? Jenkins Basic Project Documentation

1. Project Title:

Simple Jenkins CI/CD Pipeline for a Java Application

---

2. Objective:

To automate the build, test, and deployment process of a sample Java application using Jenkins.

---

3. Prerequisites:

Before starting, make sure the following are installed and configured:

- **Java (JDK 17)**
- **Git** (installed and configured)
- **Maven** (for building the Java project)
- **Jenkins** (installed and running on localhost:8080)
- **GitHub Account** (for storing the source code)

---

4. Project Flow:

Step-by-Step Process

1. Developer pushes code to GitHub repository.
2. Jenkins automatically detects the change via webhook.
3. Jenkins pulls the latest code and builds it using Maven.
4. Jenkins runs unit tests.
5. If tests pass, Jenkins deploys the artifact to a server or creates a `.war` file.

---

6. Steps to Create the Jenkins Project

Step 1: Create EC2 Instance & connect with SSH
Name - Jenkins

Step 2: Install Java & Jenkins

1: Update Your System

Update the package index to ensure all packages are up-to-date.

```
sudo apt update
```

2: Install Java

Jenkins requires Java (JDK). Install OpenJDK 21 (recommended version for recent Jenkins):

```
sudo apt install fontconfig openjdk-21-jre

java -version
```

3: Add Jenkins Repository Key & Install Jenkins

Add the Jenkins repository key to your system:

```
sudo wget -O /etc/apt/keyrings/jenkins-keyring.asc \
https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
echo "deb [signed-by=/etc/apt/keyrings/jenkins-keyring.asc]" \
https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
/etc/apt/sources.list.d/jenkins.list > /dev/null
sudo apt update
sudo apt install jenkins
```

4: Start and Enable Jenkins Service

Start Jenkins service and enable it to start at boot:

```
sudo systemctl enable jenkins
sudo systemctl start jenkins
```

Check Jenkins status:

```
sudo systemctl status jenkins
```

You should see `active (running)`.

Allow Jenkins port 8080 through the firewall:

```
sudo ufw allow 8080
sudo ufw status
```

Open your browser and go to:

```
http://<your_server_ip>:8080
# Or for local machine
```

Step 3: Access Jenkins so we need to password

Copy highlighted link & paste on the Terminal with below command;

Sudo cat …… (copy link)

Password will show now

Step 4: Create 1st Admin User

Name & password – admin

Email – admin@gmail.com

Save & finish

Jenkins Ready to use

Step 5: Create a New Item

1. Open Jenkins → click on **"New Item"**.
2. Enter a project name (e.g., `Java-CICD-Demo`).
3. Select Pipeline → click **OK**.

Step 6: Configure Git Repository

1. **Click on Pipeline syntax**
2. **Sample step – git Git**
3. **Repository URL – Add your GitHub repository URL**
4. Branch – main
5. Click on Generate pipeline script

Example:

```
https://github.com/your-username/java-sample-app.git
```

Step 7: Installed below Plugins

1. **Pipeline stage view**
2. **Maven installation (Build tool)**
3. **Maven integration plugin**

Step 8: Install Maven on Terminal with below commands

Sudo apt update
Sudo apt install maven –y
Mvn –version

Step 9. Go to tool option and make changes in name

1. jdk17

2. Maven3

Step 10. Sample Jenkins Console Output

```
Started by GitHub push
Building in workspace /var/lib/jenkins/workspace/Java-CICD-Demo
Cloning repository https://github.com/your-username/java-sample-app.git
...
[INFO] BUILD SUCCESS
Finished: SUCCESS
```

Step 11. Jenkins Pipeline (Optional Advanced Version)

If you prefer a **Jenkinsfile**, here's a basic scripted pipeline:

```
pipeline {
    agent any
    tools {
        maven 'Maven3'
        jdk 'jdk17'
    }
    stages {
        stage('Checkout') {
            steps {
                git 'https://github.com/your-username/java-sample-app.git'
            }
        }
        stage('Build') {
            steps {
                sh 'mvn clean package'
            }
        }
    }
}
```

Step12. Outcome:

✅Automated CI/CD pipeline for a Java app
✅Jenkins integrated with Git and Maven
✅Automatic build, test, and artifact storage