Project report on

CI/CD Deployment for Springboot Application

This document contains sections for:

- Sprint planning and Task completion
- Core concepts used in project.
- Technologies used in project.
- Flow of the Application.
- Demonstrating the product capabilities, appearance, and user interactions.

The code for this project is hosted at

https://github.com/surekhaitgithub/Newcodingboard.git

The project is developed by Duggasani Naga Surekha.

Sprints planning and Task completion:

The project is planned to be completed in 2 sprint. Tasks assumed to be completed in the sprints are:

- Creating the flow of the application
- Initializing git repository to track changes as development progresses.

- Writing the program to fulfill the requirements of the project.
- Testing the program with different kinds of User input
- Pushing code to GitHub.
- Creating this specification document highlighting application capabilities, appearance, and user interactions.

Core concepts used in project:

• Deployment: to deploy the local project to the end-users.

• Virtual Machine: use virtual instances to help to build, deploy and manage websites.

• Exception Handling: used to catch problems that arises in the code especially in I/O blocks. • Single Web Page: apply the concept of a website that only contains one HTML page.

• Object-Oriented: used to create and model objects for users and their credentials.

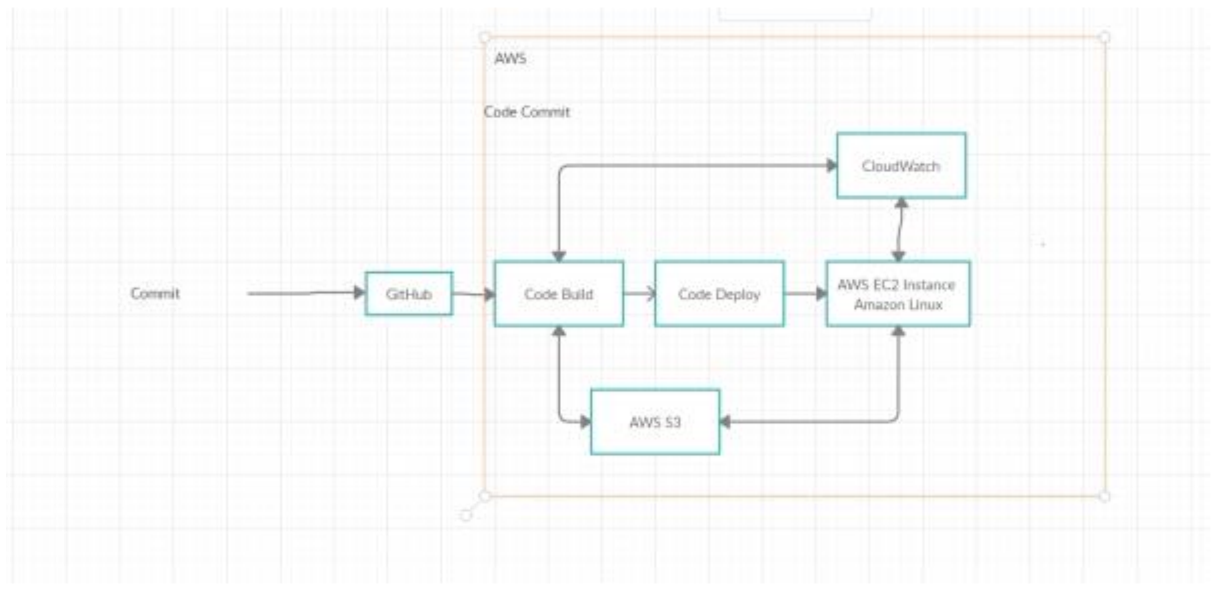
Technologies Used:

• AWS EC2 instance: to use the instances as a VM and deploy the application

• Jenkins: to build the project from GitHub.

• GitHub: to upload the source code of the project.

Flow of the Application:



Project Users Stories : (Agile and Scrum)

- As a user I want an automated integration of a Spring boot Application.
- As a user I want an automated deployment of a Spring boot Application.
- As a developer I want to automate the integration of a Spring boot Application for the user.
- As a developer I want to automate the deployment of a Spring boot Application for the user.

1. <u>SPRINTS</u>

Sprint 1

- Understanding the problem statement of the project .
- Creating the flow chart of the project.
- Creating Maven Project.
- Creating Spring boot Application.
- Adding necessary dependencies.
- Testing at each step for different user inputs.
- Initializing the git repository.
- Pushing the code to the GitHub.
- Creating AWS EC2 instance.
- Downloading MobaXterm.
- Downloading Jenkins.
- Deploying the application on Jenkins.
- Creating the Specification document for deploying the project.

Demonstrating the product capabilities, appearance, and user interactions:

To demonstrate the product capabilities, below are the sub-sections configured to highlight appearance and user interactions for the project:

Step 1: Creating a new project in Eclipse

- Open Eclipse
- Go to File -> New -> Project -> Maven Project -> Next.
- Type in any project name and click on "Finish."

Files:

Src/main/java/com/SpringTest/SpringApplication.java:

```
package com.SpringTest;


import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;


@SpringBootApplication

public class SpringJenkinsApplication {


	public static  Logger log = LoggerFactory.getLogger(SpringJenkinsApplication.class);


	public void init() {

		log.info("Spring Boot Application Started.......");


	}


	public static void main(String[] args) {
```

```
                                log.info("Application Executed .......");

                     SpringApplication.run(SpringJenkinsApplication.class, args);


          }

}
```

Src/test/java/com/SpringTest/SpringAplicationTest.java:

```
package com.SpringTest;




import org.junit.jupiter.api.Test;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.test.context.SpringBootTest;


@SpringBootTest

class SpringJenkinsApplicationTests {


          public static  Logger log = LoggerFactory.getLogger(SpringJenkinsApplication.class);
```

```
	@Test

	void contextLoads() {


		log.info("Spring Test Case Executing......");

	}


}
```

META-INF/maven/com.SpringTest/Testing-Spring-Jenkins/pom.properties:

```
#Generated by Maven Integration for Eclipse

#Tue May 10 13:03:45 IST 2022

m2e.projectLocation=C\:\\Users\\Surekha\\Desktop\\phase 5 project\\CI-CD-Deployment-for-Springboot-Application

m2e.projectName=Spring-Jenkins

groupId=com.SpringTest

artifactId=Testing-Spring-Jenkins

version=0.0.1-SNAPSHOT
```

META-INF/maven/com.SpringTest/Testing-Spring-Jenkins/pom.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
```

```xml
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

	xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">

	<modelVersion>4.0.0</modelVersion>

	<parent>

		<groupId>org.springframework.boot</groupId>

		<artifactId>spring-boot-starter-parent</artifactId>

		<version>2.5.4</version>

		<relativePath/> <!-- lookup parent from repository -->

	</parent>

	<groupId>com.SpringTest</groupId>

	<artifactId>Testing-Spring-Jenkins</artifactId>

	<version>0.0.1-SNAPSHOT</version>

	<name>Spring-Jenkins</name>

	<description> Spring Boot -Jenkins</description>

	<properties>

		<java.version>11</java.version>

	</properties>

	<dependencies>

		<dependency>

			<groupId>org.springframework.boot</groupId>

			<artifactId>spring-boot-starter-thymeleaf</artifactId>
```

```
</dependency>
<dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-web</artifactId>
</dependency>

<dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-devtools</artifactId>
        <scope>runtime</scope>
        <optional>true</optional>
</dependency>
<dependency>
        <groupId>mysql</groupId>
        <artifactId>mysql-connector-java</artifactId>
        <scope>runtime</scope>
</dependency>
<dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-test</artifactId>
        <scope>test</scope>
</dependency>
```

