**A yellow and black logo

AI-generated content may be incorrect.A cartoon of a person

AI-generated content may be incorrect.**

**🛠️**🛠️ Prerequisites

* JDK 17 and Maven 3.9.9 installed.
* Git installed on your system.
* Jenkins installed and running on http://localhost:8080.

1️⃣ Start Jenkins on Windows

1. Open Command Prompt and navigate to the directory where jenkins.war is located.
2. Run the following command:

bash

java -jar jenkins.war

Jenkins will start on http://localhost:8080. The first time you access it, you'll need to unlock Jenkins using the initial admin password, which can be found at:

bash

C:\Users\<YourUser>\.jenkins\secrets\initialAdminPassword

1. Follow the setup wizard to install suggested plugins and create an admin user.

2️⃣ Configure Tools in Jenkins

1. Go to Manage Jenkins > Global Tool Configuration.
2. Under JDK, click Add JDK:
   * Name: JDK-17
   * JAVA\_HOME: C:\Program Files\Java\jdk-17
3. Under Maven, click Add Maven:
   * Name: Maven-3.9.9
   * MAVEN\_HOME: C:\apache-maven-3.9.9
4. Under Git, ensure Git is installed or specify the path to your Git executable.

3️⃣ Install Required Plugins

1. Go to Manage Jenkins > Manage Plugins.
2. Under the Available tab, search for and install the following plugins:
   * Pipeline
   * Git
   * Maven Integration Plugin

4️⃣ Create a Sample Git Repository

For demonstration purposes, we'll use a simple Java Maven project hosted on GitHub.

1. Clone the repository to your local machine:

bash

git clone https://github.com/spring-projects/spring-petclinic.git

cd spring-petclinic

1. Ensure the project builds correctly:

bash

mvn clean install

5️⃣ Create a Jenkins Pipeline Job

1. In Jenkins, click New Item.
2. Enter a name for your job (e.g., SpringPetclinicPipeline).
3. Select Pipeline and click OK.
4. In the Pipeline section, set:
   * Definition: Pipeline script
   * Paste:

pipeline {

agent any

tools {

jdk 'jdk-17' // Make sure JDK 17 is configured in Jenkins

maven 'Maven-3.9.9' // Ensure Maven is installed in Jenkins global tools

}

environment {

MAVEN\_OPTS = "-Dmaven.test.failure.ignore=true"

}

stages {

stage('Checkout Code') {

steps {

git url: 'https://github.com/spring-projects/spring-petclinic.git', branch: 'main'

// If private repo:

// git credentialsId: 'github-creds', url: 'https://github.com/your-org/your-repo.git', branch: 'main'

}

}

stage('Build with Maven') {

steps {

sh 'mvn clean compile'

}

}

stage('Run Tests') {

steps {

sh 'mvn test'

}

}

stage('Archive Test Results') {

steps {

junit 'target/surefire-reports/\*.xml'

}

}

stage('Package App') {

steps {

sh 'mvn package'

archiveArtifacts artifacts: 'target/\*.jar', fingerprint: true

}

}

}

post {

success {

echo "✅ Build and tests succeeded."

}

failure {

echo "❌ Build or tests failed."

}

}

}

Click: Save

This pipeline script defines the following stages:

* Checkout: Clones the repository.
* Build: Compiles the project and skips tests.
* Test: Runs the unit tests.
* Deploy: Placeholder for deployment steps.

7️⃣ Trigger the Pipeline

1. In Jenkins, navigate to your pipeline job.
2. Click Build Now to trigger the pipeline.
3. Monitor the build progress in the Build History section.

✅ Conclusion

You've successfully set up a Jenkins CI pipeline on Windows using JDK 17 and Maven 3.9.9. This pipeline automates the process of building, testing, and deploying a Java application from a Git repository.