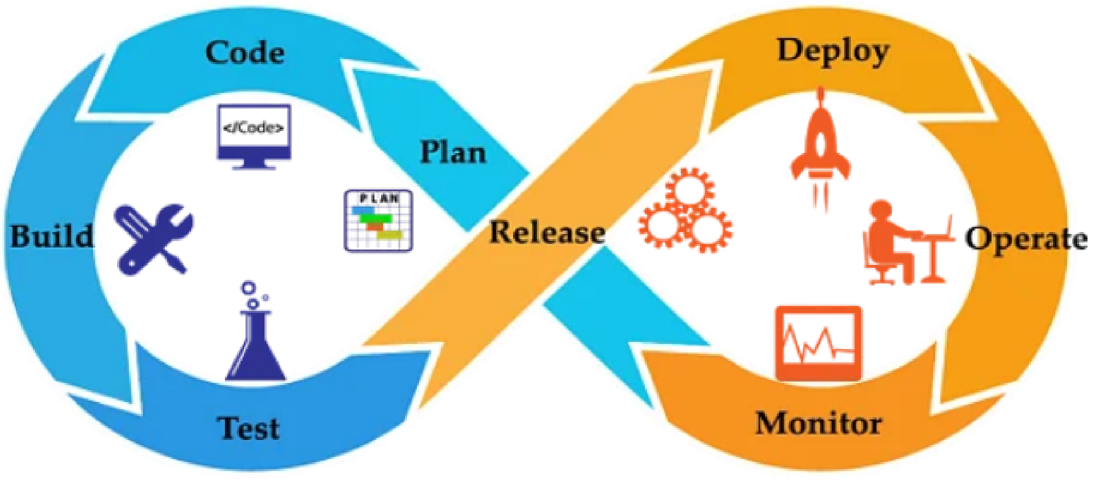
Experiment 05 - Jenkins Setup

Roll No.	70
Name	MAYURI SHRIDATTA YERANDE
Class	D15-B
Subject	DevOps Lab
LO Mapped	LO1: To understand the fundamentals of DevOps engineering and be fully proficient with DevOps terminologies, concepts, benefits, and deployment options to meet your business requirements LO3: To understand the importance of Jenkins to Build and deploy Software Applications on server environment

Aim: To understand Continuous Integration, install and configure Jenkins with Maven/Ant/Gradle to setup a build Job.

Introduction:

Continuous integration refers to the build and unit testing stages of the software release process. Every revision that is committed triggers an automated build and test. With continuous delivery, code changes are automatically built, tested, and prepared for a release to production.
t refers to the process of automating the integration of code changes coming from several sources. The process comprises several automation tools that emphasize on the code's correctness before Integration.



Continuous Integration enables better transparency and farsightedness in the process of software development and delivery. It not only benefits the developers but all the segments of that company. These benefits make sure that the organization can make better plans and execute them following the market strategy.

1. Reduces Risk

The frequent testing and deployment of code reduce the project's risk level, as now the code defects and bugs can be detected earlier. This states that these bugs and errors can be easily fixed and take less time, making the overall process cheaper. The general working speeds up the feedback mechanism that makes the communication smoother and effective.

2. Better Communication

The Continuous Integration process collaborates with the Continuous Delivery workflow that makes code sharing easy and regularized. This makes the process more transparent and collaborative among team members. In the long term, this makes the communication speed more efficient and makes sure that everyone in the organization is on the same page.

3. Higher Product Quality

Continuous Integration provides features like Code review and Code quality detection, making the identification of errors easy. If the code does not match the standard level or a mistake, it will be alerted with emails or SMS messages. Code review helps the developers to improve their programming skills continually.

4. Reduced Waiting Time

The time between the application development, integration, testing, and deployment is considerably reduced. When this time is reduced, it, in turn, reduces the waiting time that may occur in the middle. CI makes sure that all these processes continue to happen no matter what.

Jenkins:

Jenkins is an open source continuous integration/continuous delivery and deployment (CI/CD) automation software DevOps tool written in the Java programming language. It is used to implement CI/CD workflows, called pipelines.

The need for Jenkins becomes especially acute when deploying to a microservices architecture. Since one of the goals of microservices is to frequently update applications and services, the ability to do so cannot be bounded by release bandwidth. More and smaller services with faster update intervals can only be achieved by the type of automation Jenkins provides. The Jenkins X project was formerly launched in 2018 with the goal of creating a modern, cloud native Jenkins. It is also a project under the guidance of the CD Foundation. Its architecture, technology and pipeline language are completely different from Jenkins. Jenkins X is designed for Kubernetes and uses it in its own implementation. Other cloud native technologies that Jenkins X uses are Helm and Tekton.

Jenkins runs as a server on a variety of platforms including Windows, MacOS, Unix variants and especially, Linux. It requires a Java 8 VM and above and can be run on the Oracle JRE or OpenJDK. Usually, Jenkins runs as a Java servlet within a Jetty application server. It can be run on other Java application servers such as Apache Tomcat. More recently, Jenkins has been adapted to run in a Docker container. There are read-only Jenkins images available in the Docker Hub online repository. To operate Jenkins, pipelines are created. A pipeline is a series of steps the Jenkins server will take to perform the required tasks of the CI/CD process. These are stored in a plain text Jenkinsfile. The Jenkinsfile uses a curly bracket syntax that looks similar to JSON. Steps in the pipeline are declared as commands with parameters and encapsulated in curly brackets. The Jenkins server then reads the Jenkinsfile and executes its commands, pushing the code down the pipeline from committed source code to production runtime. A Jenkinsfile can be created through a GUI or by writing code directly.

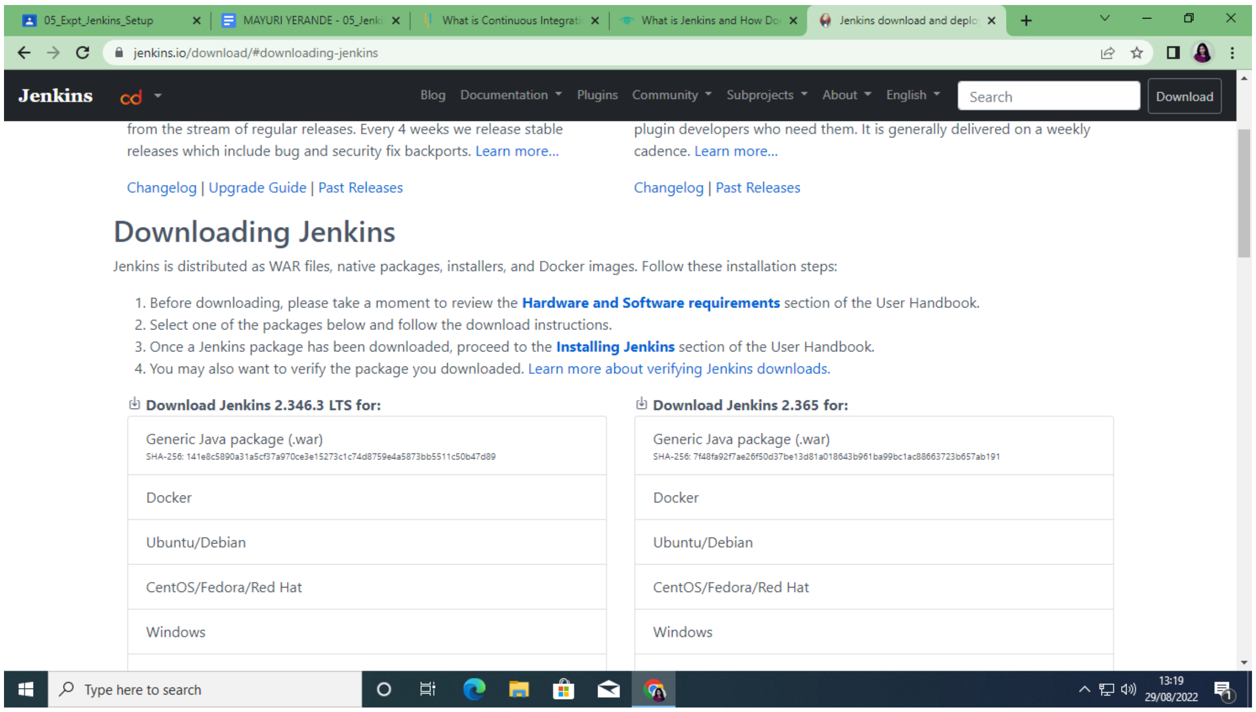
A plugin is an enhancement to the Jenkins system. They help extend Jenkins capabilities and integrate Jenkins with other software. Plugins can be downloaded from the online Jenkins Plugin repository and loaded using the Jenkins Web UI or CLI. Currently, the Jenkins community claims over 1500 plugins available for a wide range of uses.

Jenkins has been around much longer than other solutions in this space. This, plus its flexibility, has led to it being widely deployed. For this reason, Jenkins is well understood, with a broad knowledge base, extensive documentation, and abundant community resources. These resources make it easier to install, manage and troubleshoot Jenkins installation.

Installation:

Step 1: Download Jenkins

https://www.jenkins.io/doc/book/installing/



from the stream of regular releases. Every 4 weeks we release stable releases which include bug and security fix backports. Learn more...

Changelog | Upgrade Guide | Past Releases

plugin developers who need them. It is generally delivered on a weekly cadence. Learn more...

Changelog | Past Releases

Downloading Jenkins

Jenkins is distributed as WAR files, native packages, installers, and Docker images. Follow these installation steps:

1. Before downloading, please take a moment to review the **Hardware and Software requirements** section of the User Handbook.
2. Select one of the packages below and follow the download instructions.
3. Once a Jenkins package has been downloaded, proceed to the **Installing Jenkins** section of the User Handbook.
4. You may also want to verify the package you downloaded. Learn more about verifying Jenkins downloads.