```
	</dependencies>


	<build>
		<plugins>
			<plugin>
				<groupId>org.springframework.boot</groupId>
				<artifactId>spring-boot-maven-plugin</artifactId>
			</plugin>
		</plugins>
	</build>

</project>
```

MANIFEST.MF:

```
Manifest-Version: 1.0
Build-Jdk-Spec: 13
Implementation-Title: Spring-Jenkins
Implementation-Version: 0.0.1-SNAPSHOT
Created-By: Maven Integration for Eclipse
```

Maven-archiver/pom.properties:

```
artifactId=Testing-Spring-Jenkins

groupId=com.SpringTest

version=0.0.1-SNAPSHOT
```

mvnw:

```
#!/bin/sh

# ----------------------------------------------------------------------------

# Licensed to the Apache Software Foundation (ASF) under one

# or more contributor license agreements.  See the NOTICE file

# distributed with this work for additional information

# regarding copyright ownership.  The ASF licenses this file

# to you under the Apache License, Version 2.0 (the

# "License"); you may not use this file except in compliance

# with the License.  You may obtain a copy of the License at

#

#    https://www.apache.org/licenses/LICENSE-2.0

#

# Unless required by applicable law or agreed to in writing,

# software distributed under the License is distributed on an

# "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY

# KIND, either express or implied.  See the License for the
# specific language governing permissions and limitations
# under the License.
# ----------------------------------------------------------------------------

# ----------------------------------------------------------------------------
# Maven Start Up Batch script
#
# Required ENV vars:
# ------------------
#   JAVA_HOME - location of a JDK home dir
#
# Optional ENV vars
# -----------------
#   M2_HOME - location of maven2's installed home dir
#   MAVEN_OPTS - parameters passed to the Java VM when running Maven
#     e.g. to debug Maven itself, use
#       set MAVEN_OPTS=-Xdebug -
Xrunjdwp:transport=dt_socket,server=y,suspend=y,address=8000
#   MAVEN_SKIP_RC - flag to disable loading of mavenrc files
# ----------------------------------------------------------------------------

if [ -z "$MAVEN_SKIP_RC" ] ; then
```

```
  if [ -f /etc/mavenrc ] ; then
    . /etc/mavenrc
  fi

  if [ -f "$HOME/.mavenrc" ] ; then
    . "$HOME/.mavenrc"
  fi

fi

# OS specific support.  $var _must_ be set to either true or false.
cygwin=false;
darwin=false;
mingw=false
case "`uname`" in
  CYGWIN*) cygwin=true ;;
  MINGW*) mingw=true;;
  Darwin*) darwin=true
    # Use /usr/libexec/java_home if available, otherwise fall back to /Library/Java/Home
    # See https://developer.apple.com/library/mac/qa/qa1170/_index.html
    if [ -z "$JAVA_HOME" ]; then
```

```
      if [ -x "/usr/libexec/java_home" ]; then
        export JAVA_HOME="`/usr/libexec/java_home`"
      else
        export JAVA_HOME="/Library/Java/Home"
      fi
    fi
    ;;
esac

if [ -z "$JAVA_HOME" ] ; then
  if [ -r /etc/gentoo-release ] ; then
    JAVA_HOME=`java-config --jre-home`
  fi
fi

if [ -z "$M2_HOME" ] ; then
  ## resolve links - $0 may be a link to maven's home
  PRG="$0"

  # need this for relative symlinks
  while [ -h "$PRG" ] ; do
    ls=`ls -ld "$PRG"`
    link=`expr "$ls" : '.*-> \(.*\)$'`
    if expr "$link" : '/.*' > /dev/null; then
      PRG="$link"
    else
      PRG="`dirname "$PRG"`/$link"
    fi
  done

  saveddir=`pwd`

  M2_HOME=`dirname "$PRG"`/..

  # make it fully qualified
  M2_HOME=`cd "$M2_HOME" && pwd`

  cd "$saveddir"
  # echo Using m2 at $M2_HOME
fi

# For Cygwin, ensure paths are in UNIX format before anything is touched
if $cygwin ; then
  [ -n "$M2_HOME" ] &&
    M2_HOME=`cygpath --unix "$M2_HOME"`
  [ -n "$JAVA_HOME" ] &&
    JAVA_HOME=`cygpath --unix "$JAVA_HOME"`
  [ -n "$CLASSPATH" ] &&
    CLASSPATH=`cygpath --path --unix "$CLASSPATH"`
fi

# For Mingw, ensure paths are in UNIX format before anything is touched
if $mingw ; then
  [ -n "$M2_HOME" ] &&
    M2_HOME="`(cd "$M2_HOME"; pwd)`"
  [ -n "$JAVA_HOME" ] &&
    JAVA_HOME="`(cd "$JAVA_HOME"; pwd)`"
fi

if [ -z "$JAVA_HOME" ]; then
  javaExecutable="`which javac`"
  if [ -n "$javaExecutable" ] && ! [ "`expr \"$javaExecutable\" : '\([^ ]*\)'`" = "no" ]; then
    # readlink(1) is not available as standard on Solaris 10.
    readLink=`which readlink`
    if [ ! `expr "$readLink" : '\([^ ]*\)'` = "no" ]; then
      if $darwin ; then
```

```
        javaHome="`dirname \"$javaExecutable\"`"
        javaExecutable="`cd \"$javaHome\" && pwd -P`/javac"
      else
        javaExecutable="`readlink -f \"$javaExecutable\"`"
      fi
      javaHome="`dirname \"$javaExecutable\"`"
      javaHome=`expr "$javaHome" : '\(.*\)/bin'`
      JAVA_HOME="$javaHome"
      export JAVA_HOME
    fi
  fi
fi

if [ -z "$JAVACMD" ] ; then
  if [ -n "$JAVA_HOME"  ] ; then
    if [ -x "$JAVA_HOME/jre/sh/java" ] ; then
      # IBM's JDK on AIX uses strange locations for the executables
      JAVACMD="$JAVA_HOME/jre/sh/java"
    else
      JAVACMD="$JAVA_HOME/bin/java"
    fi
  else
```

```
    JAVACMD="`which java`"
  fi
fi

if [ ! -x "$JAVACMD" ] ; then
  echo "Error: JAVA_HOME is not defined correctly." >&2
  echo "  We cannot execute $JAVACMD" >&2
  exit 1
fi

if [ -z "$JAVA_HOME" ] ; then
  echo "Warning: JAVA_HOME environment variable is not set."
fi

CLASSWORLDS_LAUNCHER=org.codehaus.plexus.classworlds.launcher.Launcher

# traverses directory structure from process work directory to filesystem root
# first directory with .mvn subdirectory is considered project base directory
find_maven_basedir() {

  if [ -z "$1" ]
  then
```

```
    echo "Path not specified to find_maven_basedir"
    return 1
  fi

  basedir="$1"
  wdir="$1"
  while [ "$wdir" != '/' ] ; do
    if [ -d "$wdir"/.mvn ] ; then
      basedir=$wdir
      break
    fi
    # workaround for JBEAP-8937 (on Solaris 10/Sparc)
    if [ -d "${wdir}" ]; then
      wdir=`cd "$wdir/.."; pwd`
    fi
    # end of workaround
  done
  echo "${basedir}"
}

