

## Building Java Projects with Maven - Task 5

Ragul R R

Date: 07.02.2025

---

### Overview

Maven is a robust project management tool designed for Java applications. Just as a manager coordinates tasks, resources, and deadlines, Maven efficiently manages dependencies, builds, tests, and deployments.

---

### Step 1: Installing Java and Maven on Ubuntu

Before proceeding, ensure that both Java and Maven are installed on your Ubuntu system.

```
ubuntu@LAPTOP-DEQKQVPU:~$ java -version
openjdk version "17.0.14" 2025-01-21
OpenJDK Runtime Environment (build 17.0.14+7-Ubuntu-124.04)
OpenJDK 64-Bit Server VM (build 17.0.14+7-Ubuntu-124.04, mixed mode, sharing)
ubuntu@LAPTOP-DEQKQVPU:~$ |
```

```
ubuntu@LAPTOP-DEQKQVPU:~$ readlink -f $(which java)
/usr/lib/jvm/java-17-openjdk-amd64/bin/java
ubuntu@LAPTOP-DEQKQVPU:~$ |
```

Google Docs: Online docu... x Task\_5 - Google Docs x Task\_4 - Google Docs x Document Summary Requ x ragul2222/ar-ekart x Tools [Jenkins] x + - □ ×

localhost:8080/manage/configureTools/ ☆ 🔒

Dashboard > Manage Jenkins > Tools

JDK installations

JDK installations ^ Edited

Add JDK

≡ JDK

Name

jdk

JAVA\_HOME

/usr/lib/jvm/java-17-openjdk-amd64

☐ Install automatically ?

Add JDK

Save Apply

65°F Haze Search 22:06 11-02-2025

```
ubuntu@LAPTOP-DEQKQVPU:~$ readlink -f $(which mvn)
/usr/share/maven/bin/mvn
ubuntu@LAPTOP-DEQKQVPU:~$ |
```

Google Docs: Online docu... x Task\_5 - Google Docs x Task\_4 - Google Docs x Document Summary Requ x ragul2222/ar-ekart x Tools [Jenkins] x + - □ ×

localhost:8080/manage/configureTools/ ☆ 🔒

Dashboard > Manage Jenkins > Tools

Maven installations

Maven installations ^ Edited

Add Maven

≡ Maven

Name

maven

MAVEN\_HOME

/usr/share/maven

☐ Install automatically ?

