Jenkins is an open-source automation server that helps automate tasks related to building, testing, and deploying software. It's written in Java and can run on Windows, Linux, macOS, and other Unix-like operating systems. Jenkins can be installed through native system packages, Docker, or run standalone by any machine with a Java Runtime Environment (JRE) installed.

Jenkins is a continuous integration (CI) server that manages and controls several stages of the software delivery process, including build, documentation, automated testing, packaging, and static code analysis. It helps developers integrate any changes in their projects easily.

Jenkins provides hundreds of plugins for building, deploying, and automating projects. It also implements CI/CD workflows, called pipelines, which automate testing and reporting on isolated changes in a larger code base in real time. Pipelines also facilitate the integration of disparate branches of the code into a main branch.

Jenkins is a popular DevOps tool used by many software teams.

Chiradeep BasuMallick [Chiradeep BasuMallick](https://www.spiceworks.com/user/about/chiradeepbasumallick) Technical Writer

*Last Updated:* July 20, 2022

***Jenkins is defined as an open-source solution comprising an automation server to enable continuous integration and continuous delivery (CI/CD), automating the various stages of software development such as test, build, and deployment. This article explains how Jenkins works, its key features and use cases, and the functionality of a Jenkins pipeline.***

## **Table of Contents**

* [What Is Jenkins?](https://www.spiceworks.com/tech/devops/articles/what-is-jenkins/#_001)
* [What Is Jenkins Used For?](https://www.spiceworks.com/tech/devops/articles/what-is-jenkins/#_002)
* [How Does Jenkins Work?](https://www.spiceworks.com/tech/devops/articles/what-is-jenkins/#_003)
* [What Is a Jenkins Pipeline?](https://www.spiceworks.com/tech/devops/articles/what-is-jenkins/#_004)
* [Key Features of Jenkins](https://www.spiceworks.com/tech/devops/articles/what-is-jenkins/#_005)

What Is Jenkins?

Jenkins is an open-source solution comprising an automation server to enable continuous integration and continuous delivery (CI/CD), automating the various stages of software development such as build, test, and deployment.



Jenkins is a Java-based open-source automation platform with plugins designed for continuous integration. It is used to continually create and test software projects, making it easier for developers and DevOps engineers to integrate changes to the project and for consumers to get a new build. It also enables you to release your software continuously by interacting with various testing and deployment methods.

Organisations may use Jenkins to automate and speed up the software development process. Jenkins incorporates a variety of development life-cycle operations, such as build, document, test, package, stage, deploy, static analysis, and more.

Jenkins is a well-known continuous integration tool developed initially by Hudson before it became available on the open-source market. Hudson was created by Kohsuke Kawaguchi in 2004 while working at Sun Microsystems (acquired by Oracle). There was a disagreement between Oracle and the Hudson community about the infrastructure employed after Oracle purchased Sun Microsystems in 2010.

In 2011, the Hudson community unanimously accepted a referendum to alter the project name from Hudson to Jenkins, resulting in the creation of the first “Jenkins” project. Hudson was later donated to the Eclipse Foundation and is no longer being worked on. Jenkins development is currently administered as an open-source project under the direction of the CD Foundation, a Linux Foundation initiative.

Jenkins is a widely used program with over 300,000 installations worldwide and rising daily. Software firms may speed up their software development process by adopting Jenkins, which can automate test and build at a high pace. It’s a server-based application that necessitates using a web server such as Apache Tomcat.

What Is Jenkins Used For?

Jenkins software’s popularity stems from its ability to track and monitor repetitive activities that emerge throughout a project’s development. For example, if your team is working on a project, Jenkins will continually test your builds and alert you to any mistakes early in the process. Its top use cases include:

1. Deploying code into production

If all of the tests developed for a feature or release branch are green, Jenkins or another CI system may automatically publish code to staging or production. This is often referred to as continuous deployment. Changes are done before a merging action can also be seen. One may do this in a dynamic staging environment. Then it’s distributed to a central staging system, a pre-production system, or even a production environment when combined.

2. Enabling task automation

Another instance in which one may use Jenkins is to automate workflows and tasks. If a developer is working on several environments, they will need to install or upgrade an item on each of them. If the installation or update requires more than 100 steps to complete, it will be error-prone to do it manually. Instead, you can write down all the steps needed to complete the activity in Jenkins. It will take less time, and you can complete the installation or update without difficulty.

3. Reducing the time it takes to review a code

Jenkins is a CI system that may communicate with other DevOps tools and notify users when a merge request is ready to merge. This is typically the case when all tests have been passed and all other conditions have been satisfied. Furthermore, the merging request may indicate the difference in code coverage. Jenkins cuts the time it takes to examine a merge request in half. The number of lines of code in a component and how many of them are executed determines code coverage. Jenkins supports a transparent development process among team members by reducing the time it takes to review a code.