# concatenates all lines of a file
concat_lines() {
  if [ -f "$1" ]; then
    echo "$(tr -s '\n' ' ' < "$1")"
  fi
}

BASE_DIR=`find_maven_basedir "$(pwd)"`
if [ -z "$BASE_DIR" ]; then
  exit 1;
fi

##########################################################################################
# Extension to allow automatically downloading the maven-wrapper.jar from Maven-central
# This allows using the maven wrapper in projects that prohibit checking in binary data.
##########################################################################################
if [ -r "$BASE_DIR/.mvn/wrapper/maven-wrapper.jar" ]; then
    if [ "$MVNW_VERBOSE" = true ]; then
      echo "Found .mvn/wrapper/maven-wrapper.jar"
    fi
else
    if [ "$MVNW_VERBOSE" = true ]; then
      echo "Couldn't find .mvn/wrapper/maven-wrapper.jar, downloading it ..."
    fi
    if [ -n "$MVNW_REPOURL" ]; then
      jarUrl="$MVNW_REPOURL/io/takari/maven-wrapper/0.5.6/maven-wrapper-0.5.6.jar"
    else
      jarUrl="https://repo.maven.apache.org/maven2/io/takari/maven-wrapper/0.5.6/maven-
wrapper-0.5.6.jar"
    fi
    while IFS="=" read key value; do
      case "$key" in (wrapperUrl) jarUrl="$value"; break ;;
      esac
    done < "$BASE_DIR/.mvn/wrapper/maven-wrapper.properties"
    if [ "$MVNW_VERBOSE" = true ]; then
      echo "Downloading from: $jarUrl"
    fi
    wrapperJarPath="$BASE_DIR/.mvn/wrapper/maven-wrapper.jar"
    if $cygwin; then
      wrapperJarPath=`cygpath --path --windows "$wrapperJarPath"`
    fi

    if command -v wget > /dev/null; then
        if [ "$MVNW_VERBOSE" = true ]; then
          echo "Found wget ... using wget"
        fi
        if [ -z "$MVNW_USERNAME" ] || [ -z "$MVNW_PASSWORD" ]; then
            wget "$jarUrl" -O "$wrapperJarPath"
        else
            wget --http-user=$MVNW_USERNAME --http-password=$MVNW_PASSWORD
"$jarUrl" -O "$wrapperJarPath"
        fi
    elif command -v curl > /dev/null; then
        if [ "$MVNW_VERBOSE" = true ]; then
          echo "Found curl ... using curl"
        fi
        if [ -z "$MVNW_USERNAME" ] || [ -z "$MVNW_PASSWORD" ]; then
            curl -o "$wrapperJarPath" "$jarUrl" -f
        else
            curl --user $MVNW_USERNAME:$MVNW_PASSWORD -o "$wrapperJarPath"
"$jarUrl" -f
        fi

    else
        if [ "$MVNW_VERBOSE" = true ]; then
          echo "Falling back to using Java to download"
        fi
        javaClass="$BASE_DIR/.mvn/wrapper/MavenWrapperDownloader.java"
        # For Cygwin, switch paths to Windows format before running javac
        if $cygwin; then
          javaClass=`cygpath --path --windows "$javaClass"`
        fi
        if [ -e "$javaClass" ]; then
            if [ ! -e "$BASE_DIR/.mvn/wrapper/MavenWrapperDownloader.class" ]; then
              if [ "$MVNW_VERBOSE" = true ]; then
                echo " - Compiling MavenWrapperDownloader.java ..."
              fi
              # Compiling the Java class
              ("$JAVA_HOME/bin/javac" "$javaClass")
            fi
            if [ -e "$BASE_DIR/.mvn/wrapper/MavenWrapperDownloader.class" ]; then
              # Running the downloader
              if [ "$MVNW_VERBOSE" = true ]; then
                echo " - Running MavenWrapperDownloader.java ..."
              fi
              ("$JAVA_HOME/bin/java" -cp .mvn/wrapper MavenWrapperDownloader
"$MAVEN_PROJECTBASEDIR")
            fi
        fi
    fi
```

```
fi

##########################################################################################

# End of extension

##########################################################################################


export MAVEN_PROJECTBASEDIR=${MAVEN_BASEDIR:-"$BASE_DIR"}

if [ "$MVNW_VERBOSE" = true ]; then

  echo $MAVEN_PROJECTBASEDIR

fi

MAVEN_OPTS="$(concat_lines "$MAVEN_PROJECTBASEDIR/.mvn/jvm.config") $MAVEN_OPTS"


# For Cygwin, switch paths to Windows format before running java

if $cygwin; then

  [ -n "$M2_HOME" ] &&

    M2_HOME=`cygpath --path --windows "$M2_HOME"`

  [ -n "$JAVA_HOME" ] &&

    JAVA_HOME=`cygpath --path --windows "$JAVA_HOME"`

  [ -n "$CLASSPATH" ] &&

    CLASSPATH=`cygpath --path --windows "$CLASSPATH"`

  [ -n "$MAVEN_PROJECTBASEDIR" ] &&
    MAVEN_PROJECTBASEDIR=`cygpath --path --windows
"$MAVEN_PROJECTBASEDIR"`

fi



# Provide a "standardized" way to retrieve the CLI args that will

# work with both Windows and non-Windows executions.

MAVEN_CMD_LINE_ARGS="$MAVEN_CONFIG $@"

export MAVEN_CMD_LINE_ARGS



WRAPPER_LAUNCHER=org.apache.maven.wrapper.MavenWrapperMain



exec "$JAVACMD" \

  $MAVEN_OPTS \

  -classpath "$MAVEN_PROJECTBASEDIR/.mvn/wrapper/maven-wrapper.jar" \

  "-Dmaven.home=${M2_HOME}" "-
Dmaven.multiModuleProjectDirectory=${MAVEN_PROJECTBASEDIR}" \

  ${WRAPPER_LAUNCHER} $MAVEN_CONFIG "$@"
```

Pom.xml:

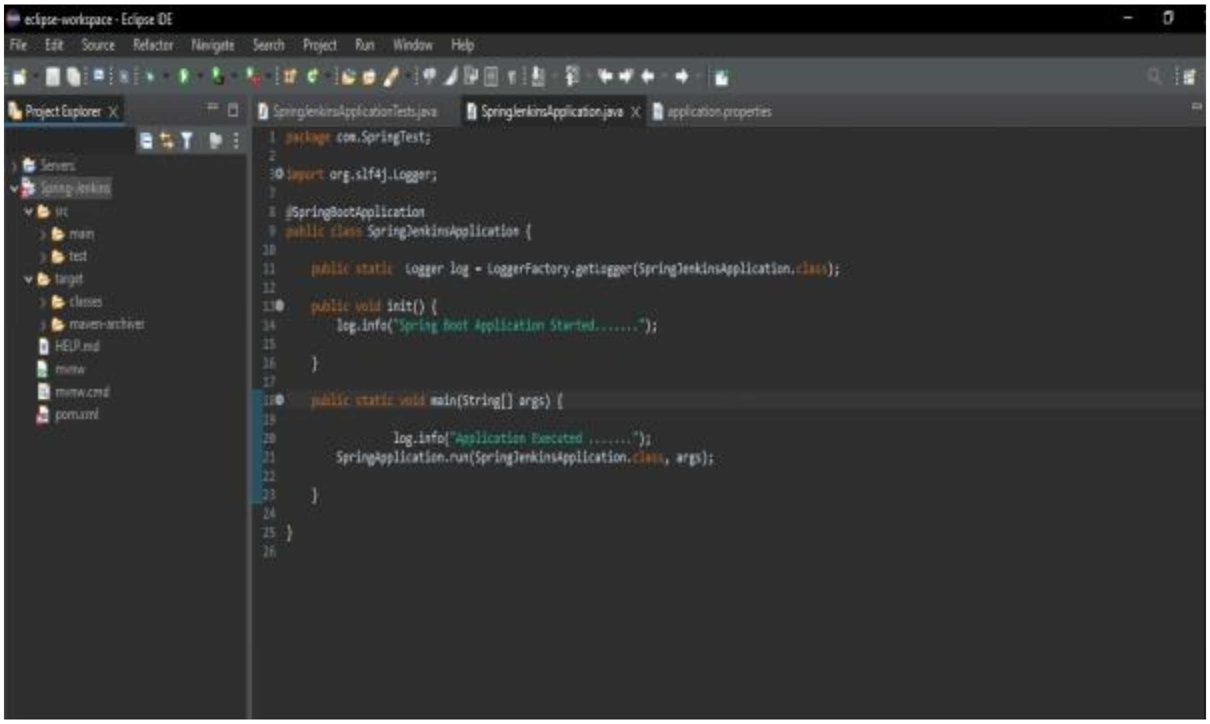
```
<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
	xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">

