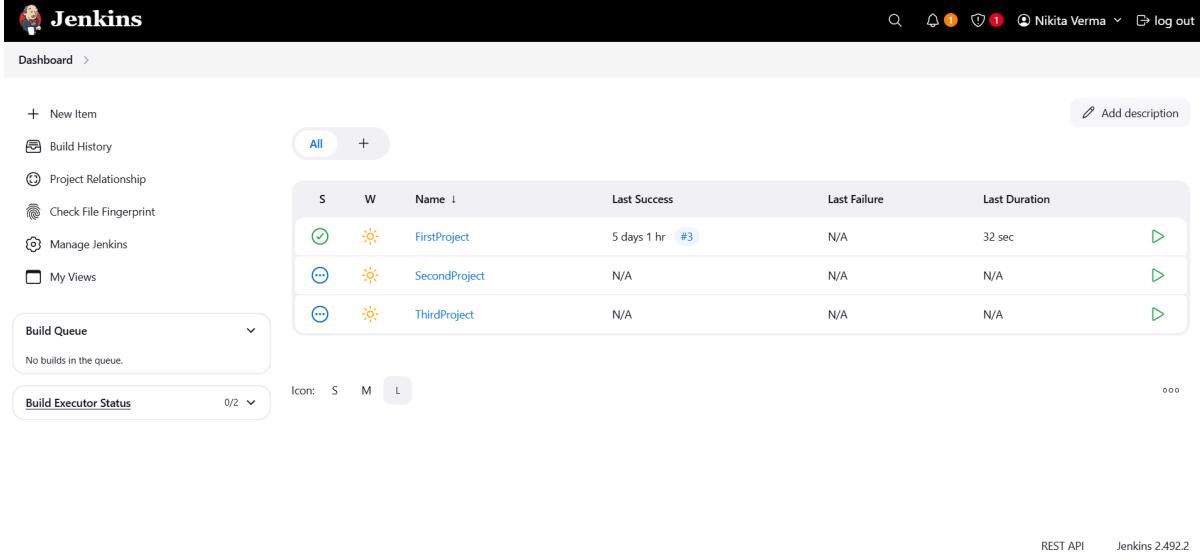
- Header:** Jenkins logo, search bar, notifications, user info (Nikita Verma), and log out link.
- Breadcrumbs:** Dashboard > SecondProject >
- Left sidebar:** Status (selected), Changes, Workspace, Build Now, Configure, Delete Project, Rename.
- Page Title:** SecondProject
- Permalinks:**
- Builds:** Shows 0/2.

## BASIC JENKINS PIPELINE

A basic Jenkins Pipeline is like : if one job is completed then the next job starts Automatically

For a basic jenkins pipeline to work there should be minimum 2 jobs

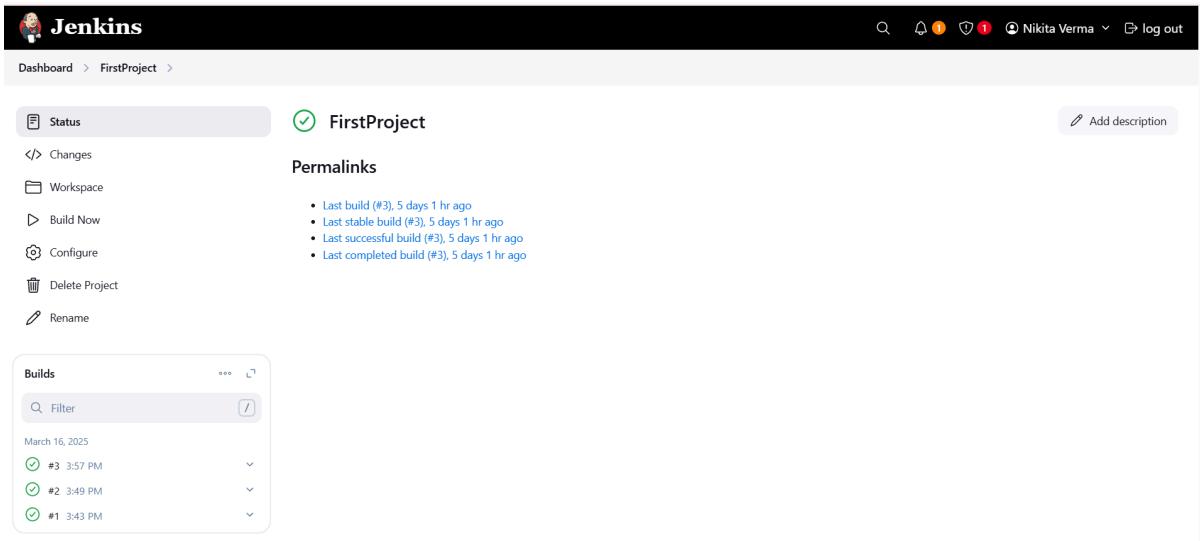
Step1 : In Dashboard, go to your 1st Project -----> Click Configure