4. Driving continuous integration

Before a change to the software can be released, it must go through a series of complex processes. The Jenkins pipeline enables the interconnection of many events and tasks in a sequence to drive continuous integration. It has a collection of plugins that make integrating and implementing continuous integration and delivery pipelines a breeze. A Jenkins pipeline’s main feature is that each assignment or job relies on another task or job.

On the other hand, continuous delivery pipelines have different states: test, build, release, deploy, and so on. These states are inextricably linked to one another. A CD pipeline is a series of events that allow certain states to function.

5. Increasing code coverage

Jenkins and other CI servers may verify code to increase test coverage. Code coverage improves as a result of tests. This encourages team members to be open and accountable. The results of the tests are presented on the build pipeline, ensuring that team members adhere to the guidelines. Like code review, comprehensive code coverage guarantees that testing is a transparent process for all team members.

6. Enhancing coding efficiency

Jenkins dramatically improves the efficiency of the development process. For example, a command prompt code may be converted into a GUI button click using Jenkins. One may accomplish this by encapsulating the script in a Jenkins task. One may parameterize Jenkins tasks to allow for customization or user input. Hundreds of lines of code can be saved as a result.

Further, it supports manual testing where necessary without switching environments. When code is hosted locally, it does not always work well when pushed to a central system on a private or public cloud. This occurs because things change by the time they push. Continuous integration on Jenkins allows for manual testing that compares code to the current state of a code base in a production-like environment.

7. Simplifying audits

When Jenkins tasks run, they collect console output from stdout and stderr parameters. This makes troubleshooting using Jenkins extremely straightforward. You may assess run timing and find the slowest step utilizing the time stamper plugin, allowing you to tweak the performance of each operation.

8. Using Slack for synchronisation

A major Jenkins use case is its interoperability with Slack. A centralized communication platform is a must-have for large teams, and one of the most popular platforms for this purpose is Slack. Jenkins may be integrated with Slack, allowing communication such as triggered activities, their times, users’ names, and outcomes to be shared with others.

How Does Jenkins Work?

Jenkins may be operated as a server on various operating systems, including Windows, macOS, Unix versions, and, most notably, Linux. It also runs on the Oracle JRE or OpenJDK and requires a Java 8 virtual machine or higher. Jenkins is often executed as a Java servlet within a Jetty application server, and other Java application servers, such as Apache Tomcat, can be used to run it.

Jenkins has recently been modified to operate in a Docker container. Further, it is available as a Web Application Resource (WAR) archive, installation packages for major operating systems, Homebrew packages, Docker images, and source code.

It automates every stage of the development process, from integration to deployment. Every time a developer pushes a change to the source code repository, it performs a build.

The commits are usually made to a development branch. Before releasing the build into production, Jenkins may deploy it to an environment that allows for any necessary user acceptance testing (UAT). To achieve continuous delivery (CD), these UAT tests may be automated using a tool like Selenium.

If the tests pass, you can merge the code into the master branch, where a “**golden**” build can be built and pushed immediately into production without any manual intervention. Amazon, Facebook, and Google are examples of companies that have achieved the 100% continuous delivery milestone and may deploy to production many times daily.

CI/CD pipeline tools are specialized software applications that help with the practices of continuous integration and continuous delivery throughout the software development lifecycle (SDLC). A CI/CD pipeline is an automated process that streamline the creation, testing, and deployment of applications. "CI" represents continuous integration, where developers frequently merge code changes into a central repository, allowing early detection of issues.

What Is a Jenkins Pipeline?

Pipelines are needed to run Jenkins. A pipeline is a set of steps the Jenkins server will execute to complete the CI/CD process’s necessary tasks. In the context of Jenkins, a pipeline refers to a collection of jobs (or events) connected in a specific order. It is a collection of plugins that allow the creation and integration of Continuous Delivery pipelines in Jenkins.