	<modelVersion>4.0.0</modelVersion>

	<parent>

		<groupId>org.springframework.boot</groupId>

		<artifactId>spring-boot-starter-parent</artifactId>

		<version>2.5.4</version>

		<relativePath/> <!-- lookup parent from repository -->

	</parent>

	<groupId>com.SpringTest</groupId>

	<artifactId>Testing-Spring-Jenkins</artifactId>

	<version>0.0.1-SNAPSHOT</version>

	<name>Spring-Jenkins</name>

	<description> Spring Boot -Jenkins</description>

	<properties>

		<java.version>11</java.version>

	</properties>

	<dependencies>

		<dependency>

			<groupId>org.springframework.boot</groupId>

			<artifactId>spring-boot-starter-thymeleaf</artifactId>

		</dependency>
```

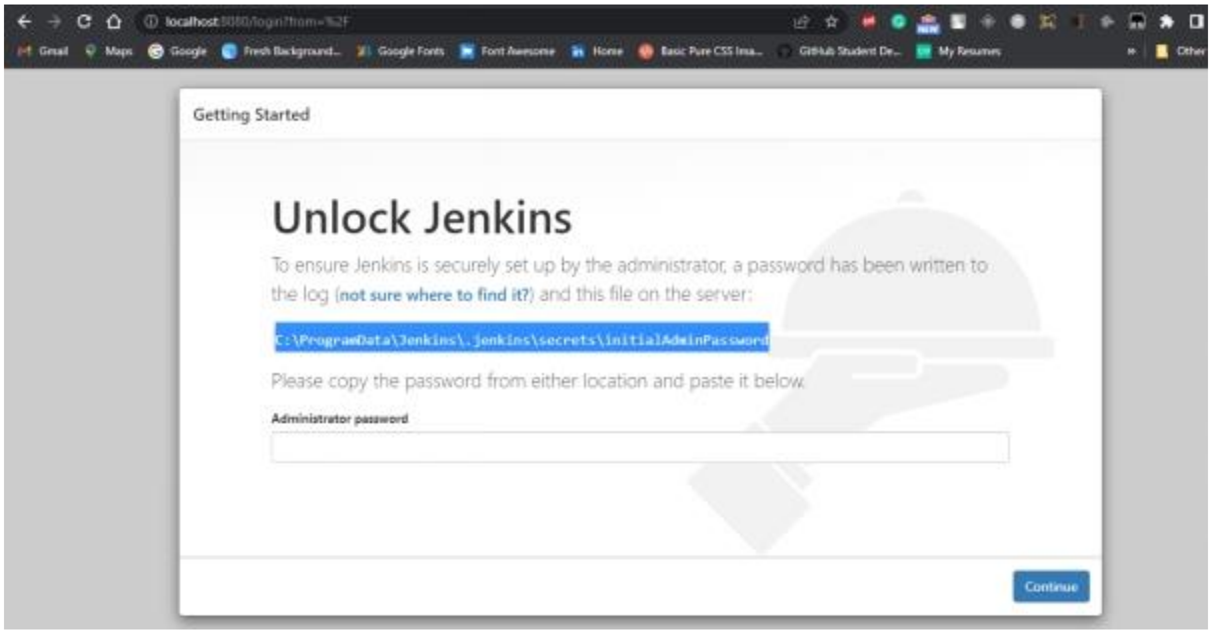
```
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>

        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-devtools</artifactId>
            <scope>runtime</scope>
            <optional>true</optional>
        </dependency>
        <dependency>
            <groupId>mysql</groupId>
            <artifactId>mysql-connector-java</artifactId>
            <scope>runtime</scope>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-test</artifactId>
            <scope>test</scope>
        </dependency>
    </dependencies>
    <build>

        <plugins>

            <plugin>

                <groupId>org.springframework.boot</groupId>

                <artifactId>spring-boot-maven-plugin</artifactId>

            </plugin>

        </plugins>

    </build>


</project>
```

OUTPUTS:

Eclipse


eclipse-workspace - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Project Explorer
Servers
Spring-Jenkins
src
main
test
target
classes
maven-archiver
HELP.md
mvnw
mvnw.cmd
pom.xml
SpringJenkinsApplicationTests.java
SpringJenkinsApplication.java
application.properties

```
package com.SpringTest;

import org.slf4j.Logger;

@SpringBootApplication
public class SpringJenkinsApplication {

    public static  Logger log = LoggerFactory.getLogger(SpringJenkinsApplication.class);

    public void init() {
        log.info("Spring Boot Application Started.......");

    }

    public static void main(String[] args) {

                log.info("Application Executed .......");
        SpringApplication.run(SpringJenkinsApplication.class, args);