Add Maven

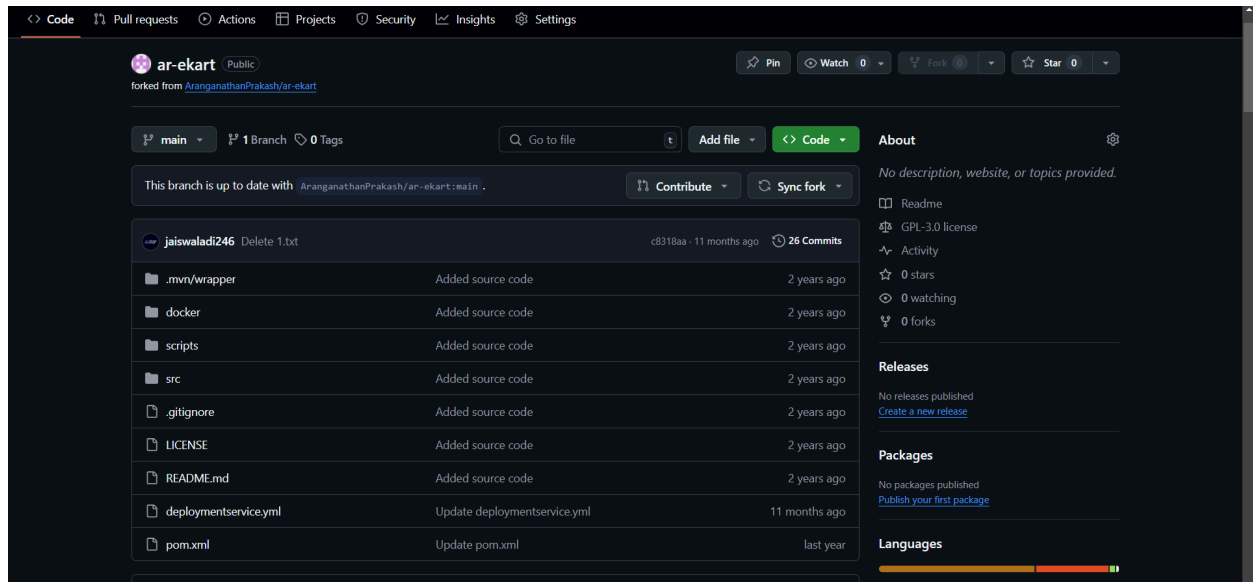
Docker installations

Save Apply

65°F Haze Search 22:07 11-02-2025

## Step 2: Forking the eKart Repository on GitHub

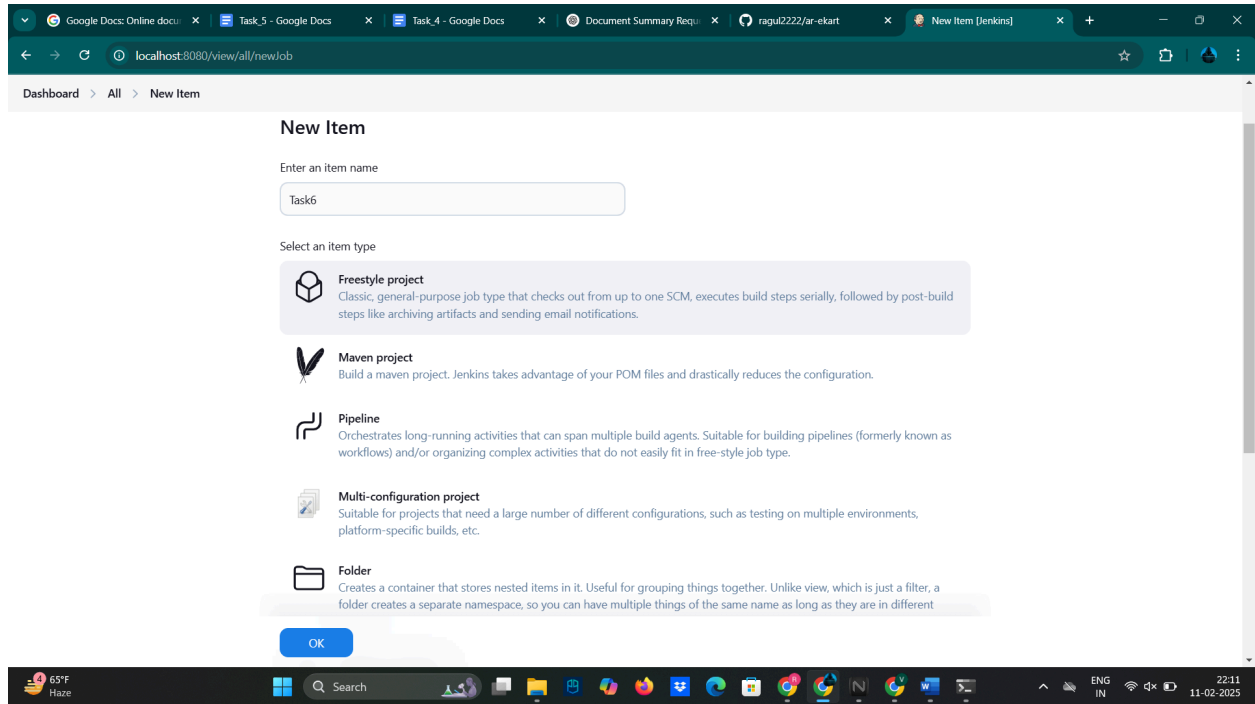
Fork the eKart repository to your personal GitHub account for seamless modification and integration.



## Step 3: Configuring Jenkins

### Creating a New Jenkins Job:

1. Open Jenkins in your browser.
2. Click **New Item** → Choose **Freestyle Project**.
3. Name the project **task6** and click **OK**.



## Job Configuration:

- **Setting Up Build Tools:**
  - Navigate to **Global Tool Configuration**.
  - Verify Java and Maven are installed, and configure if necessary.
- **Configuring GitHub Repository:**
  - Under **Source Code Management**, select **Git**.
  - Paste the forked repository URL.
  - Set the branch to **main**.

