Sanya Ramchandani
T23 – 134

EXPERIMENT NO. 4

TO UNDERSTAND CONTINUOUS INTEGRATION, INSTALL AND CONFIGURE JENKINS WITH MAVEN / ANT / GRADLE TO SETUP A BUILD JOB

Theory:

1. Introduction to Continuous Integration (CI)

Continuous Integration (CI) is a software development practice where developers frequently integrate code changes into a shared repository. Each integration is automatically verified through a build and test process to detect errors early.

1.1 Benefits of Continuous Integration

- Early detection of defects.
- Faster debugging and deployment.
- Automated testing improves code quality.
- Reduces integration issues and conflicts.

1.2 Tools for Continuous Integration

- **Jenkins** – An open-source automation server.
- **Maven, Gradle, Ant** – Build tools to compile, test, and package applications.

---

2. Installing and Configuring Jenkins

2.1 Installing Jenkins

1. **Download Jenkins:**
    - Get Jenkins from Jenkins official site.
    - Install using the appropriate installer for your OS.
    - Start Jenkins using:
        - **Windows:** Run jenkins.exe
        - **Linux/Mac:** Run java -jar jenkins.war --httpPort=8080
2. **Initial Setup:**
    - Access Jenkins at http://localhost:8080/.
    - Enter the **Administrator password** (found in jenkins_home/secrets/initialAdminPassword).
    - Install suggested plugins and create an admin user.

2.2 Installing Required Plugins

- Navigate to **Manage Jenkins > Plugin Manager**.

Sanya Ramchandani
T23 – 134

- Install the following plugins:
    - **Maven Integration Plugin** (for Maven builds).
    - **Gradle Plugin** (for Gradle support).
    - **Pipeline Plugin** (for defining CI/CD pipelines).

---

3. Configuring Maven, Gradle, or Ant in Jenkins

3.1 Configuring Maven in Jenkins

1. **Install Maven on the system** (or use Jenkins' built-in Maven).
2. Go to **Manage Jenkins > Global Tool Configuration**.
3. Under **Maven**, click **Add Maven**, set a name (e.g., Maven-3.8.1), and provide the installation path.

3.2 Configuring Gradle in Jenkins

1. Install Gradle on your system.
2. In Jenkins, go to **Global Tool Configuration**.
3. Under **Gradle**, click **Add Gradle**, enter a name, and provide the installation path.

3.3 Configuring Ant in Jenkins

1. Install Apache Ant and set the environment variable (ANT_HOME).
2. In **Global Tool Configuration**, add Ant with its installation path.