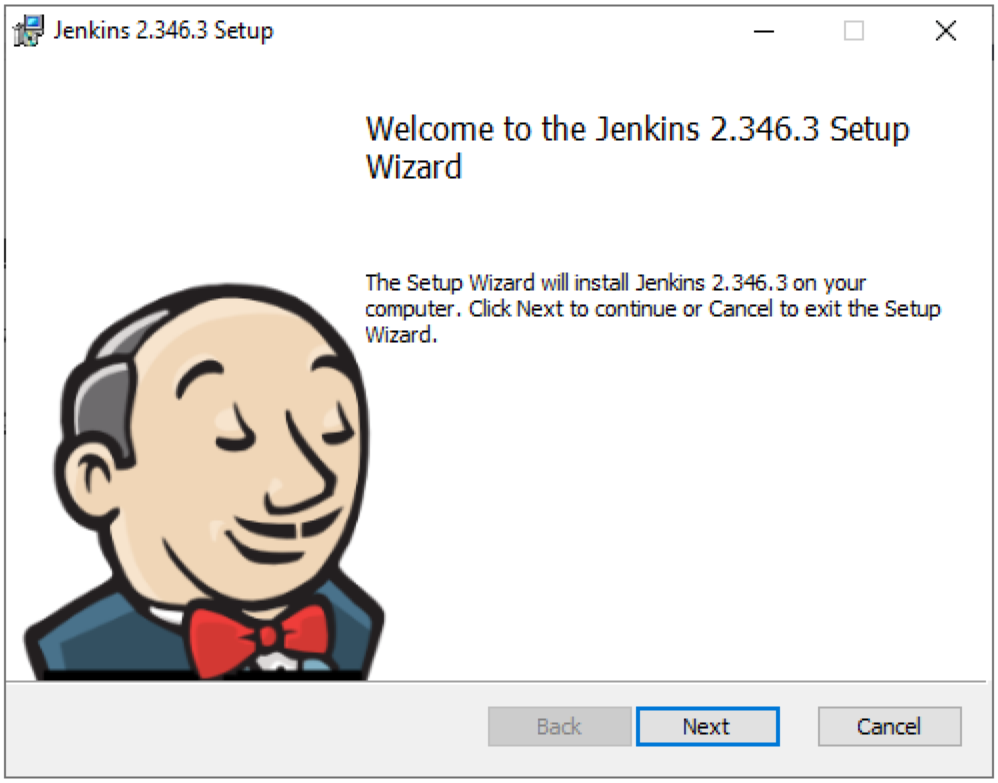
Download Jenkins 2.346.3 LTS for:

Generic Java package (.war)
SHA-256: 141e8c5890a31a5cf37a970ce3e15273c1c74d8759e4a5873bb5511c50b47d89
Docker
Ubuntu/Debian
CentOS/Fedora/Red Hat
Windows

Download Jenkins 2.365 for:

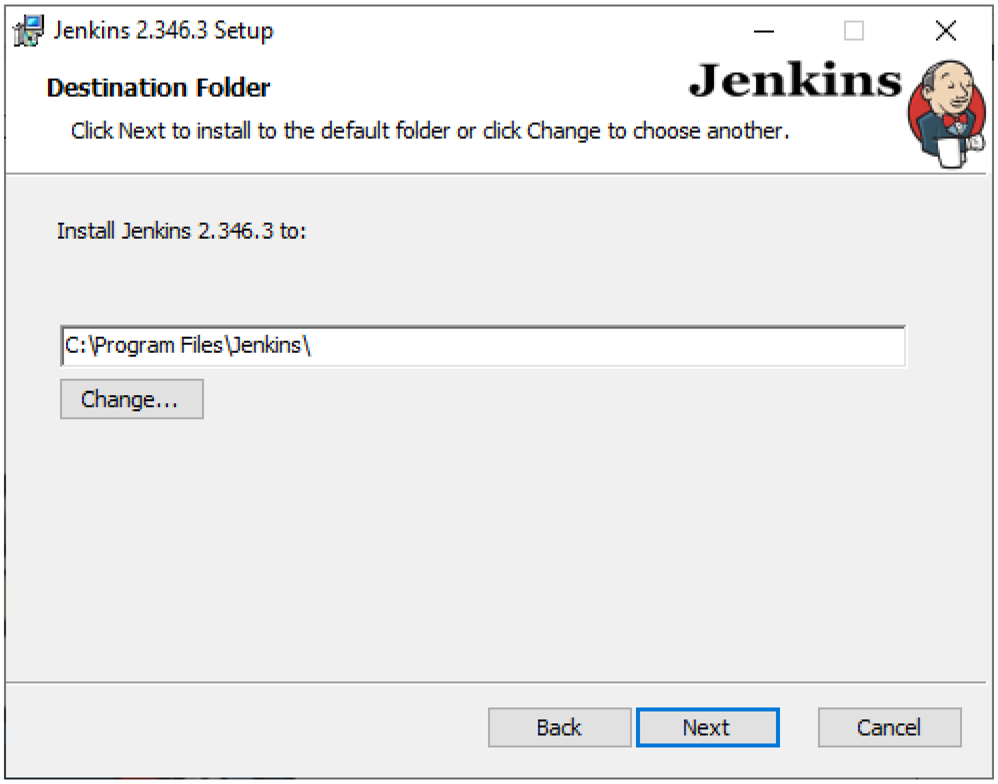
Generic Java package (.war)
SHA-256: 7f48fa92f7ae26f50d37be13d81a018643b961ba99bc1ac88663723b657ab191
Docker
Ubuntu/Debian
CentOS/Fedora/Red Hat
Windows

Step 2: Install it,click next



Welcome to the Jenkins 2.346.3 Setup Wizard

The Setup Wizard will install Jenkins 2.346.3 on your computer. Click Next to continue or Cancel to exit the Setup Wizard.



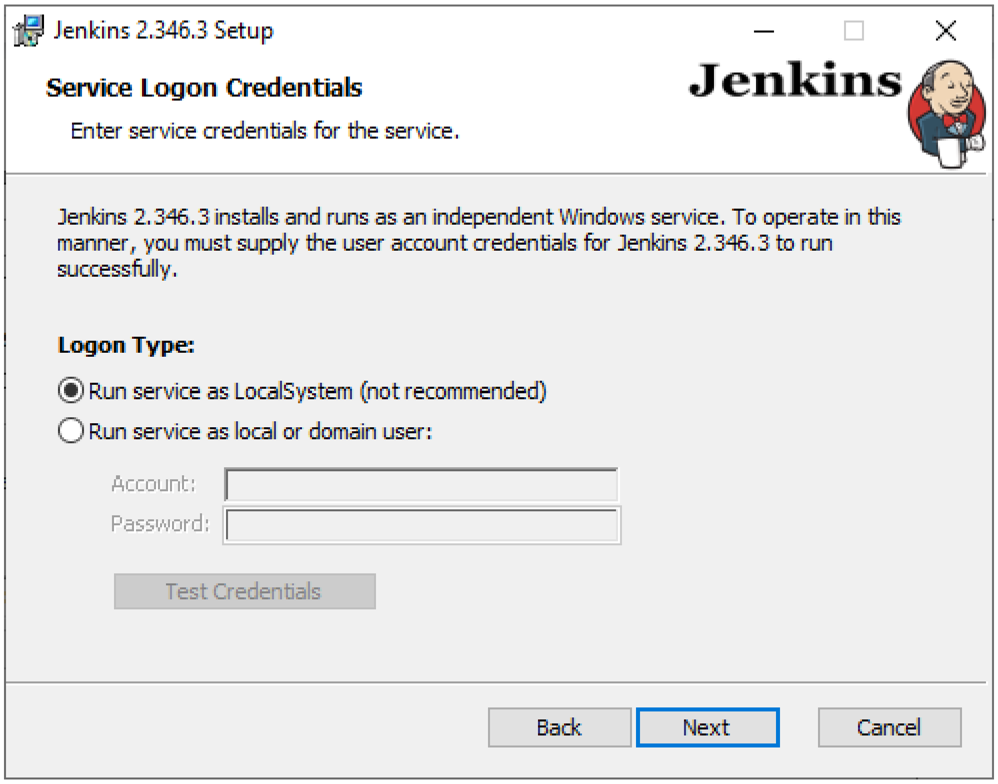
Destination Folder

Click Next to install to the default folder or click Change to choose another.

Install Jenkins 2.346.3 to:

C:\Program Files\Jenkins\

Step 3: Choose "run service as local system" option



Jenkins 2.346.3 Setup

Service Logon Credentials

Enter service credentials for the service.

