Let’s say your team is developing a Java web application using Spring Boot. The code is stored on GitHub. You want Jenkins to:

1. Pull the latest code from GitHub.
2. Build the code using Maven (`mvn clean package`).
3. Run unit tests.
4. Archive the generated JAR file.
5. Send a build success/failure email.

✅ Step-by-Step Breakdown in Jenkins (based on your screenshots)

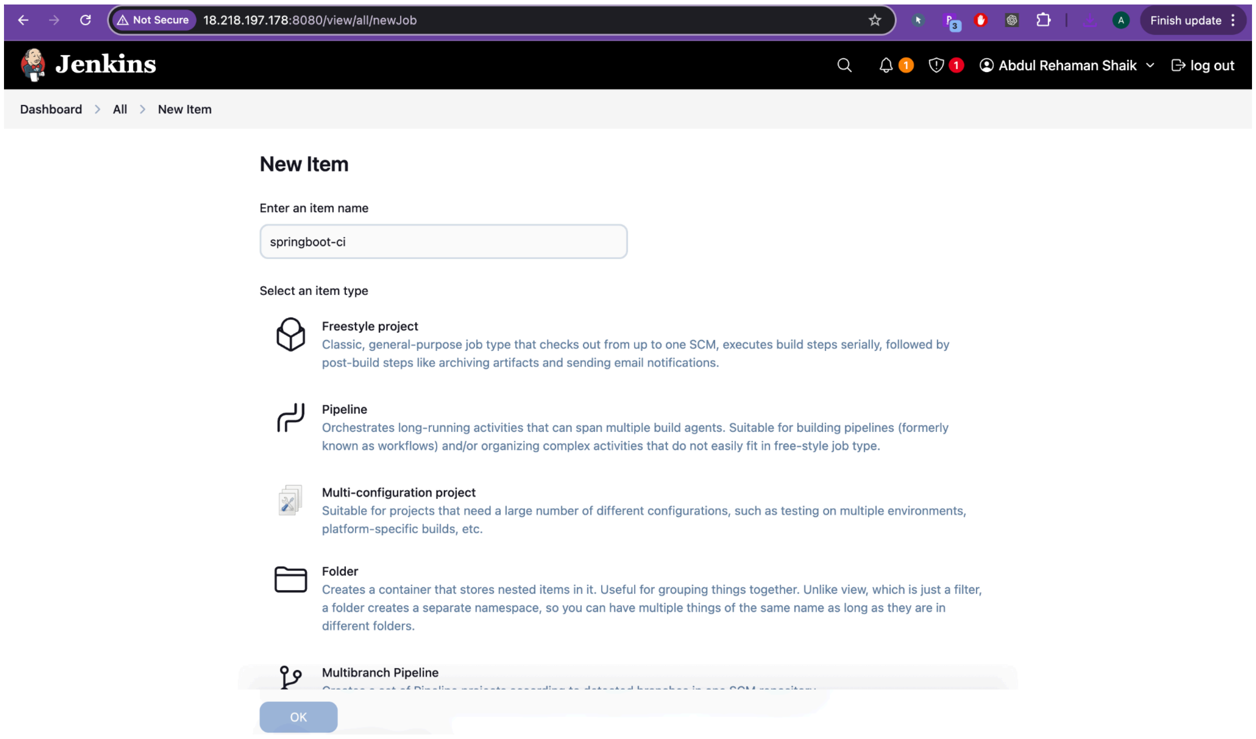
🔹 Step 1: Jenkins Dashboard (Screenshot 1)

- This is your **home screen** after logging into Jenkins.
- You’ll start by clicking on **"New Item"** to create a job.

🔹 Step 2: Create a New Freestyle Project (Screenshot 2)

- **Name:** `myjob01` (you can name it like `springboot-ci`)
- **Select Type:** Freestyle project
- ✅ Real-world reason to use Freestyle:

Ideal for simple CI jobs like build + test + notify, without complex pipeline scripts.