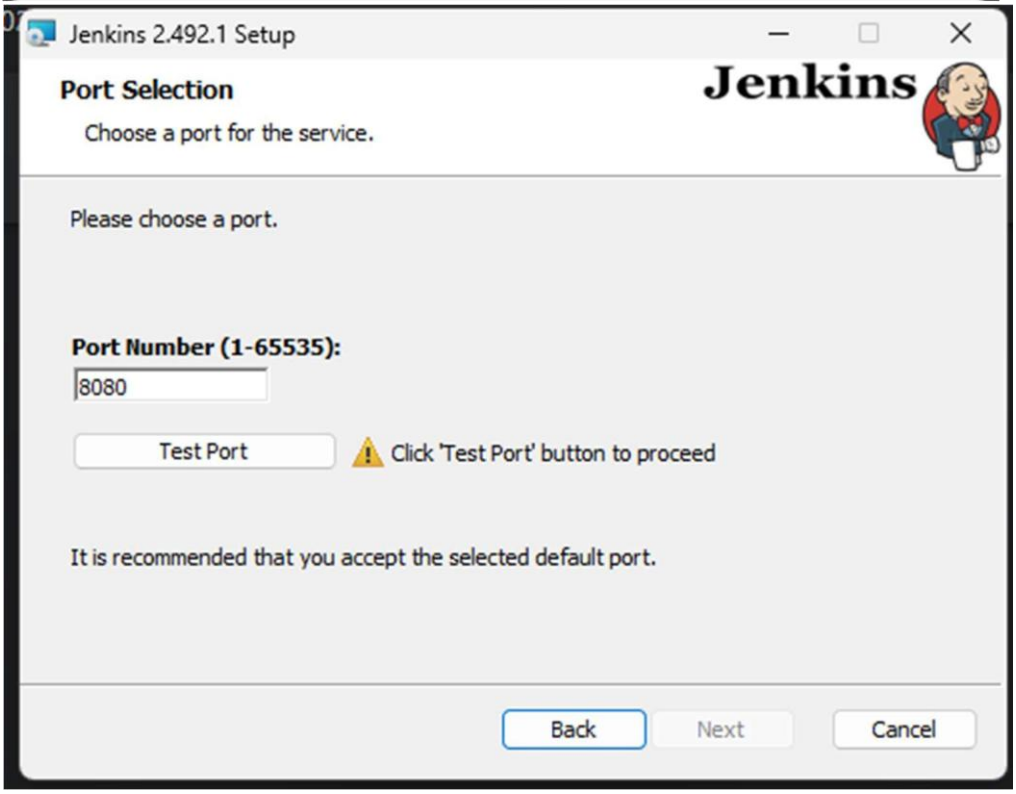
---

4. Creating a Build Job in Jenkins

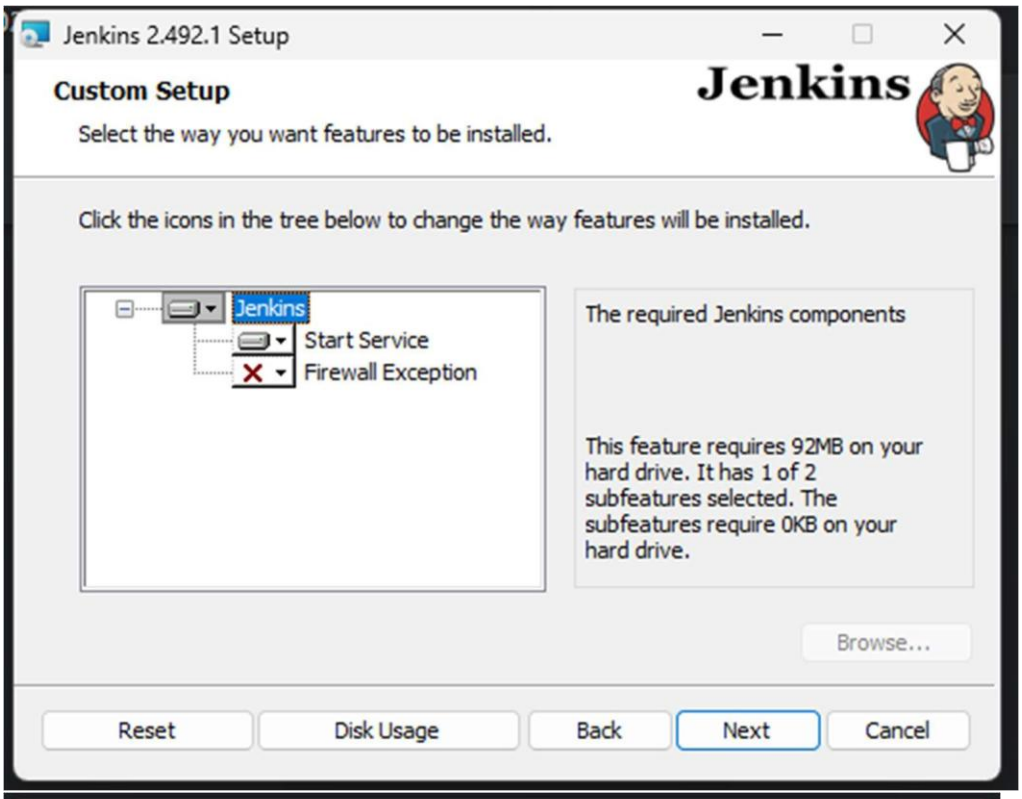
4.1 Steps to Set Up a Build Job

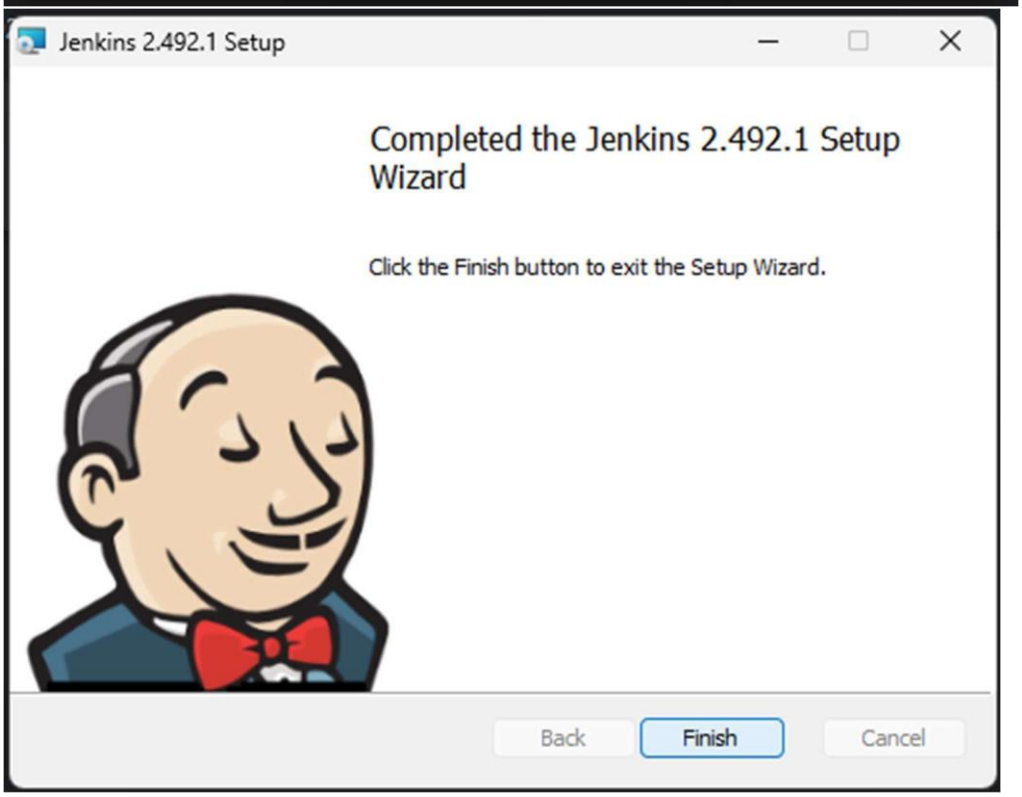
1. **Create a New Job**
    - Open Jenkins and click **New Item**.
    - Select **Freestyle Project**, enter a job name, and click **OK**.
2. **Configure Source Code Management**
    - Select **Git** and enter the repository URL.
    - Provide credentials if required.
3. **Configure Build Steps**
    - **For Maven:** Add a build step → **Invoke Maven Targets** → Enter clean package.
    - **For Gradle:** Add a build step → **Invoke Gradle Script** → Enter build.