    }

}
```



Unlock Jenkins

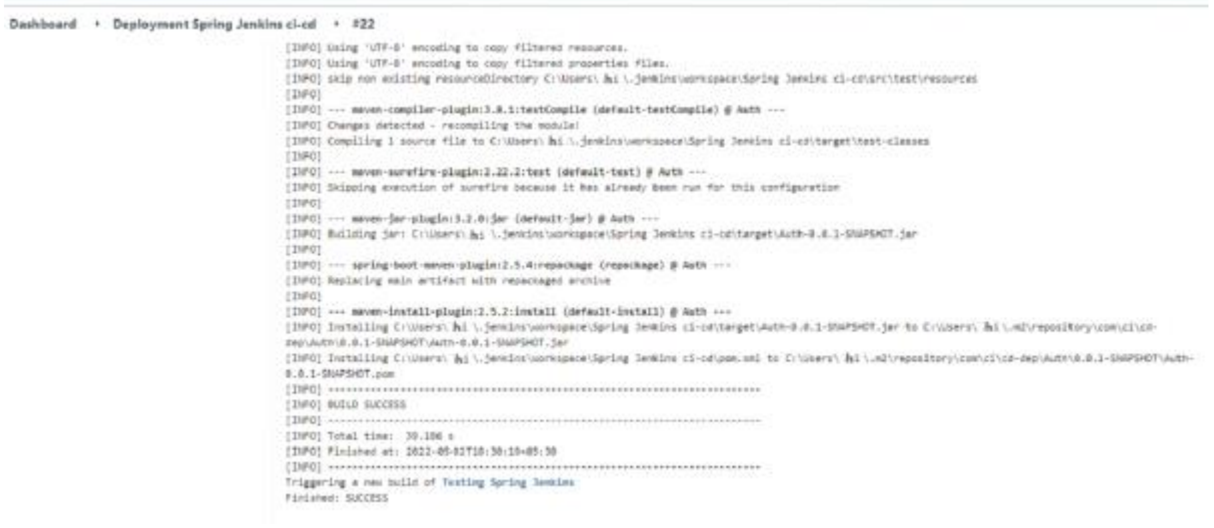
To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:

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

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

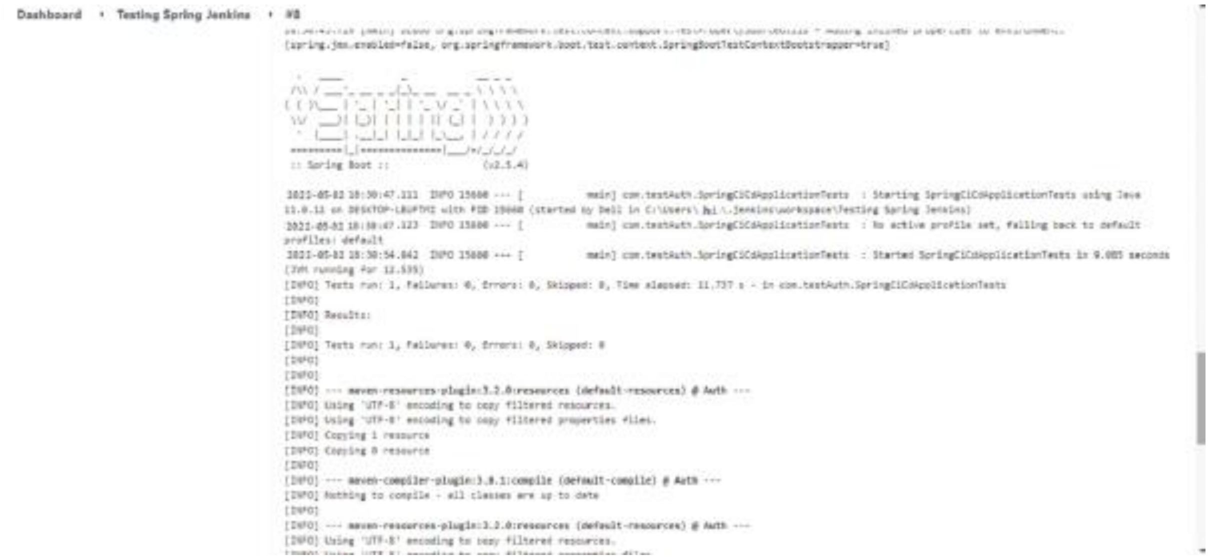
Administrator password

1. I create Simple Spring boot Application to build a CI/CD pipeline to demonstrate continuous deployment:



Dashboard › Deployment Spring Jenkins ci-cd › #22

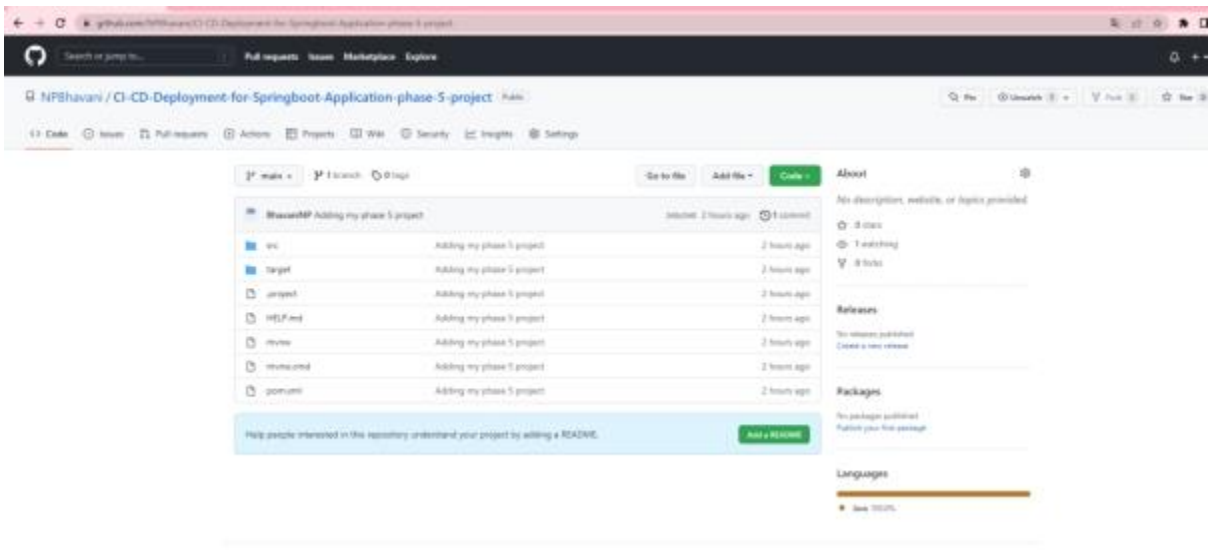
```
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] Using 'UTF-8' encoding to copy filtered properties files.
[INFO] skip non existing resourceDirectory C:\Users\ hi \.jenkins\workspace\Spring Jenkins ci-cd\src\test\resources
[INFO]
[INFO] --- maven-compiler-plugin:3.8.1:testCompile (default-testCompile) @ Auth ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 1 source file to C:\Users\ hi \.jenkins\workspace\Spring Jenkins ci-cd\target\test-classes
[INFO]
[INFO] --- maven-surefire-plugin:2.22.2:test (default-test) @ Auth ---
[INFO] Skipping execution of surefire because it has already been run for this configuration
[INFO]
[INFO] --- maven-jar-plugin:3.2.0:jar (default-jar) @ Auth ---
[INFO] Building jar: C:\Users\ hi \.jenkins\workspace\Spring Jenkins ci-cd\target\Auth-0.0.1-SNAPSHOT.jar
[INFO]
[INFO] --- spring-boot-maven-plugin:2.5.4:repackage (repackage) @ Auth ---
[INFO] Replacing main artifact with repackaged archive
[INFO]
[INFO] --- maven-install-plugin:2.5.2:install (default-install) @ Auth ---
[INFO] Installing C:\Users\ hi \.jenkins\workspace\Spring Jenkins ci-cd\target\Auth-0.0.1-SNAPSHOT.jar to C:\Users\ hi \.m2\repository\com\ci\cd-
dep\Auth\0.0.1-SNAPSHOT\Auth-0.0.1-SNAPSHOT.jar
[INFO] Installing C:\Users\ hi \.jenkins\workspace\Spring Jenkins ci-cd\pom.xml to C:\Users\ hi \.m2\repository\com\ci\cd-dep\Auth\0.0.1-SNAPSHOT\Auth-
0.0.1-SNAPSHOT.pom
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  30.106 s
[INFO] Finished at: 2022-05-02T18:30:10+05:30
[INFO] ------------------------------------------------------------------------
Triggering a new build of Testing Spring Jenkins
Finished: SUCCESS
```



Dashboard › Testing Spring Jenkins › #8

```
[illegible]
[spring.jms.enabled=false, org.springframework.boot.test.context.SpringBootTestContextBootstrapper=true]

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::                (v2.5.4)