Not Secure 18.218.197.178:8080/view/all/newJob

Jenkins

Abdul Rehaman Shaik log out

Dashboard > All > New Item

New Item

Enter an item name

springboot-ci

Select an item type

Freestyle project
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

Multibranch Pipeline

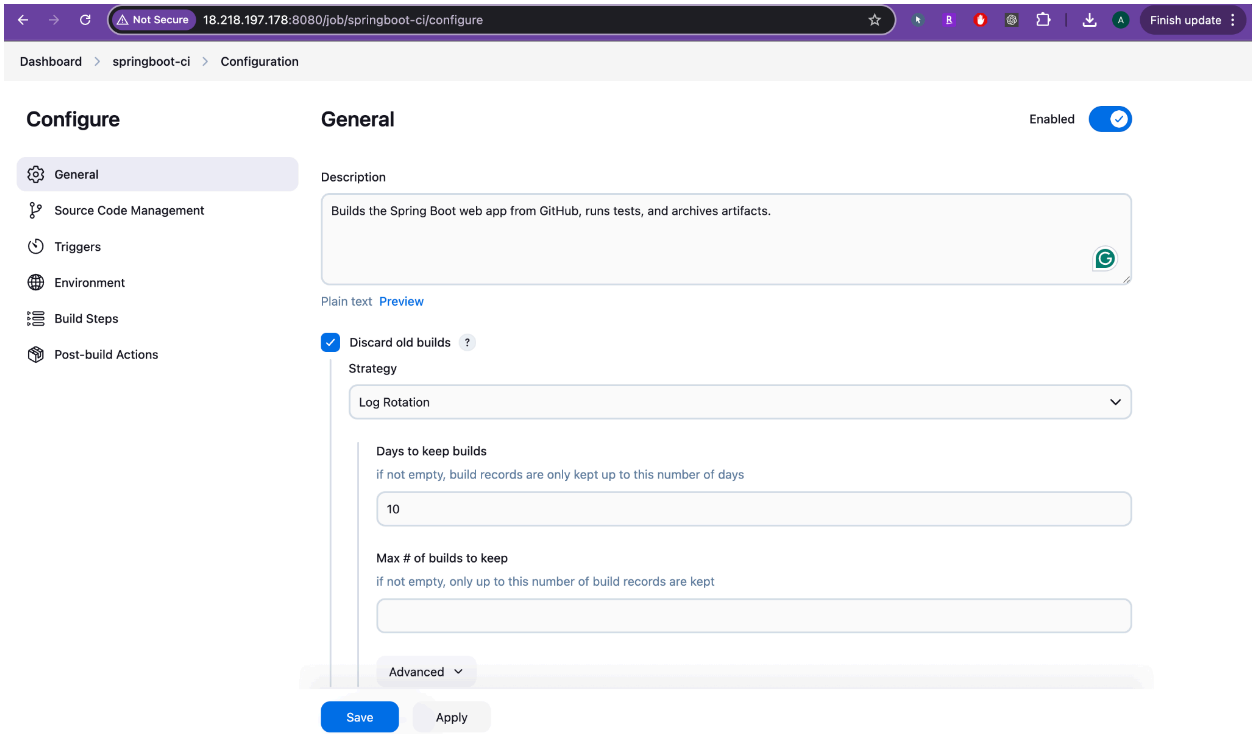
OK

🔹 Step 3: General Configuration (Screenshot 3)

- Add a **description** like:

  “Builds the Spring Boot web app from GitHub, runs tests, and archives artifacts.”

- Check:
  - `Discard old builds`: Retain only last N builds (say 10).
  - `This project is parameterized`: If you want to build with options (e.g., `ENV=dev` or `ENV=prod`).



18.218.197.178:8080/job/springboot-ci/configure

Dashboard > springboot-ci > Configuration

Configure

General

Enabled ✓

Description

Builds the Spring Boot web app from GitHub, runs tests, and archives artifacts.