Google Docs: Online docu...Task\_5 - Google DocsTask\_4 - Google DocsDocument Summary Requ...ragul2222/ar-ekartTask\_6 Config [Jenkins]

localhost:8080/job/Task\_6/configure

DashboardTask\_6Configuration

Configure

General

Source Code Management

Build Triggers

Build Environment

Build Steps

Post-build Actions

Source Code Management

None

Git

Repositories

Repository URL

https://github.com/ragul2222/ar-ekart.git

Credentials

- none -

+ Add

Advanced

Add Repository

Branches to build

Branch Specifier (blank for 'any')

Save

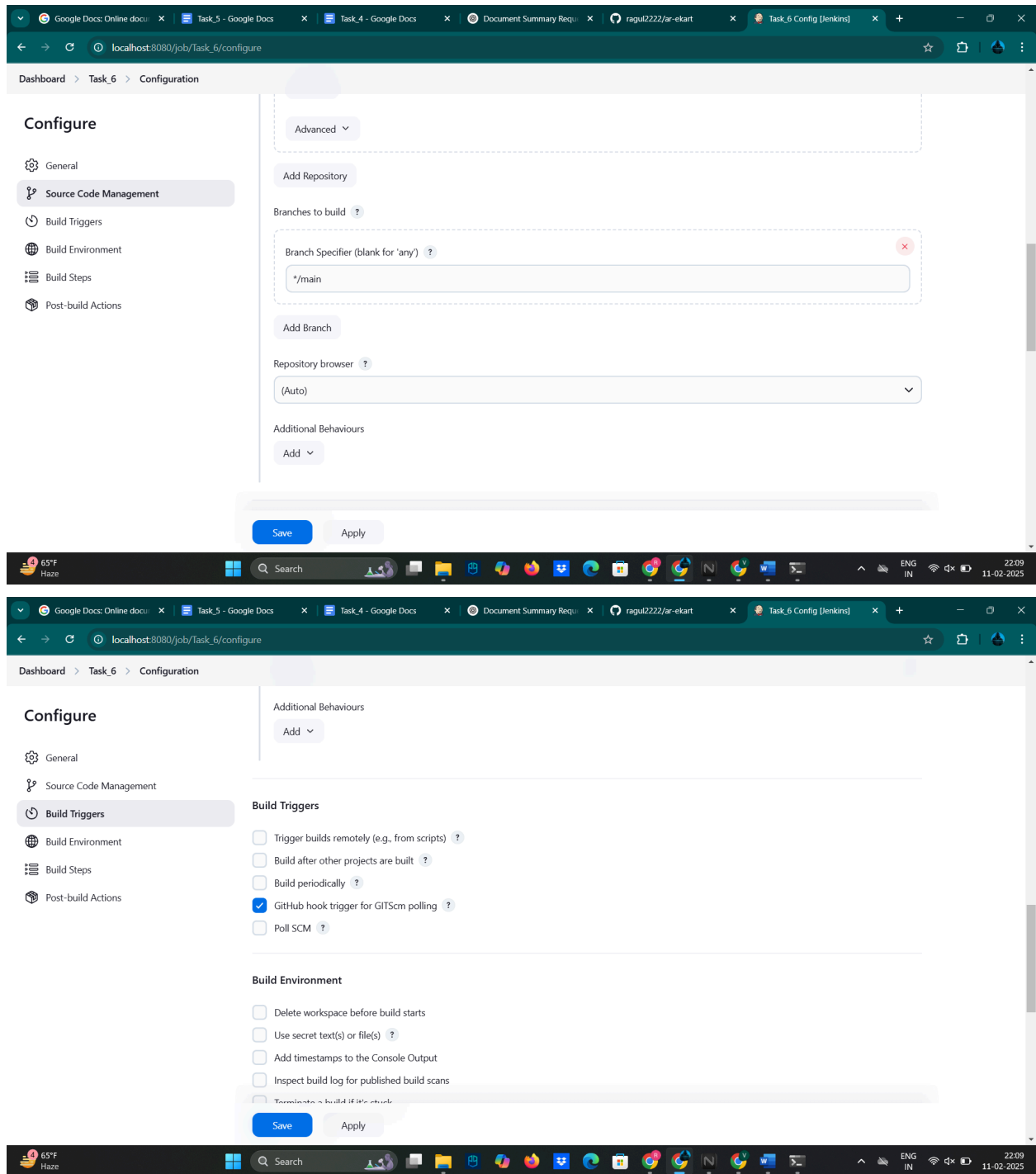
Apply

65°F Haze