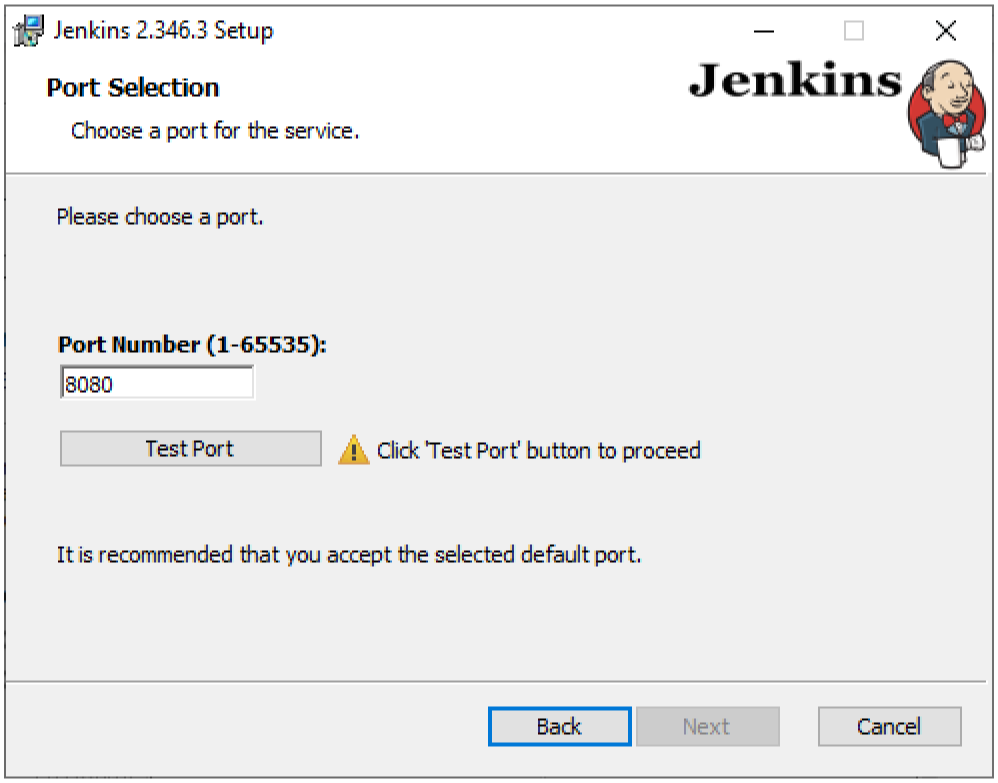
Jenkins 2.346.3 installs and runs as an independent Windows service. To operate in this manner, you must supply the user account credentials for Jenkins 2.346.3 to run successfully.

Logon Type:

[✓ Run service as LocalSystem (not recommended)]
Run service as local or domain user:

Account:
Password:

Step 4: Click on "Test port" and then click next



Jenkins 2.346.3 Setup

Port Selection

Choose a port for the service.

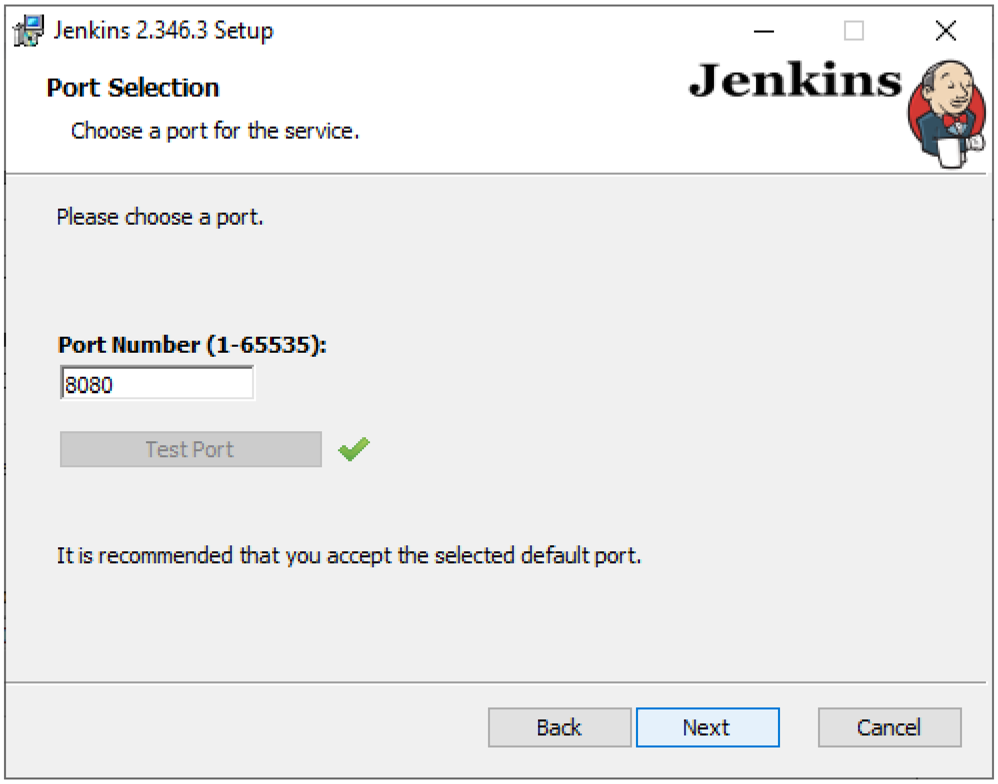
Please choose a port.

Port Number (1-65535):

8080

Click 'Test Port' button to proceed

It is recommended that you accept the selected default port.



Jenkins 2.346.3 Setup

Port Selection

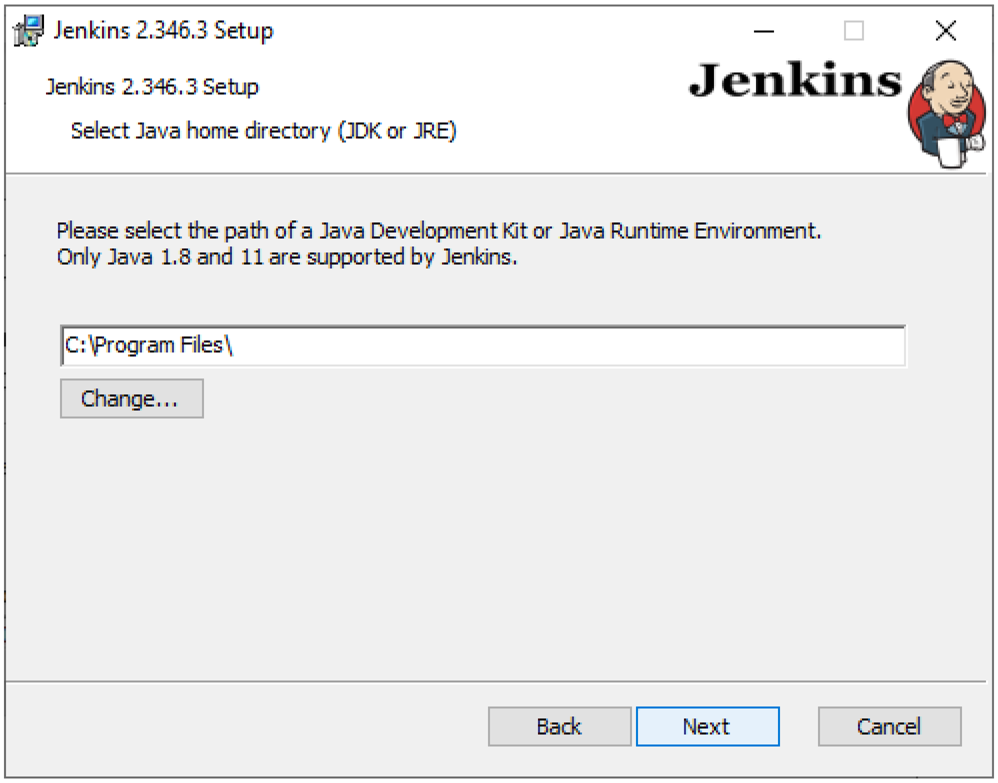
Choose a port for the service.

Please choose a port.

Port Number (1-65535):

8080

It is recommended that you accept the selected default port.



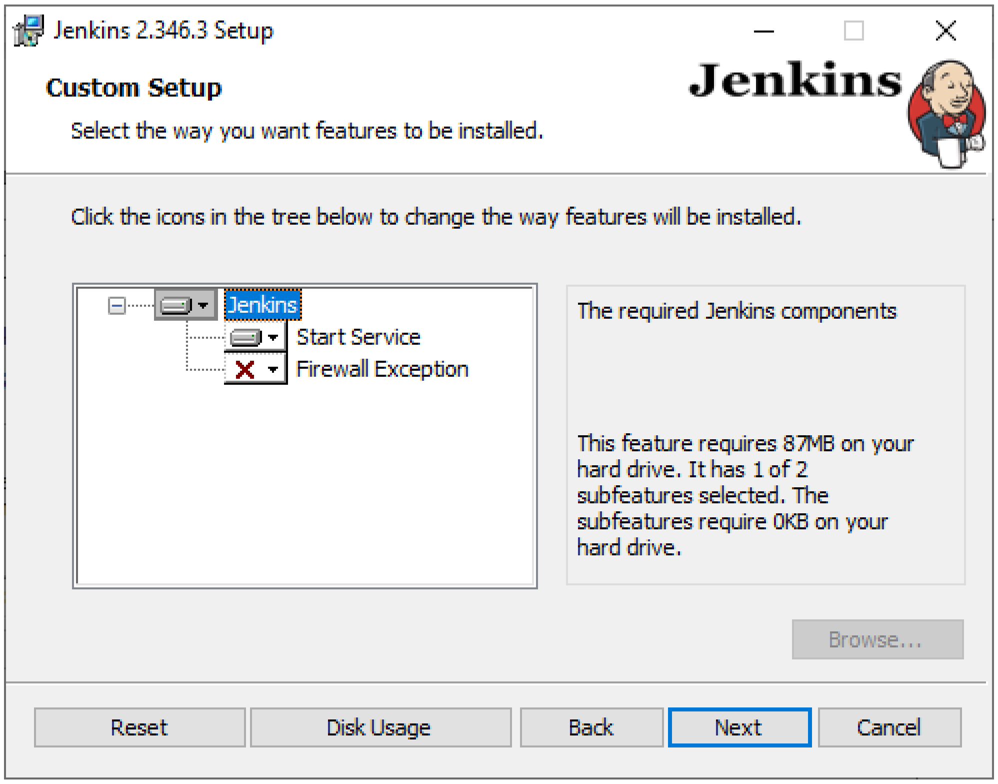
Jenkins 2.346.3 Setup

Jenkins 2.346.3 Setup

Select Java home directory (JDK or JRE)

Please select the path of a Java Development Kit or Java Runtime Environment.
Only Java 1.8 and 11 are supported by Jenkins.