2022-05-02 18:30:47.111  INFO 15888 --- [           main] com.testAuth.SpringCiCdApplicationTests  : Starting SpringCiCdApplicationTests using Java
11.0.12 on DESKTOP-LBUPTMI with PID 15888 (started by Dell in C:\Users\ hi \.jenkins\workspace\Testing Spring Jenkins)
2022-05-02 18:30:47.123  INFO 15888 --- [           main] com.testAuth.SpringCiCdApplicationTests  : No active profile set, falling back to default
profiles: default
2022-05-02 18:30:54.842  INFO 15888 --- [           main] com.testAuth.SpringCiCdApplicationTests  : Started SpringCiCdApplicationTests in 9.085 seconds
(JVM running for 12.535)
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 11.737 s - in com.testAuth.SpringCiCdApplicationTests
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO]
[INFO] --- maven-resources-plugin:3.2.0:resources (default-resources) @ Auth ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] Using 'UTF-8' encoding to copy filtered properties files.
[INFO] Copying 1 resource
[INFO] Copying 0 resource
[INFO]
[INFO] --- maven-compiler-plugin:3.8.1:compile (default-compile) @ Auth ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-resources-plugin:3.2.0:resources (default-resources) @ Auth ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
```

Git Hub: Application Deploy on git Hub



NPBhavani / CI-CD-Deployment-for-Springboot-Application-phase-5-project Public

BhavaniNP Adding my phase 5 project 2 hours ago 1 commit

src	Adding my phase 5 project	2 hours ago
target	Adding my phase 5 project	2 hours ago
.project	Adding my phase 5 project	2 hours ago
HELP.md	Adding my phase 5 project	2 hours ago
mvnw	Adding my phase 5 project	2 hours ago
mvnw.cmd	Adding my phase 5 project	2 hours ago
pom.xml	Adding my phase 5 project	2 hours ago

Help people interested in this repository understand your project by adding a README. Add a README

About

No description, website, or topics provided.

0 stars
1 watching
0 forks

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

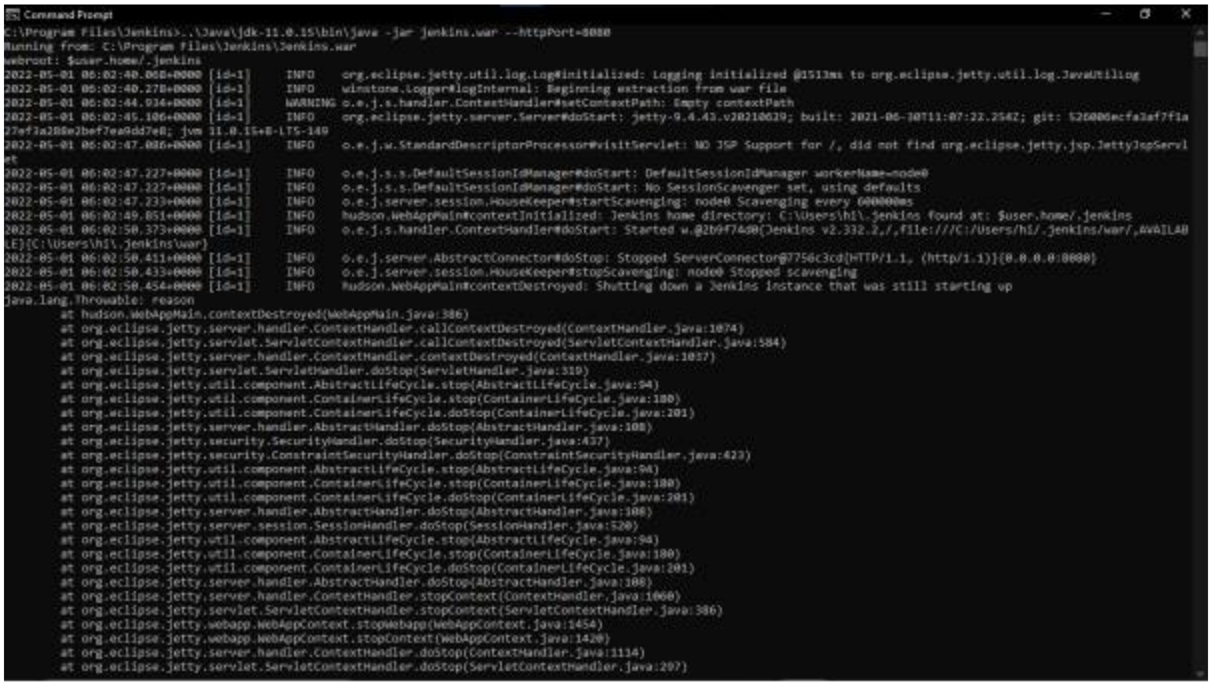
Languages

Java 100.0%

Github repository:

https://github.com/surekhaitgithub/Newcodingboard.git

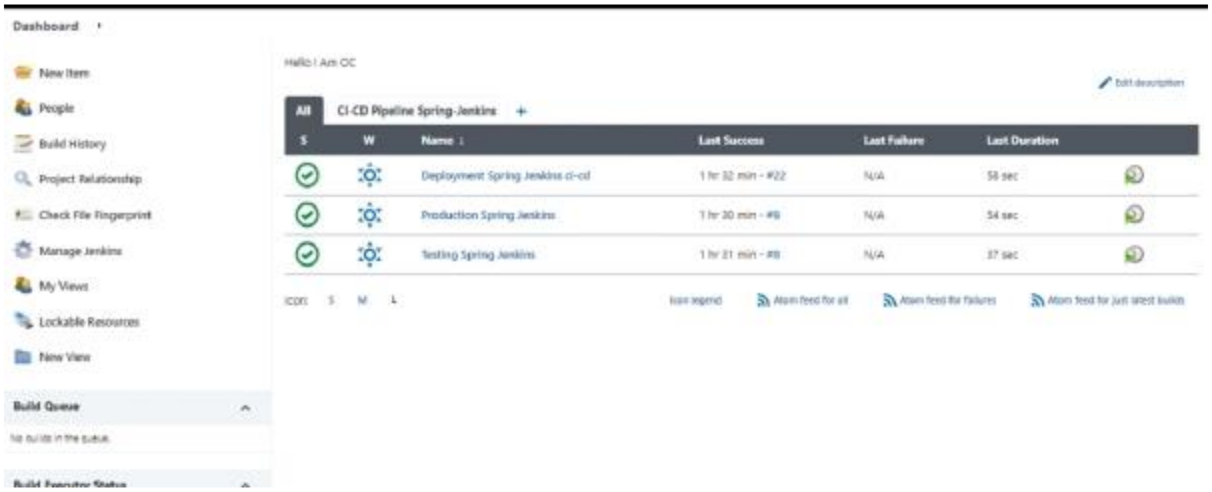
Jenkins : Run Jenkins