Search

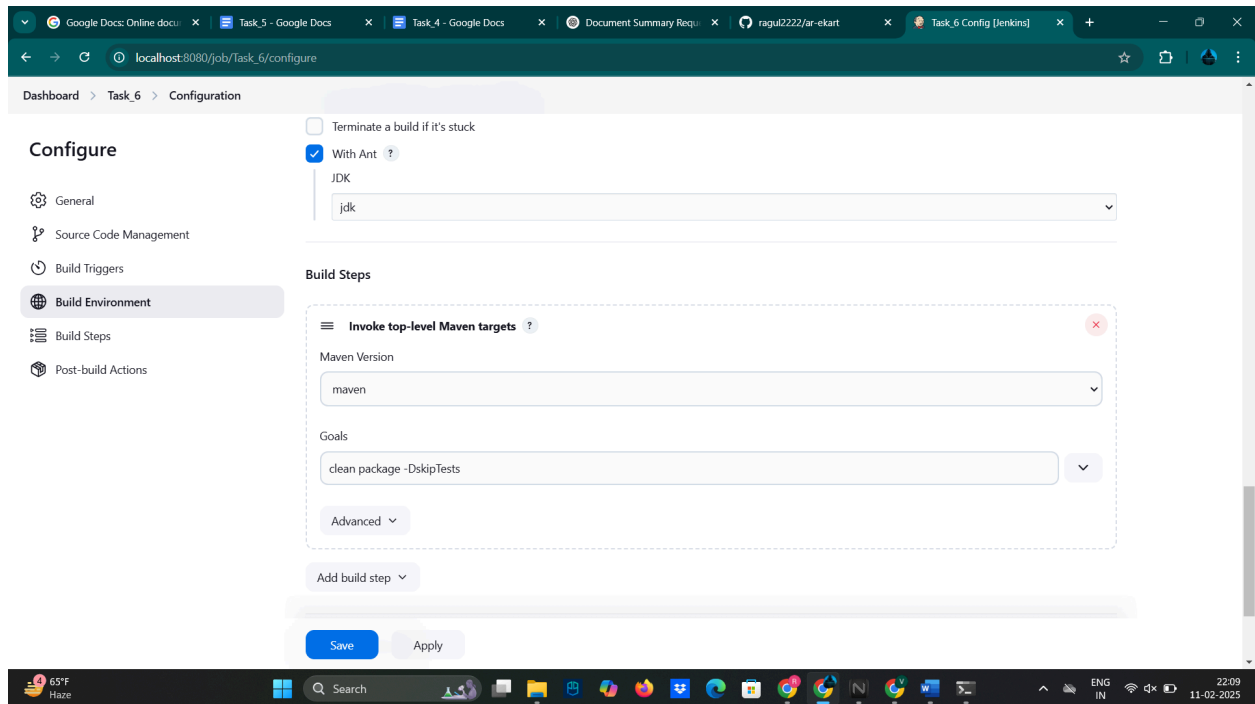
ENG IN

22:09 11-02-2025



- **Adding a Build Command:**

- Go to **Build** → **Add Build Step** → Choose **Invoke top-level Maven targets**.



Enter the following command:

```
clean package -DskipTests
```

- Click **Build Now** to execute.

---

## Step 4: Navigating Jenkins Workspace

Move to the Jenkins workspace directory:

```
cd /var/lib/jenkins/workspace
```

1.

List all available projects:

```
ls
```

2.

Navigate to the project folder:

```
cd Maven_task5/target
```

3.

Verify generated artifacts (e.g., **.jar** file):

```
ls
```

4.