C:\Program Files\



Jenkins 2.346.3 Setup

Custom Setup

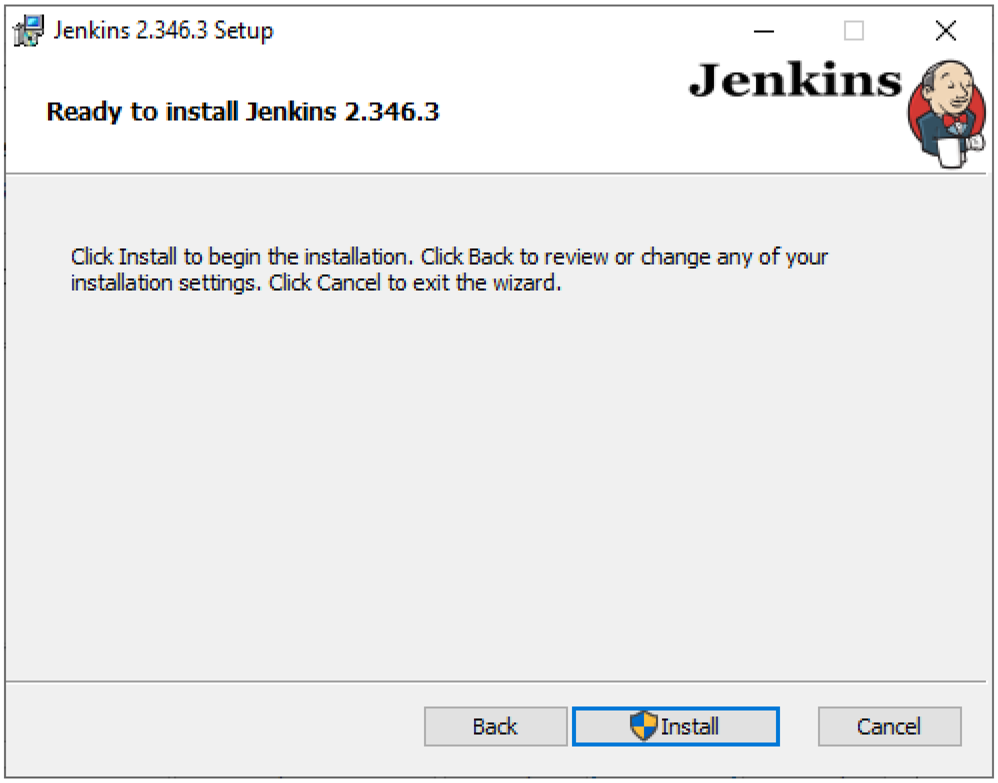
Select the way you want features to be installed.

Click the icons in the tree below to change the way features will be installed.

- Jenkins
  - Start Service
  - Firewall Exception

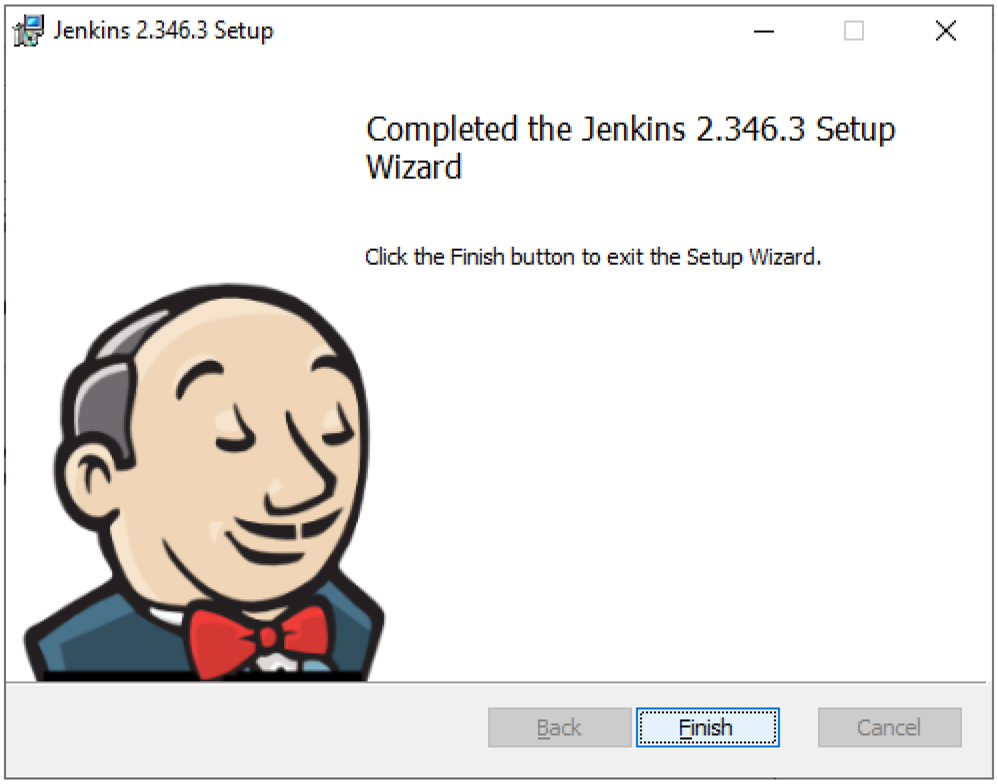
The required Jenkins components

This feature requires 87MB on your hard drive. It has 1 of 2 subfeatures selected. The subfeatures require 0KB on your hard drive.



Ready to install Jenkins 2.346.3

Click Install to begin the installation. Click Back to review or change any of your installation settings. Click Cancel to exit the wizard.

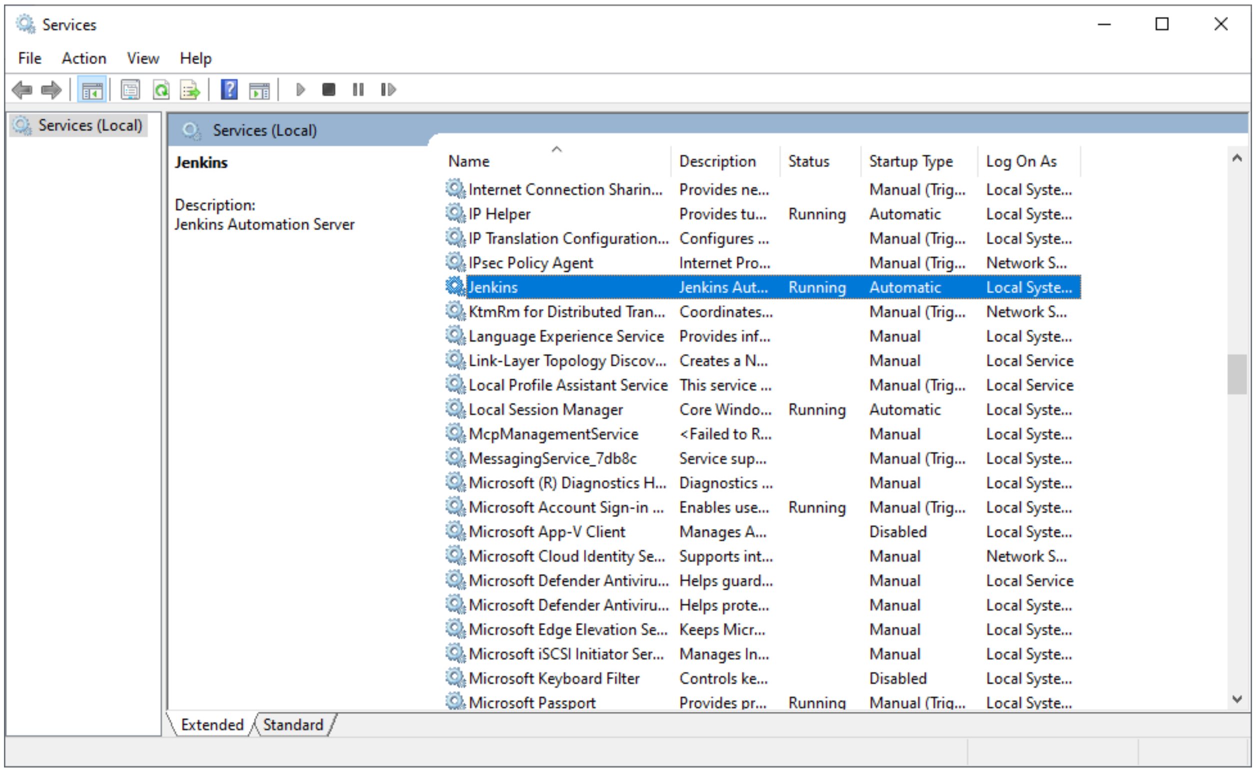


Completed the Jenkins 2.346.3 Setup Wizard

Click the Finish button to exit the Setup Wizard.

Jenkins is successfully installed.