Here are some CI/CD pipeline tools:

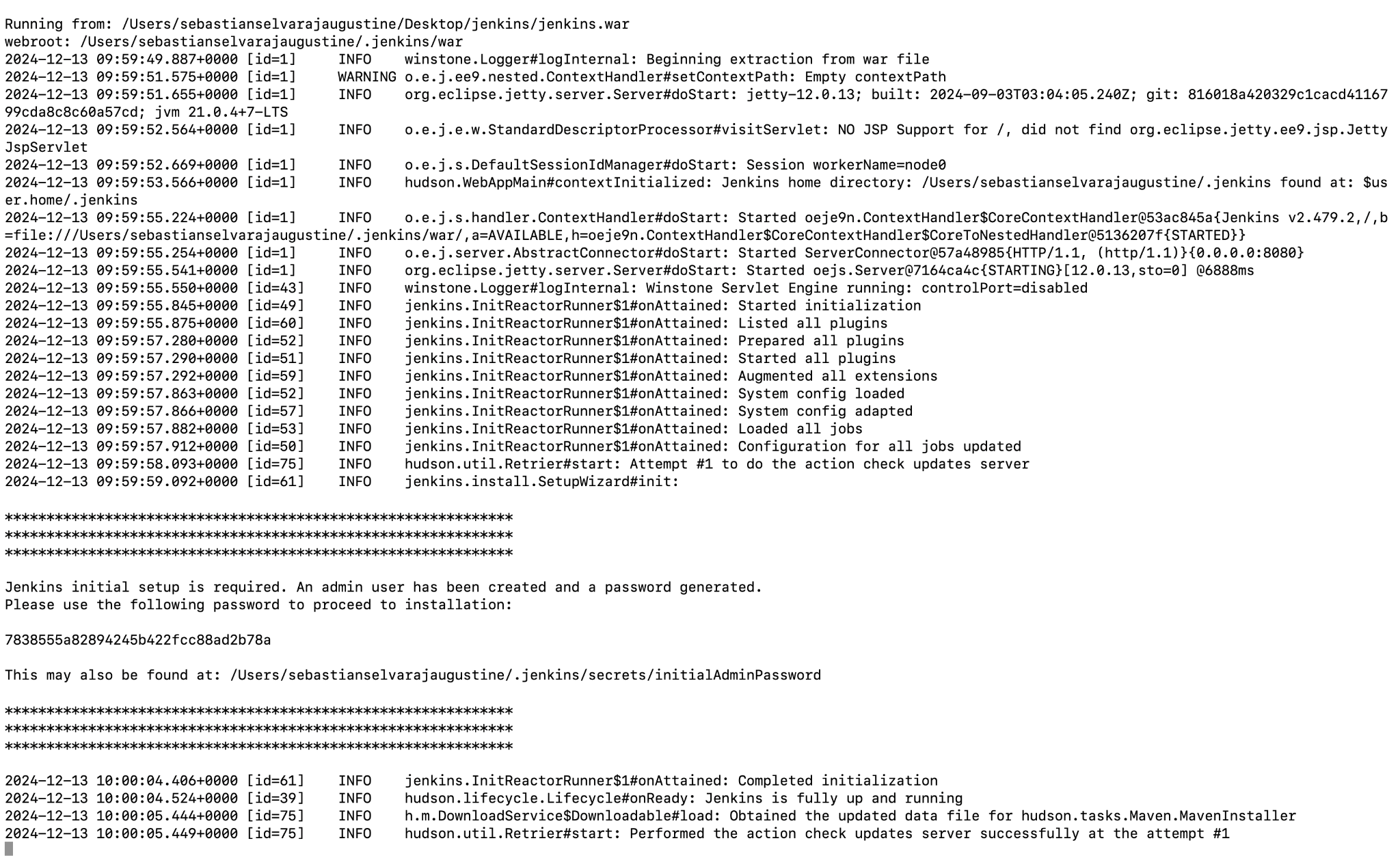
* 
* Jenkins
* An open-source automation server that can run everything from a CI server to a CD hub. Jenkins is written in the Java programming language and offers many different tools, languages, and automation tasks to aid in pipeline creation.
* 
* CircleCI
* A cloud platform that supports container-based builds for a consistent environment with parallel and distributed testing. CircleCI also supports multiple programming languages and frameworks, allowing development teams to customize their workflows.
* 
* Travis CI
* A cloud-based continuous integration service that specializes in building and testing software projects. Travis CI is particularly noted for its seamless integration with GitHub and Bitbucket.
* 
* GitLab
* A collection of tools for controlling various phases of the software development lifecycle. The primary offering is a Git repository manager for the web with tools for issue tracking, analytics, and a wiki.