---

## Step 5: Deploying via Docker and Kubernetes

### Building and Pushing Docker Image:

```
docker build -t test -f docker/Dockerfile
```

```
docker push rragul/sample
```

### Deploying on Kubernetes:

```
kubectl create deployment maven --image=test --port=80
```

```
kubectl expose deployment maven --type=NodePort --port=80 --target-port=8070
```

### Verifying Deployment:

```
docker images | grep nivethitha24/mave # Confirm Docker image creation
```

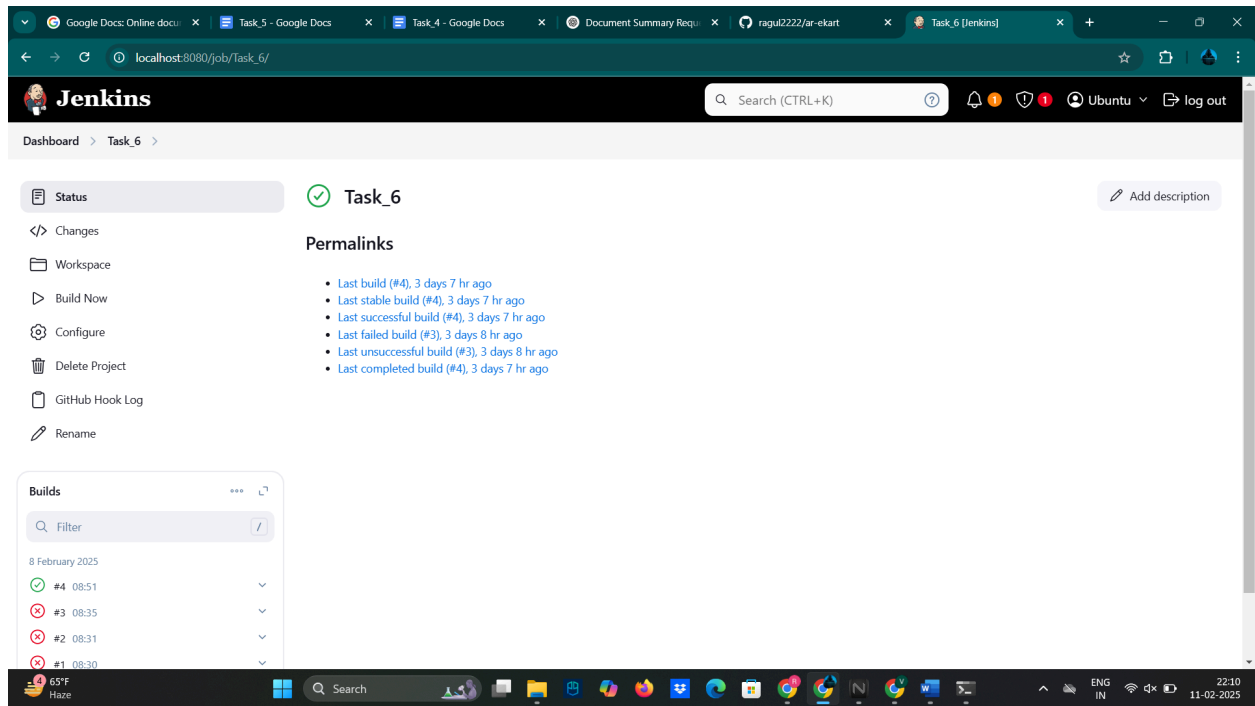
```
kubectl get pods # List active pods
```

```
minikube service maven # Retrieve service URL
```

```
ubuntu@LAPTOP-DEQKQVPU:~$ minikube start
minikube v1.35.0 on Ubuntu 24.04 (amd64)
👉 Using the docker driver based on existing profile
👉 Starting "minikube" primary control-plane node in "minikube" cluster
👉 Pulling base image v0.0.46 ...
👉 Restarting existing docker container for "minikube" ...
👉 Preparing Kubernetes v1.32.0 on Docker 27.4.1 ...
👉 Verifying Kubernetes components...
   ▪ Using image gcr.io/k8s-minikube/storage-provisioner:v5
👉 Enabled addons: storage-provisioner, default-storageclass
👉 Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
```

```
ubuntu@LAPTOP-DEQKQVPU:~$ minikube service test-service
-----
| NAMESPACE | NAME       | TARGET PORT | URL                               |
|-----|-----|-----|-----|
| default   | test-service | http/80      | http://192.168.49.2:32279        |
|-----|-----|-----|-----|
👉 Starting tunnel for service test-service.
-----
| NAMESPACE | NAME       | TARGET PORT | URL                               |
|-----|-----|-----|-----|
| default   | test-service |             | http://127.0.0.1:37439          |
|-----|-----|-----|-----|
👉 Opening service default/test-service in default browser...
👉 http://127.0.0.1:37439
👉 ! Because you are using a Docker driver on linux, the terminal needs to be open to run it.
```





## Jenkins Configuration & Output