Step 5: Make sure that Jenkins is "running"

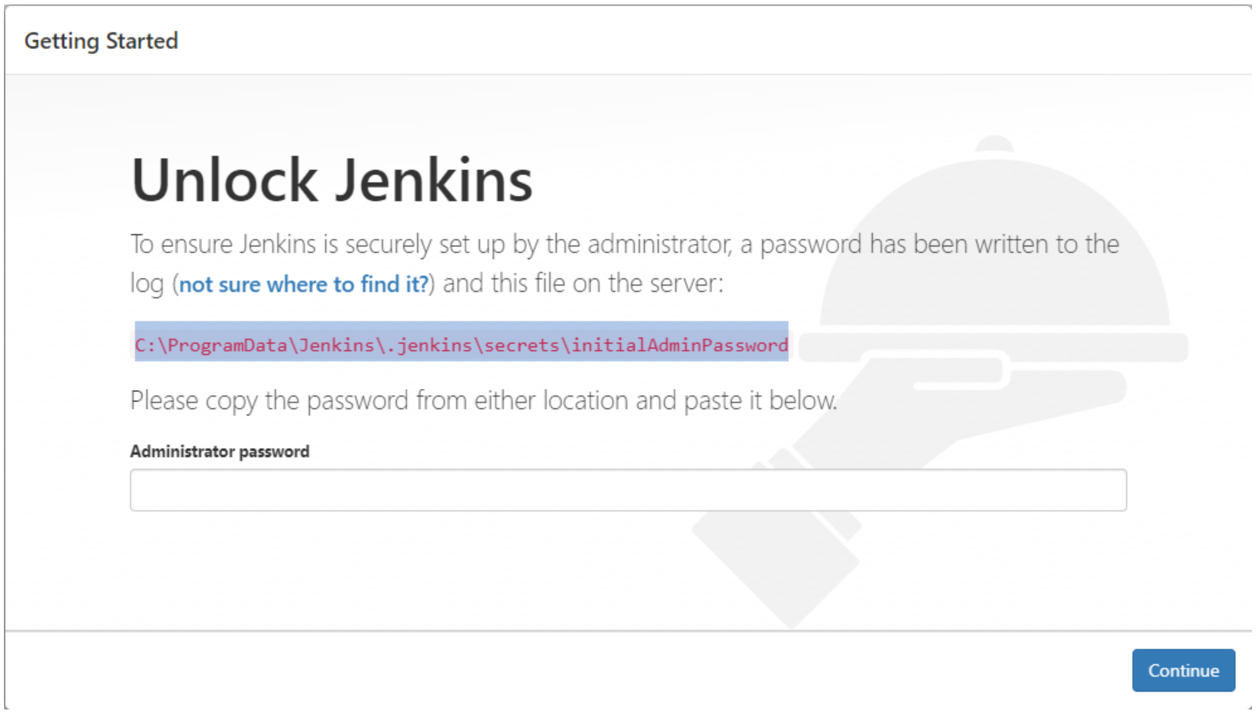


Jenkins

Description:
Jenkins Automation Server

Name	Description	Status	Startup Type	Log On As
Internet Connection Sharin...	Provides ne...		Manual (Trig...	Local Syste...
IP Helper	Provides tu...	Running	Automatic	Local Syste...
IP Translation Configuration...	Configures ...		Manual (Trig...	Local Syste...
IPsec Policy Agent	Internet Pro...		Manual (Trig...	Network S...
Jenkins	Jenkins Aut...	Running	Automatic	Local Syste...
KtmRm for Distributed Tran...	Coordinates...		Manual (Trig...	Network S...
Language Experience Service	Provides inf...		Manual	Local Syste...
Link-Layer Topology Discov...	Creates a N...		Manual	Local Service
Local Profile Assistant Service	This service ...		Manual (Trig...	Local Service
Local Session Manager	Core Windo...	Running	Automatic	Local Syste...
McpManagementService	<Failed to R...		Manual	Local Syste...
MessagingService_7db8c	Service sup...		Manual (Trig...	Local Syste...
Microsoft (R) Diagnostics H...	Diagnostics ...		Manual	Local Syste...
Microsoft Account Sign-in ...	Enables use...	Running	Manual (Trig...	Local Syste...
Microsoft App-V Client	Manages A...		Disabled	Local Syste...
Microsoft Cloud Identity Se...	Supports int...		Manual	Network S...
Microsoft Defender Antiviru...	Helps guard...		Manual	Local Service
Microsoft Defender Antiviru...	Helps prote...		Manual	Local Syste...
Microsoft Edge Elevation Se...	Keeps Micr...		Manual	Local Syste...
Microsoft iSCSI Initiator Ser...	Manages In...		Manual	Local Syste...
Microsoft Keyboard Filter	Controls ke...		Disabled	Local Syste...
Microsoft Passport	Provides pr...	Running	Manual (Trig...	Local Syste...

Step 6: Enter the password in the jenkins folder downloaded



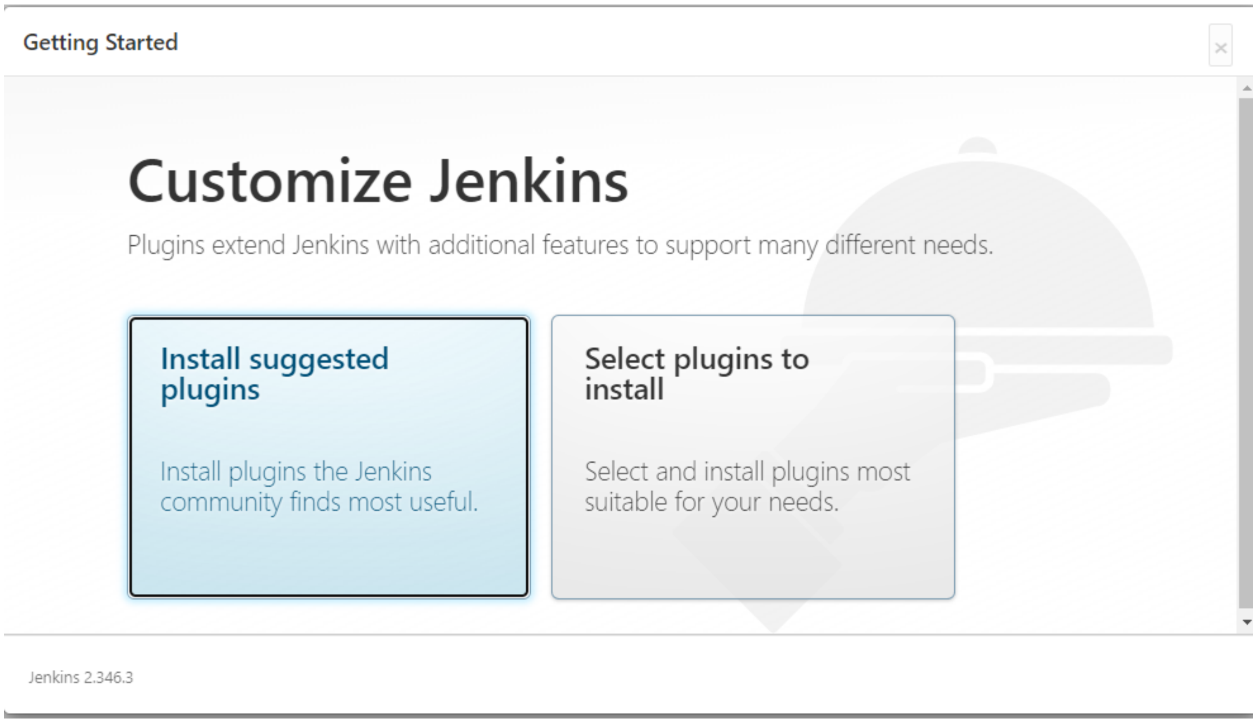
Unlock Jenkins

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



Getting Started

Customize Jenkins

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

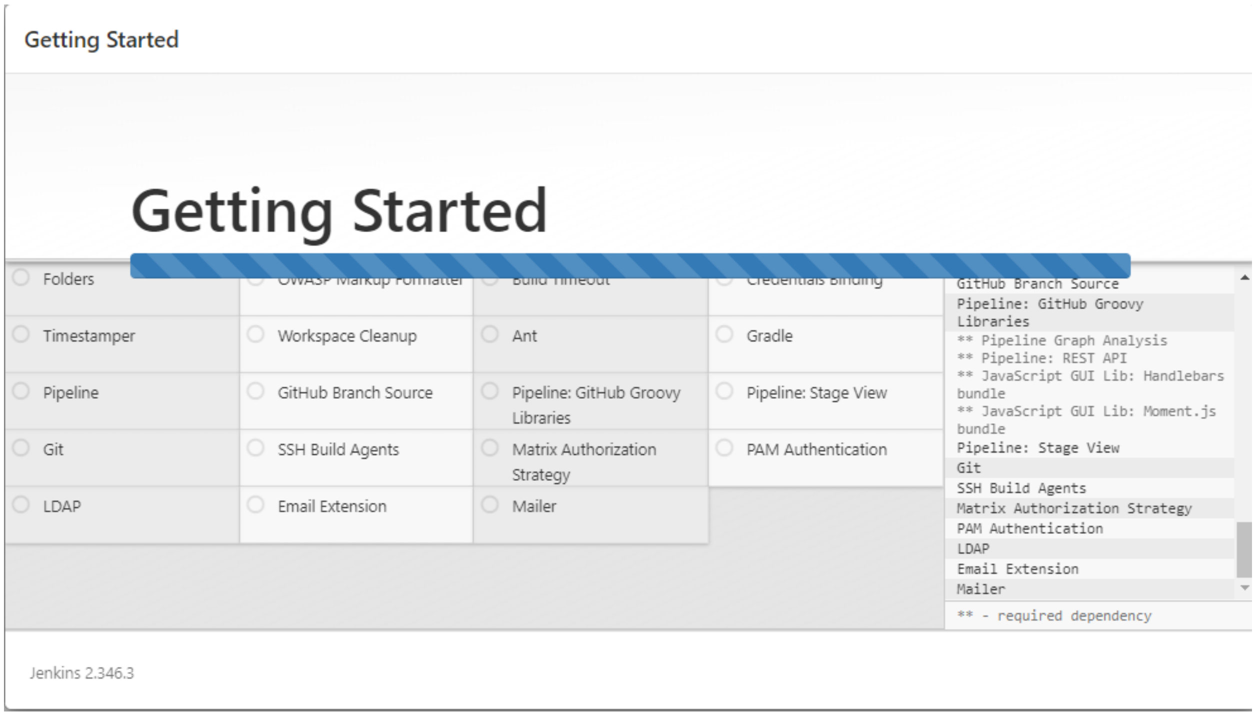
Install suggested plugins

Install plugins the Jenkins community finds most useful.