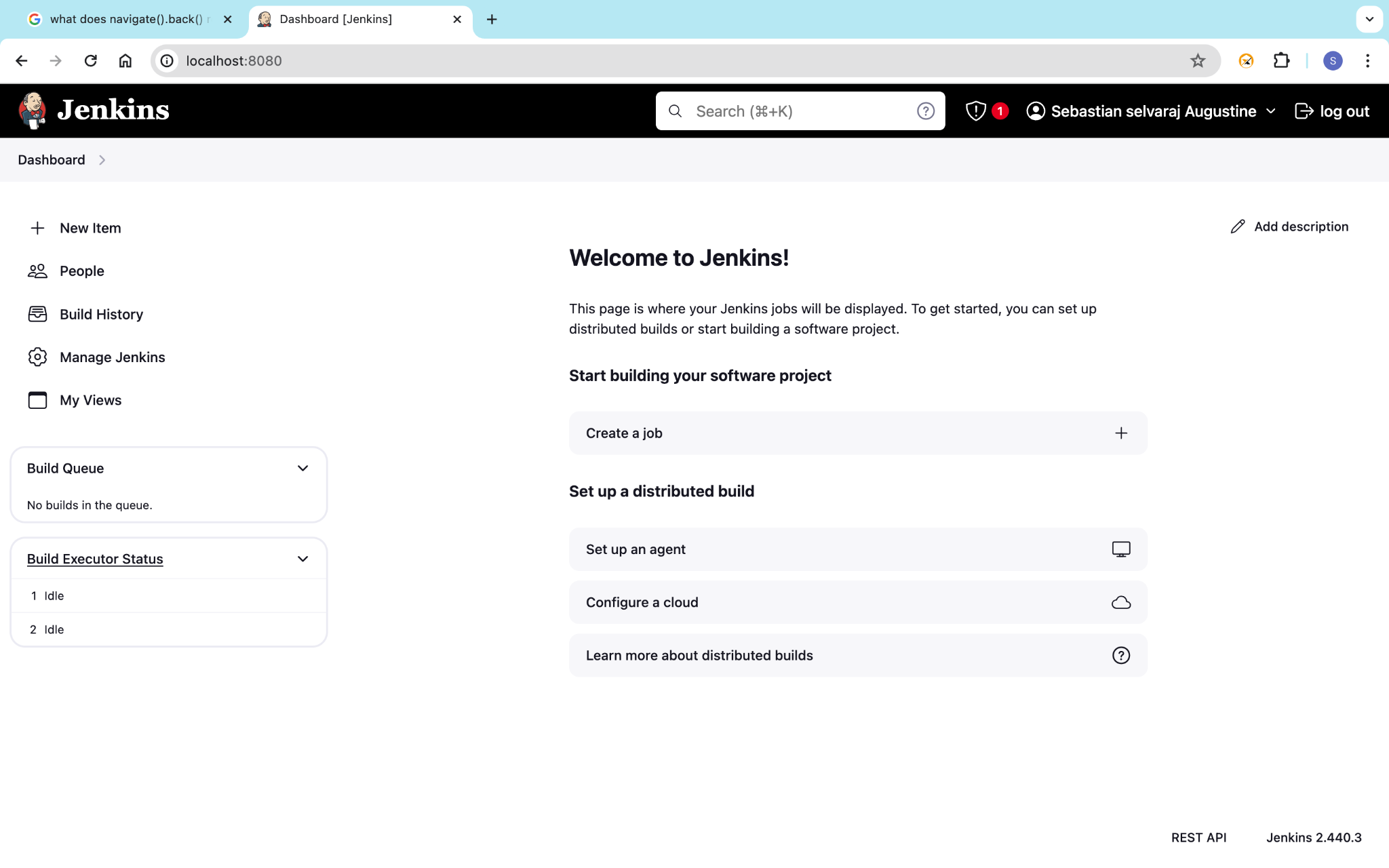
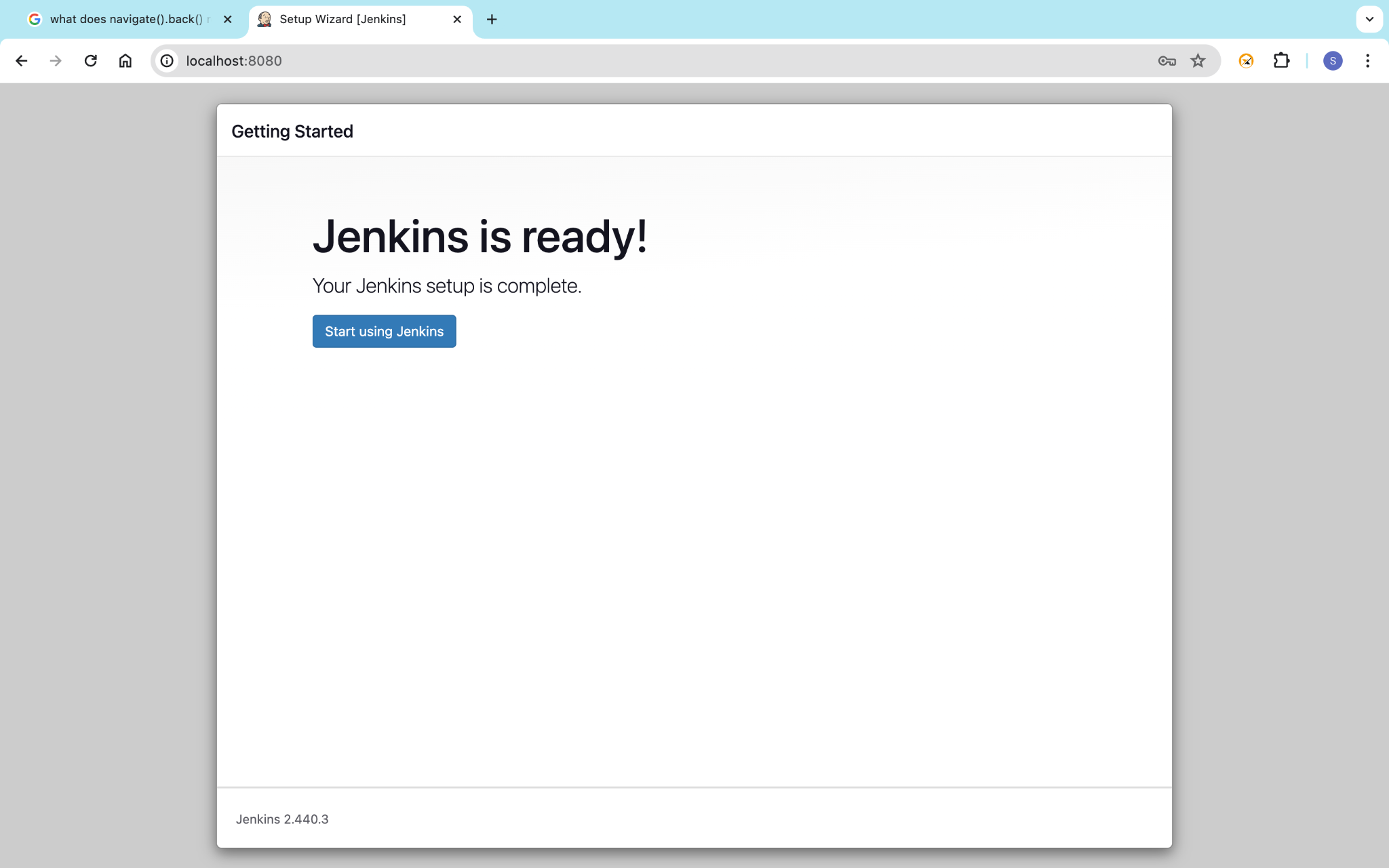
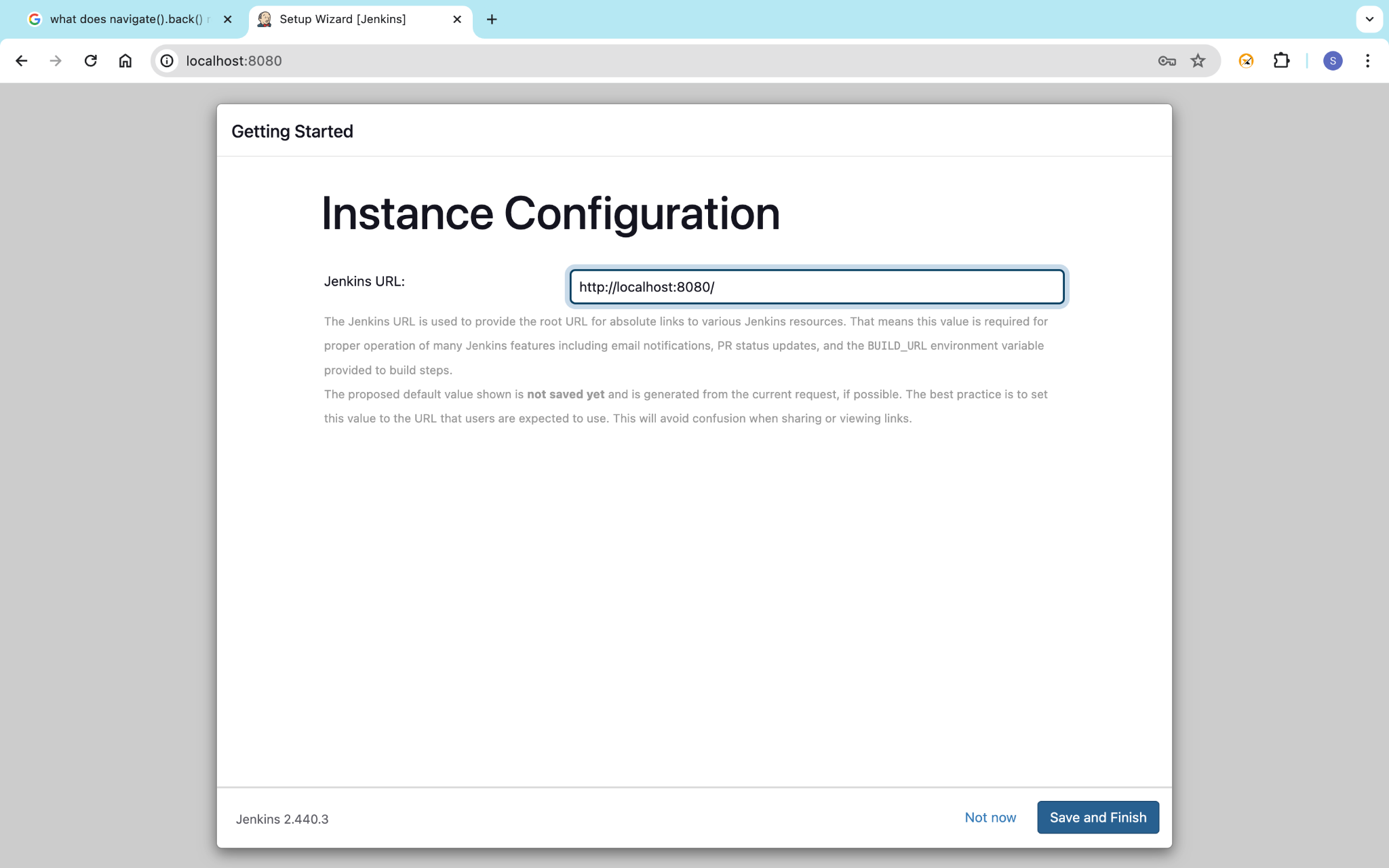