    - **For Ant:** Add a build step → **Invoke Ant** → Enter build.xml.
4. **Save and Build the Job**
    - Click **Save**, then **Build Now** to execute.
    - Monitor logs in **Console Output**.

Sanya Ramchandani
T23 – 134

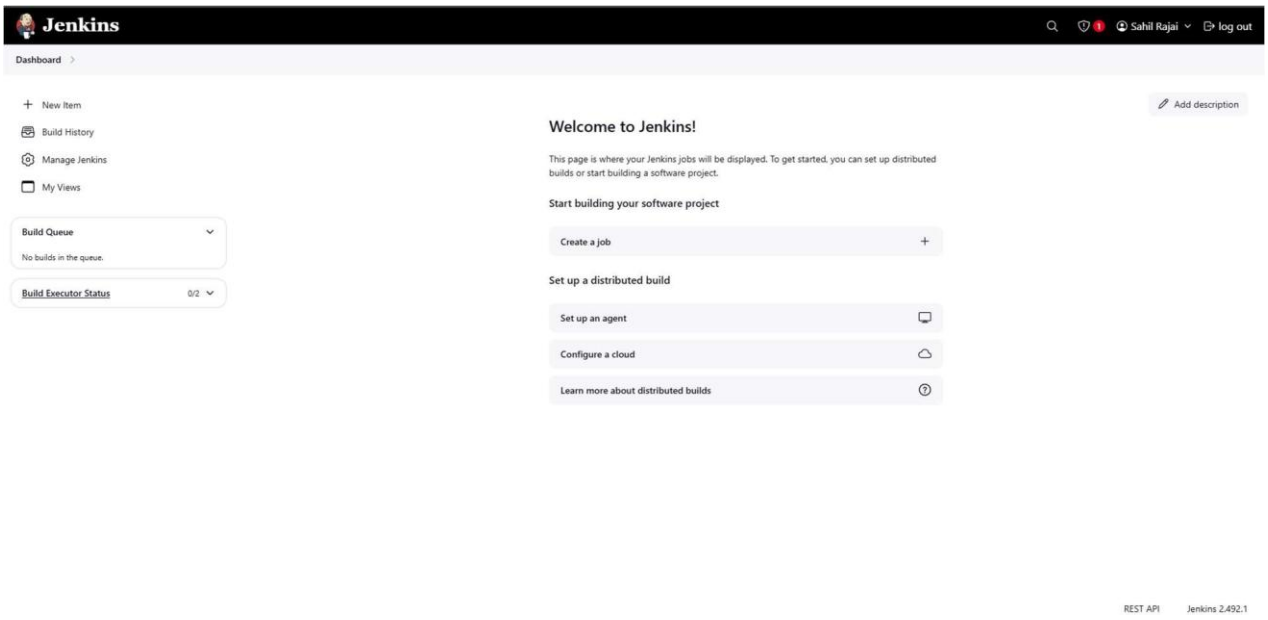


Sanya Ramchandani
T23 – 134





Sanya Ramchandani
T23 – 134



Conclusion

This experiment covered installing and configuring Jenkins for Continuous Integration, setting up build automation using Maven, Gradle, or Ant. By automating build jobs, we improve software quality, detect errors early, and enhance team collaboration.