Select plugins to install

Select and install plugins most suitable for your needs.

Jenkins 2.346.3



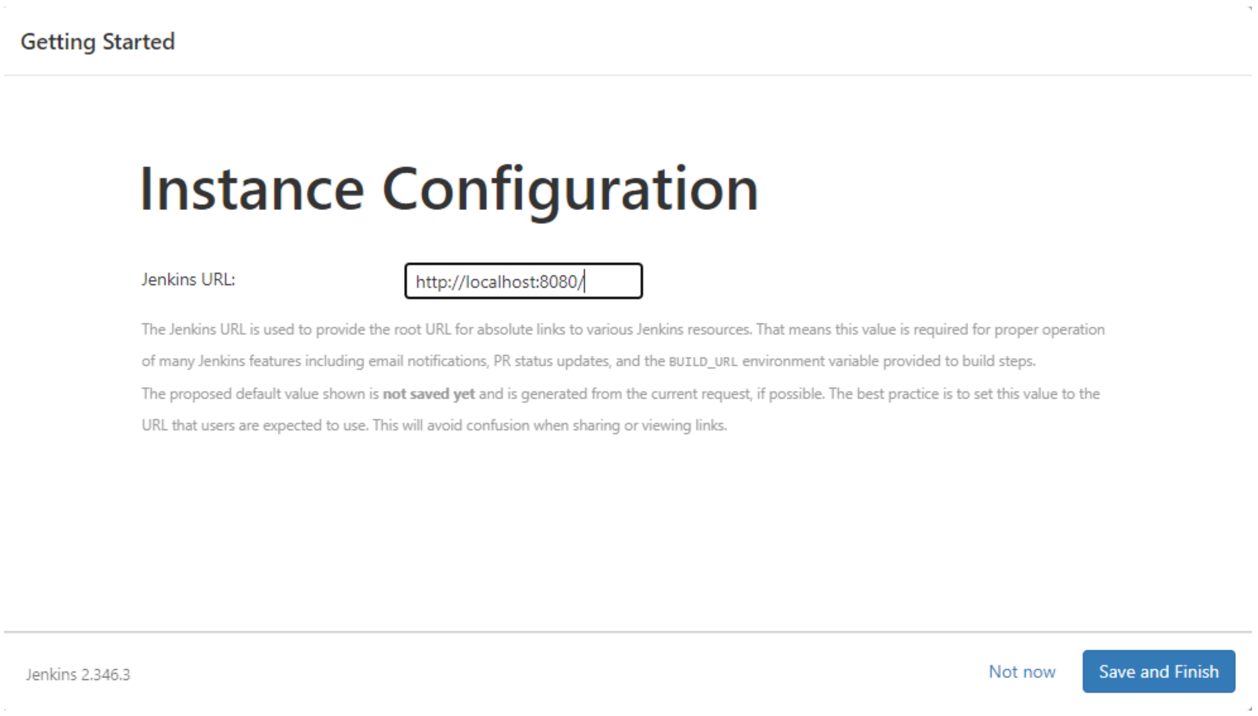
Getting Started

Getting Started

Folders	OWASP Markup Formatter	Build Timeout	Credentials Binding	
Timestamper	Workspace Cleanup	Ant	Gradle	
Pipeline	GitHub Branch Source	Pipeline: GitHub Groovy Libraries	Pipeline: Stage View	
Git	SSH Build Agents	Matrix Authorization Strategy	PAM Authentication	
LDAP	Email Extension	Mailer		

```
GitHub Branch Source
Pipeline: GitHub Groovy
Libraries
** Pipeline Graph Analysis
** Pipeline: REST API
** JavaScript GUI Lib: Handlebars
bundle
** JavaScript GUI Lib: Moment.js
bundle
Pipeline: Stage View
Git
SSH Build Agents
Matrix Authorization Strategy
PAM Authentication
LDAP
Email Extension
Mailer
** - required dependency
```

Jenkins 2.346.3



Getting Started

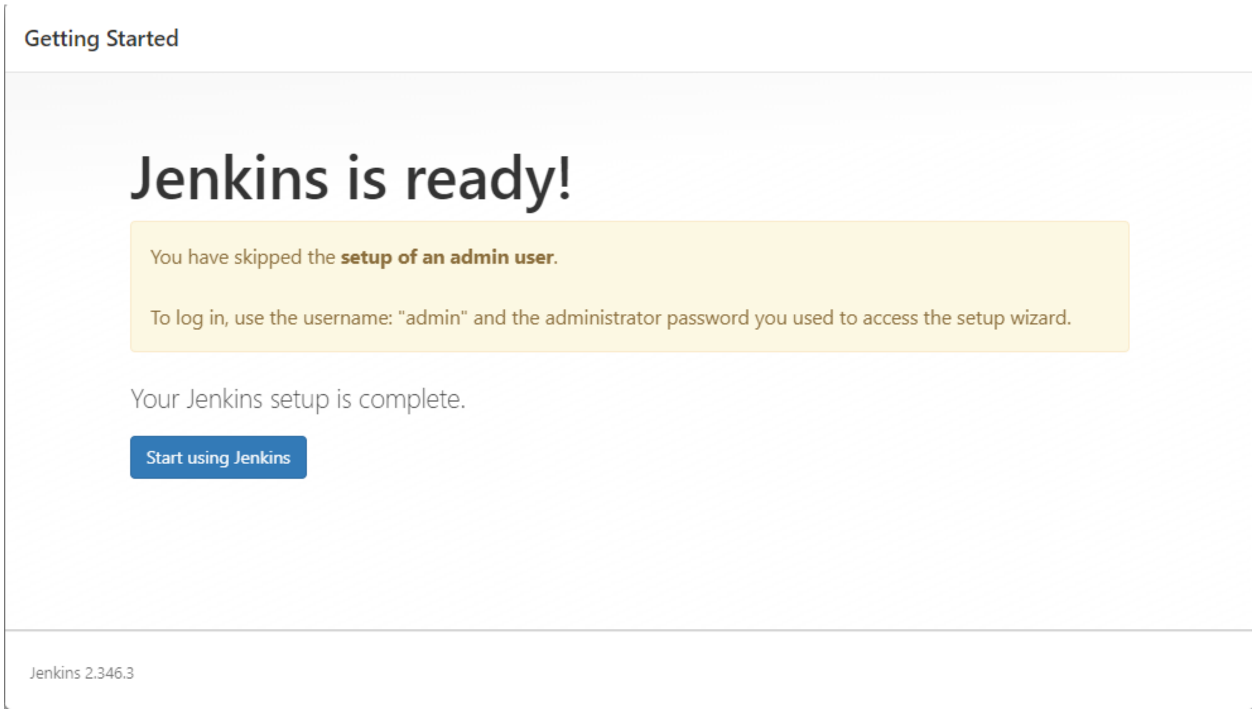
Instance Configuration

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 2.346.3



Getting Started

Jenkins is ready!

You have skipped the **setup of an admin user**.

To log in, use the username: "admin" and the administrator password you used to access the setup wizard.