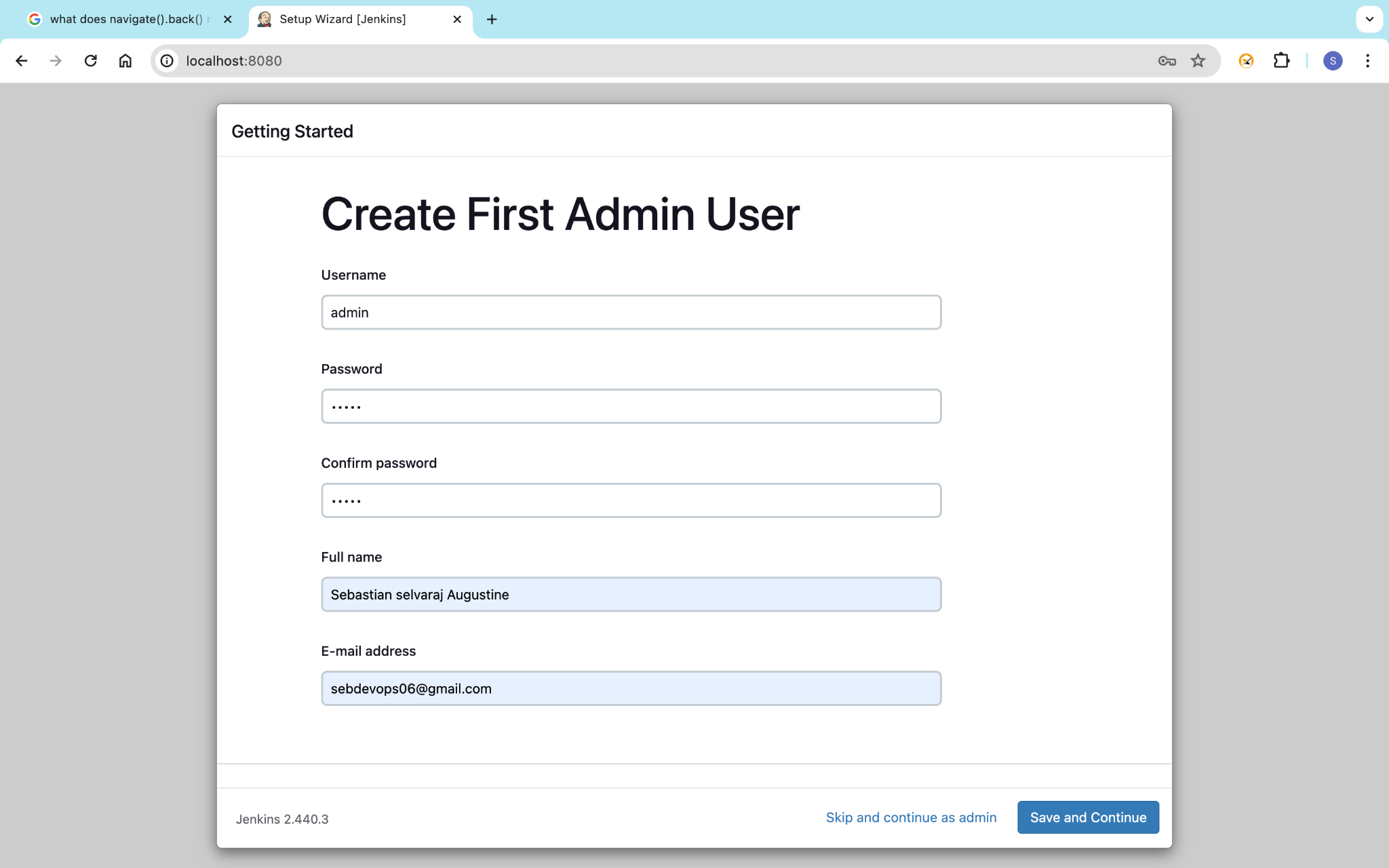
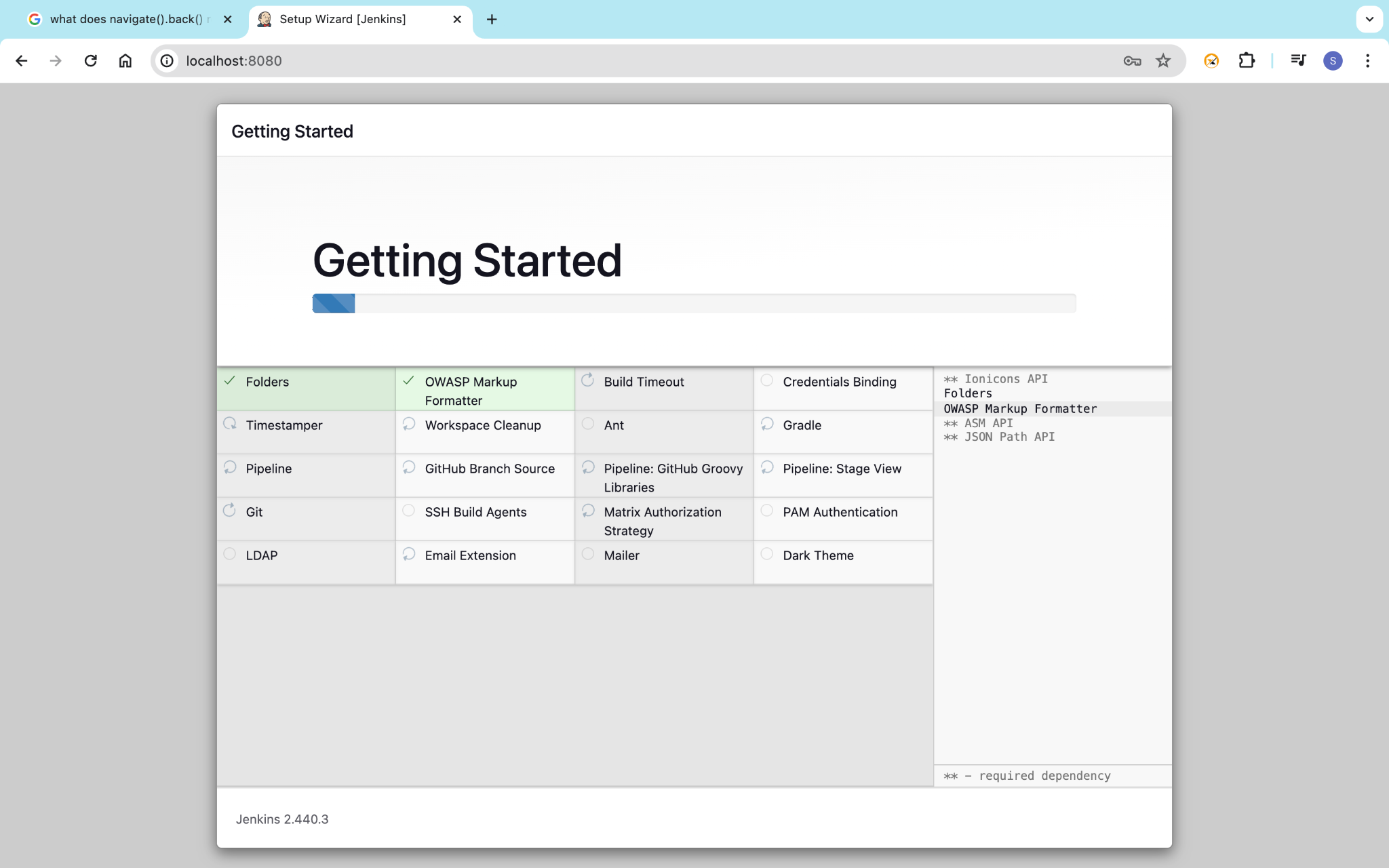
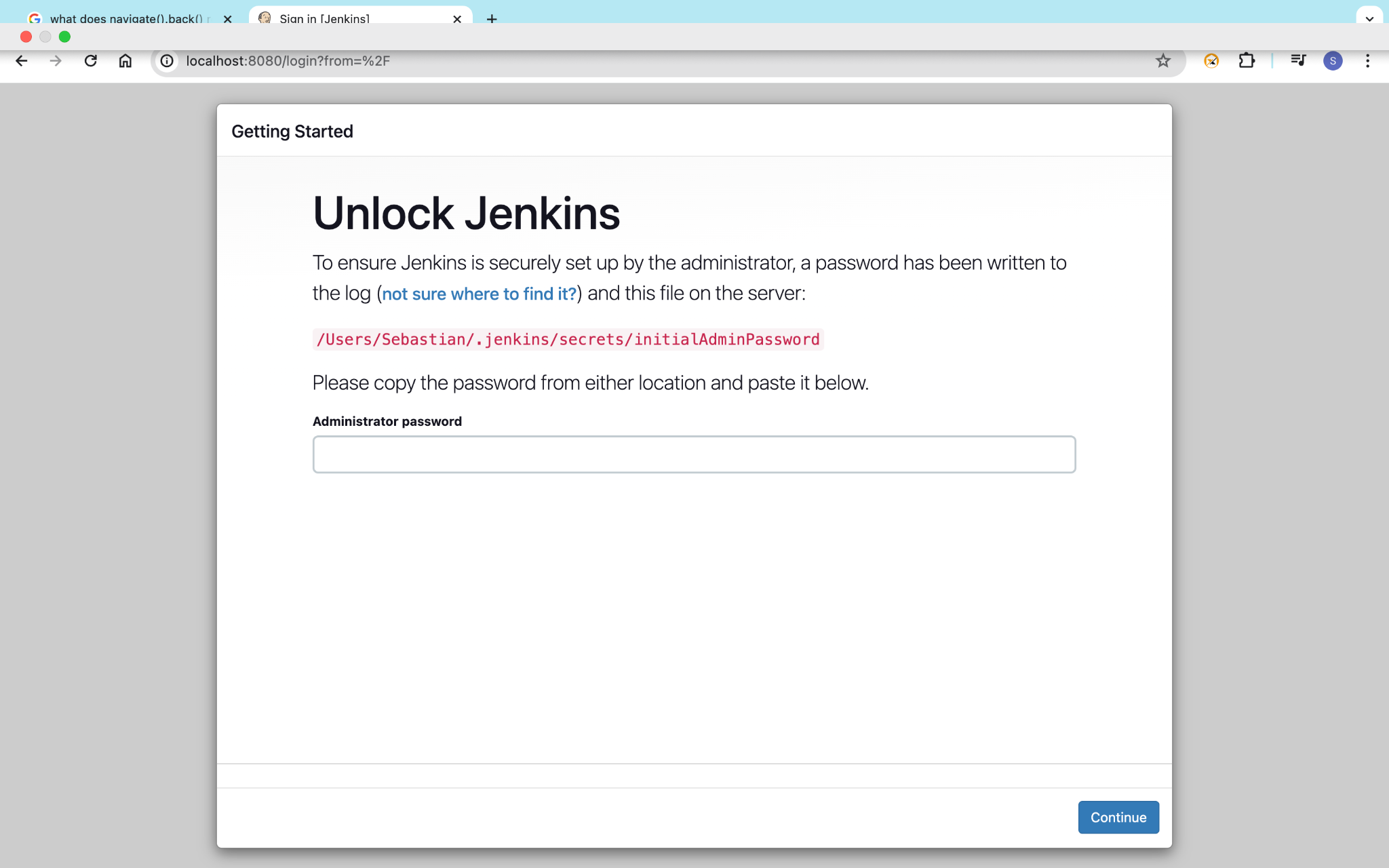
Java JDK

Maven

Go to the path where Jenkins war is installed

java -jar jenkins.war -httpPort=8080





<profiles>

<profile>

<id>webAppTest</id>

<build>

<pluginManagement>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-surefire-plugin</artifactId>

<version>3.5.2</version>

<configuration>

<suiteXmlFiles>

<suiteXmlFile>testng.xml</suiteXmlFile>

</suiteXmlFiles>

</configuration>

</plugin>

</plugins>

</pluginManagement>

</build>

</profiles>