The screenshot shows the Jenkins dashboard with the following interface elements:

- Header:** Jenkins logo, search bar, notifications, user info (Nikita Verma), and log out link.
- Left sidebar:** New Item, Build History, Project Relationship, Check File Fingerprint, Manage Jenkins, My Views, Build Queue (empty), and Build Executor Status (0/2).
- Center content:** A table listing three projects:

S	W	Name	Last Success	Last Failure	Last Duration
Green checkmark	Sun icon	FirstProject	5 days 1 hr #3	N/A	32 sec
Ellipsis	Sun icon	SecondProject	N/A	N/A	N/A
Ellipsis	Sun icon	ThirdProject	N/A	N/A	N/A
- Bottom right:** REST API and Jenkins 2.492.2 links.



The screenshot shows the Jenkins project page for "FirstProject" with the following interface elements:

- Header:** Jenkins logo, search bar, notifications, user info (Nikita Verma), and log out link.
- Left sidebar:** Status (selected), Changes, Workspace, Build Now, Configure, Delete Project, and Rename.
- Center content:** Project name "FirstProject" with a green checkmark icon, Permalinks (list of recent builds), and a Builds section showing:
  - Builds table with columns: #, Status, and Date.
  - Recent builds: #3 (3:57 PM), #2 (3:49 PM), and #1 (3:43 PM).

Step 2 : Scroll Down and go to Post Build Actions and Click On Add Build Actions

**Configure**

- General
- Source Code Management
- Triggers
- Environment
- Build Steps**
- Post-build Actions

**Invoke top-level Maven targets**

Maven Version: maven  
Goals: test -DsuiteFile=packages

**Post-build Actions**

Add post-build action

Save Apply

REST API Jenkins 2.492.2

### Step 3 : Select BUILD other Projects

**Configure**

- General
- Source Code Management
- Triggers
- Environment
- Build Steps
- Post-build Actions

**Invoke top-level Maven targets**

Maven Version: Filter

**Post-build Actions**

Aggregate downstream test results  
Archive the artifacts  
**Build other projects**  
Publish JUnit test result report  
Record fingerprints of files to track usage  
Git Publisher  
E-mail Notification  
Editable Email Notification  
Set GitHub commit status (universal)  
Set build status on GitHub commit (deprecated)  
Delete workspace when build is done

Add post-build action

Save Apply

REST API Jenkins 2.492.2

Step 4 : Write the name of your Job in “ project to Build” section that you want to get executed next. Then Click SAVE

**Configure**

- General
- Source Code Management
- Triggers
- Environment
- Build Steps
- Post-build Actions

**Post-build Actions**

Define what happens after a build completes, like sending notifications, archiving artifacts, or triggering other jobs.