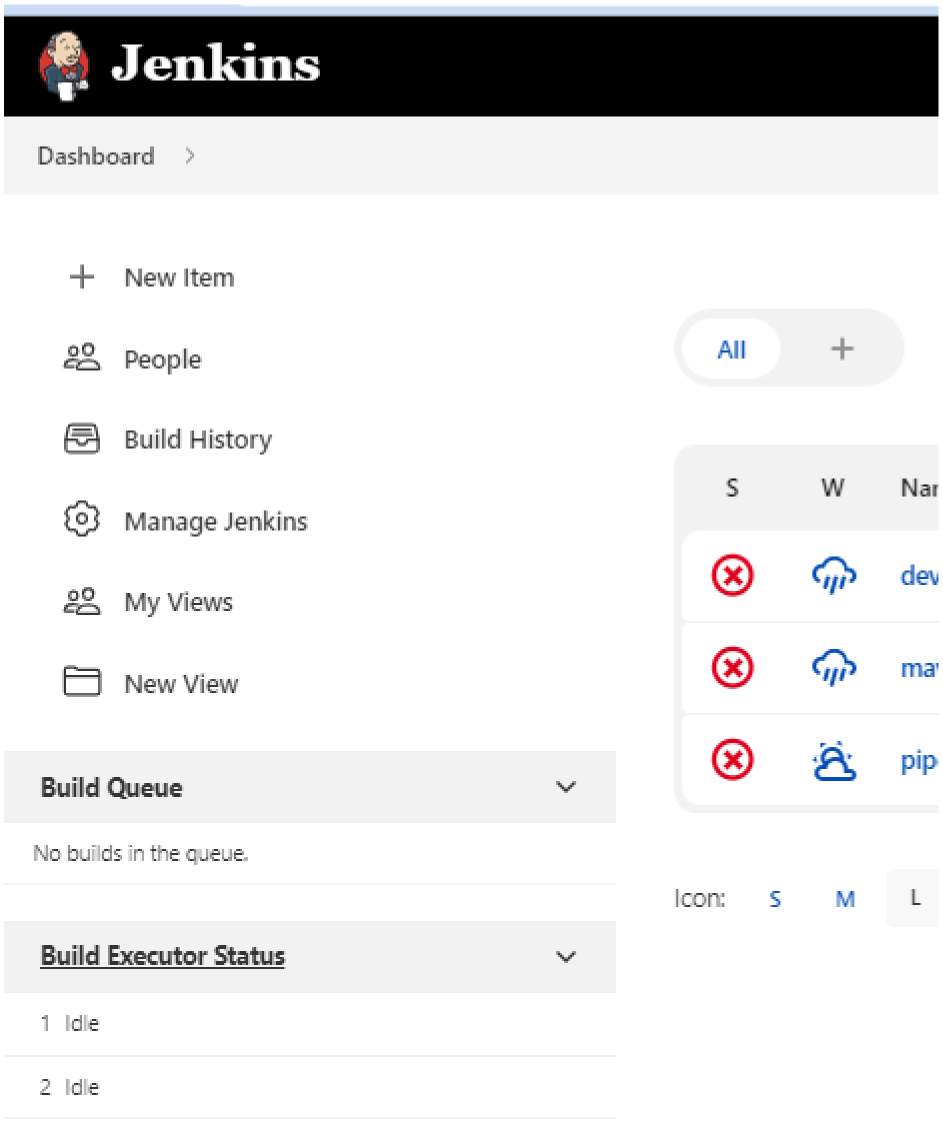
Your Jenkins setup is complete.

Jenkins 2.346.3

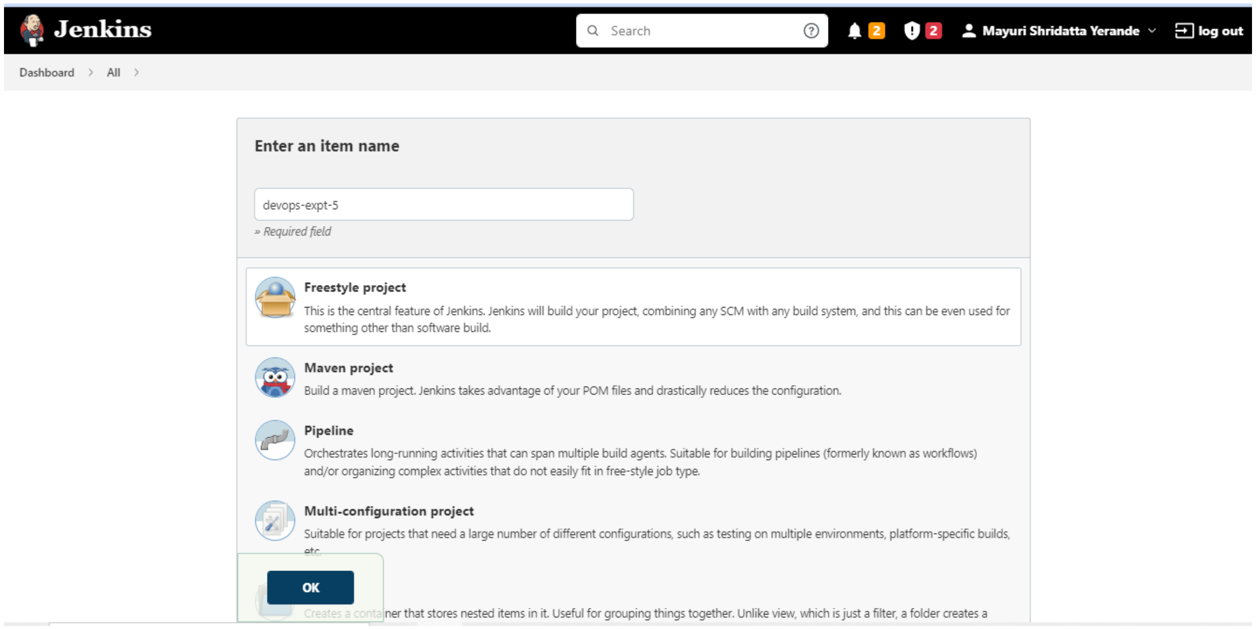
Your Jenkins is ready to use.

Build Job:

Step 1: Click on "New Item" at the top left-hand side of your dashboard



Step 2: Enter the name of the item you want to create (Devops-expt-5). Select Freestyle project



Enter an item name

devops-expt-5

» Required field

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

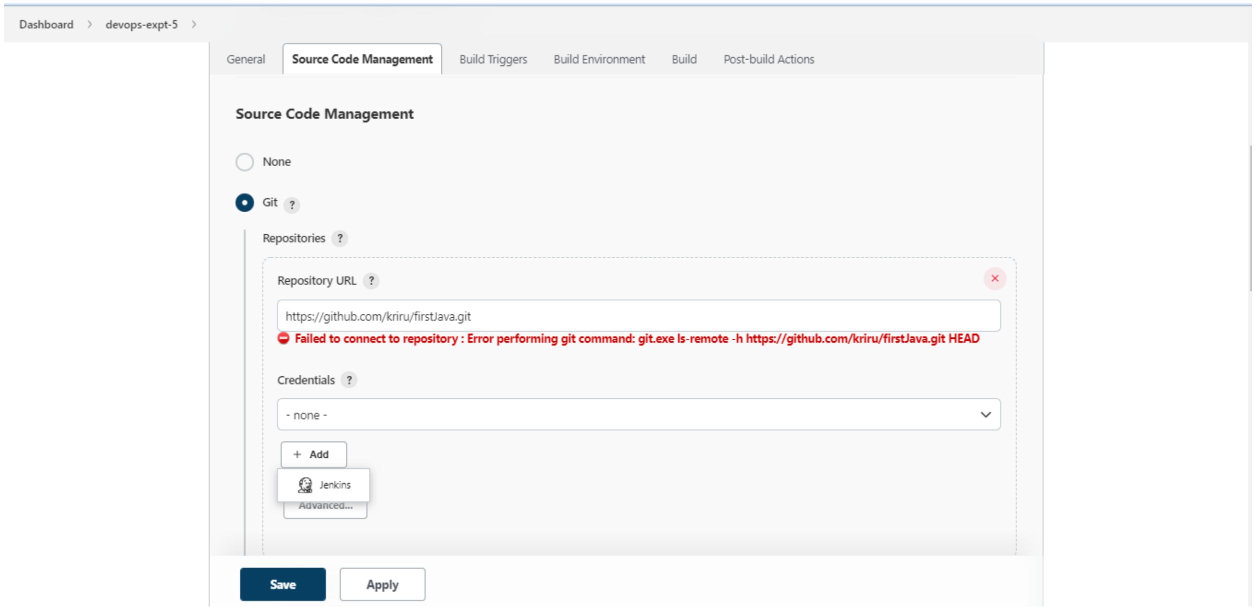
Maven project
Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a

Step 3: Enter the details of the project you want to test in description if you want Under Source Code Management, Enter your repository URL. I have a test repository located at https://github.com/kriru/firstJava.git



Source Code Management

None

Git

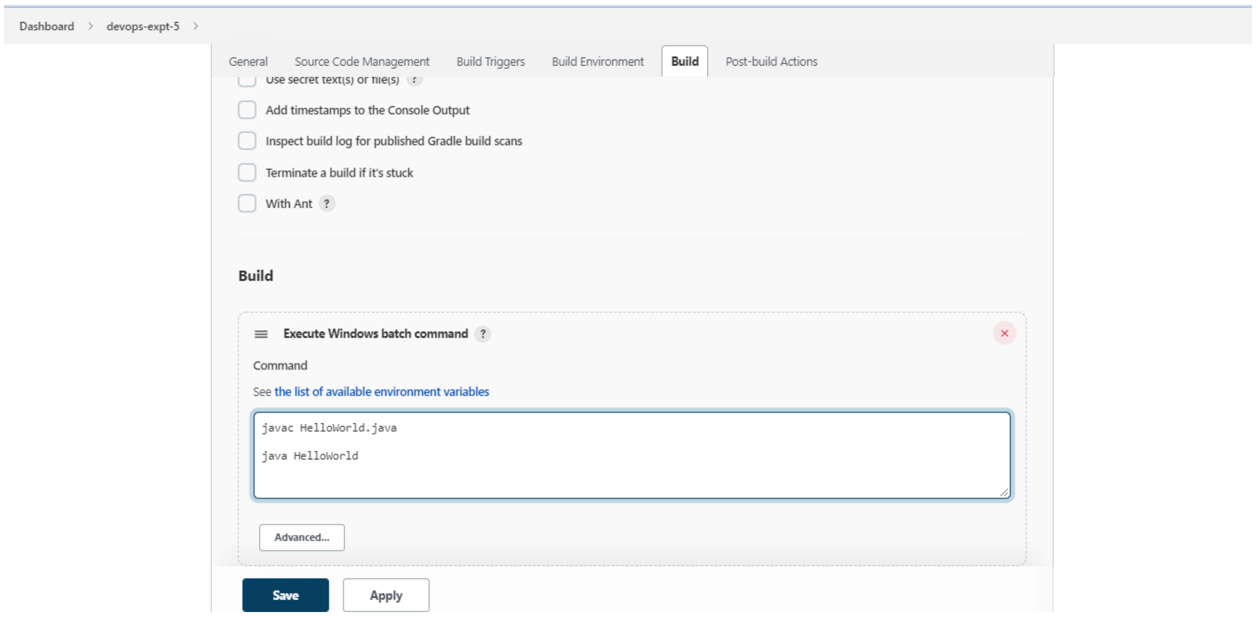
Repositories

Repository URL: https://github.com/kriru/firstJava.git

Failed to connect to repository : Error performing git command: git.exe ls-remote -h https://github.com/kriru/firstJava.git HEAD