```
C:\Program Files\Jenkins>..\Java\jdk-11.0.15\bin\java -jar jenkins.war --httpPort=8080
Running from: C:\Program Files\Jenkins\Jenkins.war
webroot: $user.home/.jenkins
2022-05-01 06:02:40.068+0000 [id=1]     INFO    org.eclipse.jetty.util.log.Log#initialized: Logging initialized @1513ms to org.eclipse.jetty.util.log.JavaUtilLog
2022-05-01 06:02:40.278+0000 [id=1]     INFO    winstone.Logger#logInternal: Beginning extraction from war file
2022-05-01 06:02:44.934+0000 [id=1]     WARNING o.e.j.s.handler.ContextHandler#setContextPath: Empty contextPath
2022-05-01 06:02:45.106+0000 [id=1]     INFO    org.eclipse.jetty.server.Server#doStart: jetty-9.4.43.v20210629; built: 2021-06-30T11:07:22.254Z; git: 526006ecfa3af7f1a
27ef3a288e2bef7ea9dd7e8; jvm 11.0.15+8-LTS-149
2022-05-01 06:02:47.086+0000 [id=1]     INFO    o.e.j.w.StandardDescriptorProcessor#visitServlet: NO JSP Support for /, did not find org.eclipse.jetty.jsp.JettyJspServl
et
2022-05-01 06:02:47.227+0000 [id=1]     INFO    o.e.j.s.s.DefaultSessionIdManager#doStart: DefaultSessionIdManager workerName=node0
2022-05-01 06:02:47.227+0000 [id=1]     INFO    o.e.j.s.s.DefaultSessionIdManager#doStart: No SessionScavenger set, using defaults
2022-05-01 06:02:47.233+0000 [id=1]     INFO    o.e.j.server.session.HouseKeeper#startScavenging: node0 Scavenging every 600000ms
2022-05-01 06:02:49.851+0000 [id=1]     INFO    hudson.WebAppMain#contextInitialized: Jenkins home directory: C:\Users\hi\.jenkins found at: $user.home/.jenkins
2022-05-01 06:02:50.373+0000 [id=1]     INFO    o.e.j.s.handler.ContextHandler#doStart: Started w.@2b9f74d0{Jenkins v2.332.2,/,file:///C:/Users/hi/.jenkins/war/,AVAILAB
LE}{C:\Users\hi\.jenkins\war}
2022-05-01 06:02:50.411+0000 [id=1]     INFO    o.e.j.server.AbstractConnector#doStop: Stopped ServerConnector@7756c3cd{HTTP/1.1, (http/1.1)}{0.0.0.0:8080}
2022-05-01 06:02:50.433+0000 [id=1]     INFO    o.e.j.server.session.HouseKeeper#stopScavenging: node0 Stopped scavenging
2022-05-01 06:02:50.454+0000 [id=1]     INFO    hudson.WebAppMain#contextDestroyed: Shutting down a Jenkins instance that was still starting up
java.lang.Throwable: reason
        at hudson.WebAppMain.contextDestroyed(WebAppMain.java:386)
        at org.eclipse.jetty.server.handler.ContextHandler.callContextDestroyed(ContextHandler.java:1074)
        at org.eclipse.jetty.servlet.ServletContextHandler.callContextDestroyed(ServletContextHandler.java:584)
        at org.eclipse.jetty.server.handler.ContextHandler.contextDestroyed(ContextHandler.java:1037)
        at org.eclipse.jetty.servlet.ServletHandler.doStop(ServletHandler.java:319)
        at org.eclipse.jetty.util.component.AbstractLifeCycle.stop(AbstractLifeCycle.java:94)
        at org.eclipse.jetty.util.component.ContainerLifeCycle.stop(ContainerLifeCycle.java:180)
        at org.eclipse.jetty.util.component.ContainerLifeCycle.doStop(ContainerLifeCycle.java:201)
        at org.eclipse.jetty.server.handler.AbstractHandler.doStop(AbstractHandler.java:108)
        at org.eclipse.jetty.security.SecurityHandler.doStop(SecurityHandler.java:437)
        at org.eclipse.jetty.security.ConstraintSecurityHandler.doStop(ConstraintSecurityHandler.java:423)
        at org.eclipse.jetty.util.component.AbstractLifeCycle.stop(AbstractLifeCycle.java:94)
        at org.eclipse.jetty.util.component.ContainerLifeCycle.stop(ContainerLifeCycle.java:180)
        at org.eclipse.jetty.util.component.ContainerLifeCycle.doStop(ContainerLifeCycle.java:201)
        at org.eclipse.jetty.server.handler.AbstractHandler.doStop(AbstractHandler.java:108)
        at org.eclipse.jetty.server.session.SessionHandler.doStop(SessionHandler.java:520)
        at org.eclipse.jetty.util.component.AbstractLifeCycle.stop(AbstractLifeCycle.java:94)
        at org.eclipse.jetty.util.component.ContainerLifeCycle.stop(ContainerLifeCycle.java:180)
        at org.eclipse.jetty.util.component.ContainerLifeCycle.doStop(ContainerLifeCycle.java:201)
        at org.eclipse.jetty.server.handler.AbstractHandler.doStop(AbstractHandler.java:108)
        at org.eclipse.jetty.server.handler.ContextHandler.stopContext(ContextHandler.java:1060)
        at org.eclipse.jetty.servlet.ServletContextHandler.stopContext(ServletContextHandler.java:386)
        at org.eclipse.jetty.webapp.WebAppContext.stopWebapp(WebAppContext.java:1454)
        at org.eclipse.jetty.webapp.WebAppContext.stopContext(WebAppContext.java:1420)
        at org.eclipse.jetty.server.handler.ContextHandler.doStop(ContextHandler.java:1114)
        at org.eclipse.jetty.servlet.ServletContextHandler.doStop(ServletContextHandler.java:297)
```

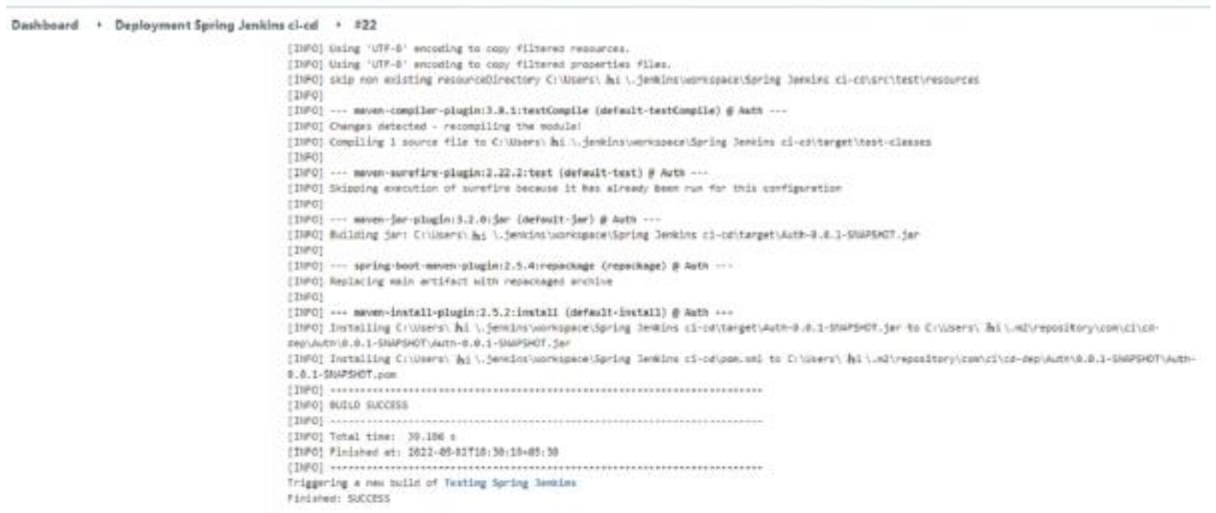
CREATE THREE JENKINS PROJECT DEPLOYMENT:-

DEPLOYMENT-TESTING-PRODUCTION



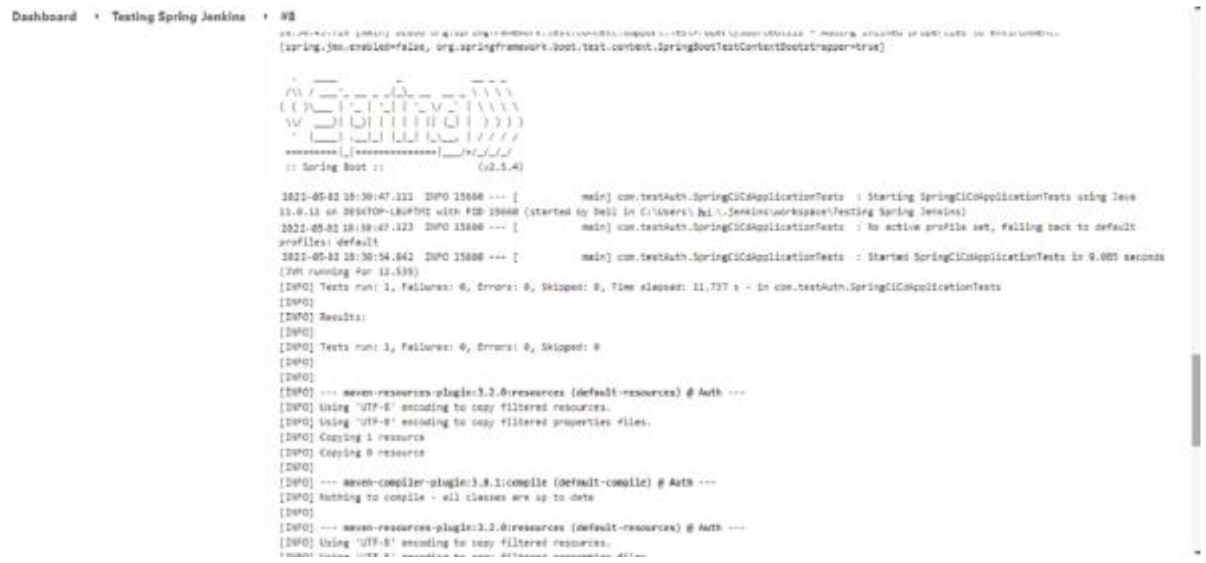
Hello I Am OC

S	W	Name ↓	Last Success	Last Failure	Last Duration
✓	☼	Deployment Spring Jenkins ci-cd	1 hr 32 min - #22	N/A	58 sec
✓	☼	Production Spring Jenkins	1 hr 30 min - #8	N/A	54 sec
✓	☼	Testing Spring Jenkins	1 hr 21 min - #8	N/A	37 sec



Dashboard › Deployment Spring Jenkins ci-cd › #22