**Build other projects**

Projects to build

No project specified

Trigger only if build is stable  
 Trigger even if the build is unstable  
 Trigger even if the build fails

Add post-build action

Save Apply

REST API Jenkins 2.492.2

Step 5 : Now go to 1st Job and Click Build Now

The screenshot shows the Jenkins interface for the 'FirstProject' job. The top navigation bar includes links for 'Dashboard', 'FirstProject', 'Status' (highlighted), 'Changes', 'Workspace', 'Build Now', 'Configure', 'Delete Project', and 'Rename'. The main content area displays the 'FirstProject' status with a green checkmark icon. Below it is a 'Permalinks' section with a bulleted list of recent builds. A 'Builds' card shows a table with three rows of build information, each with a green circle icon and a build number (#1, #2, #3) followed by a timestamp (3:43 PM, 3:49 PM, 3:57 PM). At the bottom right of the card, there are 'REST API' and 'Jenkins 2.492.2' links.

Step 6 : Your Build will start, click on that build number and in console you will see it

The screenshot shows the Jenkins interface for the 'FirstProject' job. The top navigation bar includes links for 'Dashboard', 'FirstProject', 'Status' (highlighted), 'Changes', 'Workspace', 'Build Now', 'Configure', 'Delete Project', and 'Rename'. The main content area displays the 'FirstProject' status with a green checkmark icon. Below it is a 'Permalinks' section with a bulleted list of recent builds. A 'Builds' card shows a table with four rows of build information. The fourth row, labeled '#4 5:54 PM', has a green circle icon and is highlighted with a red border, indicating it is the currently selected build. The URL 'localhost:8080/job/FirstProject/build?delay=0sec' is visible at the bottom of the card. The bottom half of the screen shows the Jenkins log terminal with the following output:

```
> C:\Program Files\Git\bin\git.exe config core.sparsecheckout # timeout=10
> C:\Program Files\Git\bin\git.exe checkout -f cf08d451ab403fd4e8ed65b65e494bd69f1713bb # timeout=10
Commit message: "Merge pull request #61 from TheQAGuy007/Shantanu"
> C:\Program Files\Git\bin\git.exe rev-list --no-walk cf08d451ab403fd4e8ed65b65e494bd69f1713bb # timeout=10
[test] $ cmd.exe /C "C:\apache-maven-3.9.9\bin\mvn.cmd test -suiteFile=packages && exit %ERRORLEVEL%"
[INFO] Scanning for projects...
[INFO]
[INFO] -----< org.example:NIT-9AM-Jan2025 >-----
[INFO] Building NIT-9AM-Jan2025 1.0-SNAPSHOT
[INFO]   from pom.xml
[INFO] -----[ jar ]-----
[INFO]
[INFO] --- resources:3.3.1:resources (default-resources) @ NIT-9AM-Jan2025 ---
[INFO] skip non existing resourceDirectory C:\ProgramData\Jenkins\.jenkins\workspace\test\src\main\resources
[INFO]
[INFO] --- compiler:3.10.1:compile (default-compile) @ NIT-9AM-Jan2025 ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 173 source files to C:\ProgramData\Jenkins\.jenkins\workspace\test\target\classes
```

Once this job completes its execution immediately the next Build that you have added will start executing automatically.

In the console you can also see the message on the top “ Started by upstream Project FirstProject ...“

The screenshot shows the Jenkins interface with the "Console Output" tab selected. The output window displays the following log entries:

```
Started by upstream project "FirstProject" build number 7
originally caused by:
Started by user Nikita Verma
Running as SYSTEM
Building in workspace C:\ProgramData\Jenkins\.jenkins\workspace\SecondProject
The recommended git tool is: NONE
```

This is how a basic pipeline works.

## Jenkins: Cron Syntax:

Used for Scheduling Jobs in Jenkins.

The order of the Syntax is as follows:

**Minute Hour Day(Month) Month Day(Week)**

To Schedule using Cron Syntax do the following:

- Open the Job -> Configure -> Build Triggers -> Build Periodically -> Enter the appropriate Cron Syntax

Example Cron Syntax below:

0 10 \* \* \* - Every day at 10 AM.

0 10 30 \* \* - Every 30th of the month at 10 AM.

0 0 \* \* 0 - 12 AM Every Sunday of the week.