Credentials: - none -

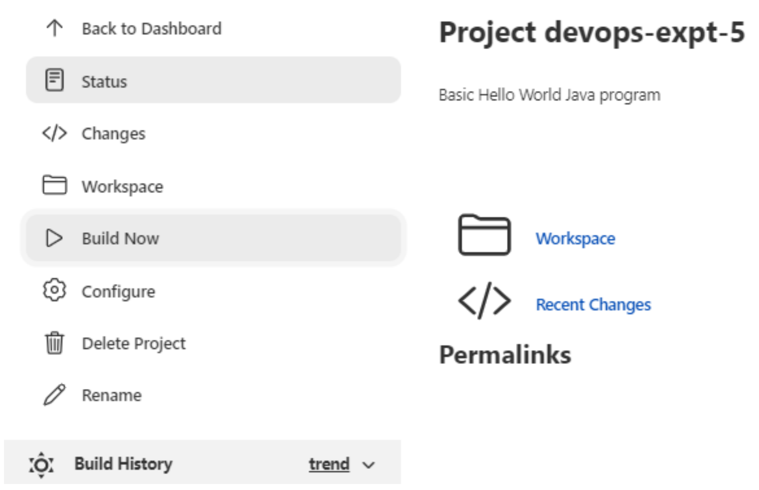
Step 4: Click on "Add build step". Click on "Execute Windows batch command" and add the commands you want to execute during the build process.
javac HelloWorld.java
java HelloWorld


Dashboard > devops-expt-5 >
General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
Use secret text(s) or file(s)
Add timestamps to the Console Output
Inspect build log for published Gradle build scans
Terminate a build if it's stuck
With Ant
Build
Execute Windows batch command
Command
See the list of available environment variables
Advanced...
Save
Apply

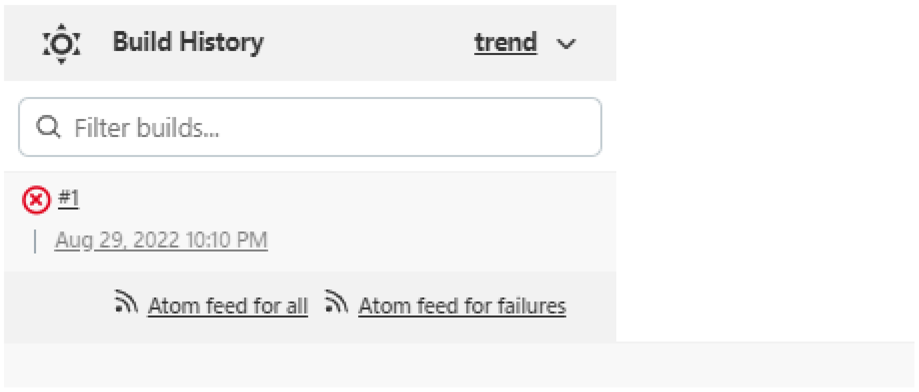
```
javac HelloWorld.java
java HelloWorld
```

Step 5: Save the project. When you have entered all the data,
1. **Click Apply**
2. **Save the project.**

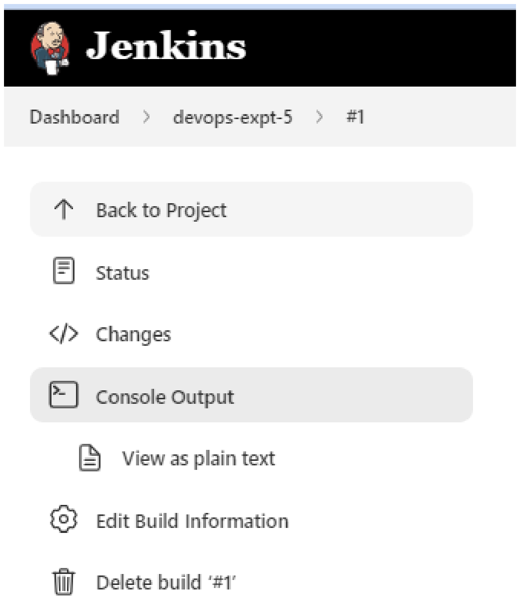
Step 6: Now, in the main screen, Click the Build Now button on the left-hand side to build the source code.

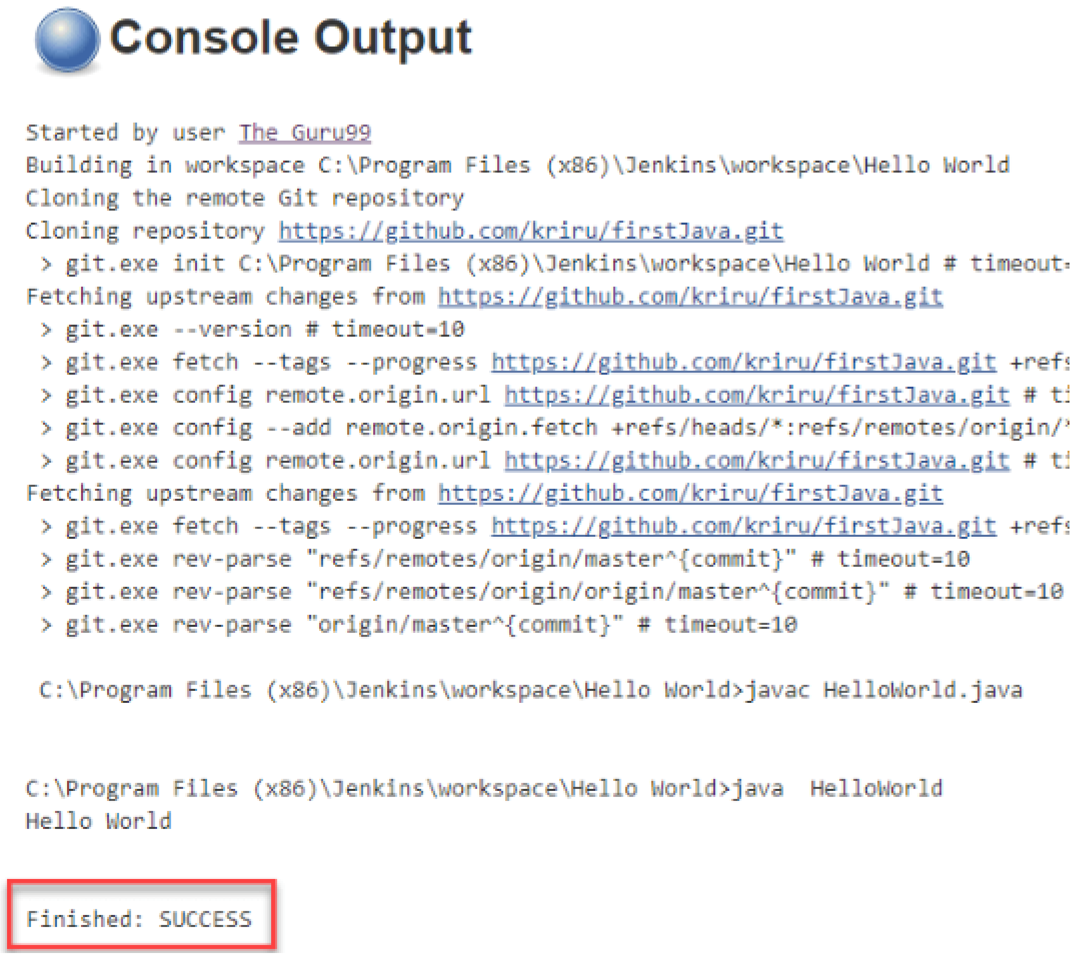


Step 7: After clicking on Build now, you can see the status of the build you run under Build History.



Step 8: Click on the build number and then Click on console output to see the status of the build you run. It should show you a success message,





Console Output

```
Started by user The Guru99
Building in workspace C:\Program Files (x86)\Jenkins\workspace\Hello World
Cloning the remote Git repository
Cloning repository https://github.com/kriru/firstJava.git
 > git.exe init C:\Program Files (x86)\Jenkins\workspace\Hello World # timeout
Fetching upstream changes from https://github.com/kriru/firstJava.git
 > git.exe --version # timeout=10
 > git.exe fetch --tags --progress https://github.com/kriru/firstJava.git +ref
 > git.exe config remote.origin.url https://github.com/kriru/firstJava.git # t
 > git.exe config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/
 > git.exe config remote.origin.url https://github.com/kriru/firstJava.git # t
Fetching upstream changes from https://github.com/kriru/firstJava.git
 > git.exe fetch --tags --progress https://github.com/kriru/firstJava.git +ref
 > git.exe rev-parse "refs/remotes/origin/master^{commit}" # timeout=10
 > git.exe rev-parse "refs/remotes/origin/origin/master^{commit}" # timeout=10
 > git.exe rev-parse "origin/master^{commit}" # timeout=10

 C:\Program Files (x86)\Jenkins\workspace\Hello World>javac HelloWorld.java


C:\Program Files (x86)\Jenkins\workspace\Hello World>java  HelloWorld
Hello World

Finished: SUCCESS
```

You have successfully built a job in jenkins.

Conclusion: We learnt about Continuous Integration and Its importance, We successfully installed Jenkins and Built a job in it.