```
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] Using 'UTF-8' encoding to copy filtered properties files.
[INFO] skip non existing resourceDirectory C:\Users\ hi \.jenkins\workspace\Spring Jenkins ci-cd\src\test\resources
[INFO]
[INFO] --- maven-compiler-plugin:3.8.1:testCompile (default-testCompile) @ Auth ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 1 source file to C:\Users\ hi \.jenkins\workspace\Spring Jenkins ci-cd\target\test-classes
[INFO]
[INFO] --- maven-surefire-plugin:2.22.2:test (default-test) @ Auth ---
[INFO] Skipping execution of surefire because it has already been run for this configuration
[INFO]
[INFO] --- maven-jar-plugin:3.2.0:jar (default-jar) @ Auth ---
[INFO] Building jar: C:\Users\ hi \.jenkins\workspace\Spring Jenkins ci-cd\target\Auth-0.0.1-SNAPSHOT.jar
[INFO]
[INFO] --- spring-boot-maven-plugin:2.5.4:repackage (repackage) @ Auth ---
[INFO] Replacing main artifact with repackaged archive
[INFO]
[INFO] --- maven-install-plugin:2.5.2:install (default-install) @ Auth ---
[INFO] Installing C:\Users\ hi \.jenkins\workspace\Spring Jenkins ci-cd\target\Auth-0.0.1-SNAPSHOT.jar to C:\Users\ hi \.m2\repository\com\ci\cd-
dep\Auth\0.0.1-SNAPSHOT\Auth-0.0.1-SNAPSHOT.jar
[INFO] Installing C:\Users\ hi \.jenkins\workspace\Spring Jenkins ci-cd\pom.xml to C:\Users\ hi \.m2\repository\com\ci\cd-dep\Auth\0.0.1-SNAPSHOT\Auth-
0.0.1-SNAPSHOT.pom
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  30.106 s
[INFO] Finished at: 2022-05-02T18:30:10+05:30
[INFO] ------------------------------------------------------------------------
Triggering a new build of Testing Spring Jenkins
Finished: SUCCESS
```

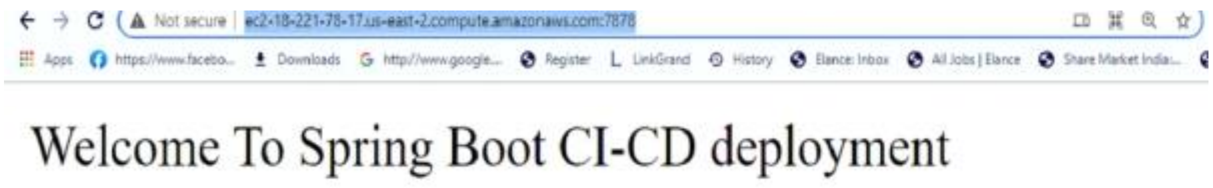


Dashboard › Testing Spring Jenkins › #8

```
[spring.jmx.enabled=false, org.springframework.boot.test.context.SpringBootTestContextBootstrapper=true]

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::                (v2.5.4)

2022-05-02 18:30:47.111  INFO 15080 --- [           main] com.testAuth.SpringCiCdApplicationTests  : Starting SpringCiCdApplicationTests using Java
11.0.12 on DESKTOP-LBUFTHI with PID 15080 (started by Dell in C:\Users\ hi \.jenkins\workspace\Testing Spring Jenkins)
2022-05-02 18:30:47.123  INFO 15080 --- [           main] com.testAuth.SpringCiCdApplicationTests  : No active profile set, falling back to default
profiles: default
2022-05-02 18:30:54.842  INFO 15080 --- [           main] com.testAuth.SpringCiCdApplicationTests  : Started SpringCiCdApplicationTests in 9.085 seconds
(JVM running for 12.535)
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 11.737 s - in com.testAuth.SpringCiCdApplicationTests
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO]
[INFO] --- maven-resources-plugin:3.2.0:resources (default-resources) @ Auth ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] Using 'UTF-8' encoding to copy filtered properties files.
[INFO] Copying 1 resource
[INFO] Copying 0 resource
[INFO]
[INFO] --- maven-compiler-plugin:3.8.1:compile (default-compile) @ Auth ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-resources-plugin:3.2.0:resources (default-resources) @ Auth ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
```

Final Output:-

Last host the application on AWS EC2 instance:



Welcome To Spring Boot CI-CD deployment

Step 2: Pushing the code to GitHub repository

- Open your command prompt and navigate to the folder where you have created your files.

  cd <folder path>

- Initialize repository using the following command:

  git init

- Add all the files to your git repository using the following command:

  git add .

- Commit the changes using the following command:

  git commit . -m <commit message>

- Push the files to the folder you initially created using the following command:

  git push -u origin master.