Plain text Preview

✓ Discard old builds

Strategy

Log Rotation

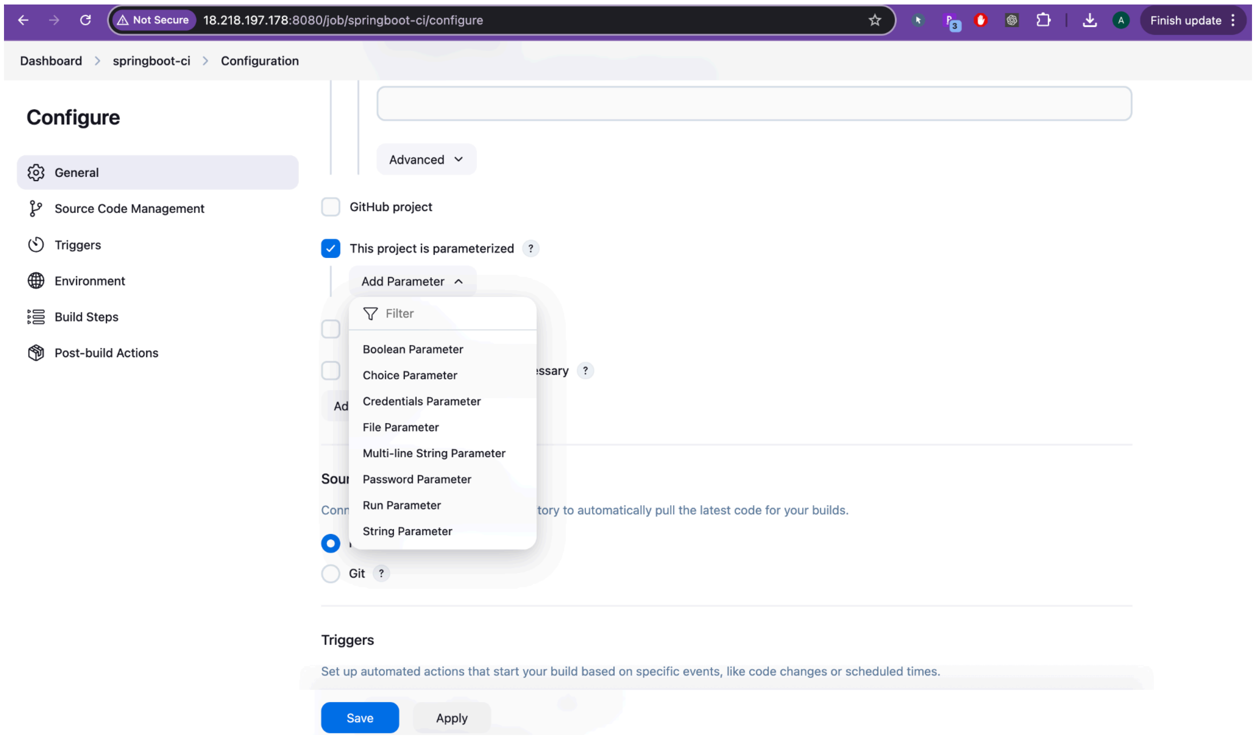
Days to keep builds

if not empty, build records are only kept up to this number of days

10

Max # of builds to keep

if not empty, only up to this number of build records are kept



18.218.197.178:8080/job/springboot-ci/configure

Dashboard > springboot-ci > Configuration

☐ GitHub project

✓ This project is parameterized

Add Parameter

Boolean Parameter
Choice Parameter
Credentials Parameter
File Parameter
Multi-line String Parameter
Password Parameter
Run Parameter
String Parameter

Triggers

Set up automated actions that start your build based on specific events, like code changes or scheduled times.

🔹 Step 4: Source Code Management (Screenshot 4)

- **Select:** `Git`

Repository URL:

```
https://github.com/your-org/springboot-webapp.git
```

- **Credentials:** Add GitHub token (so Jenkins can access private repos).

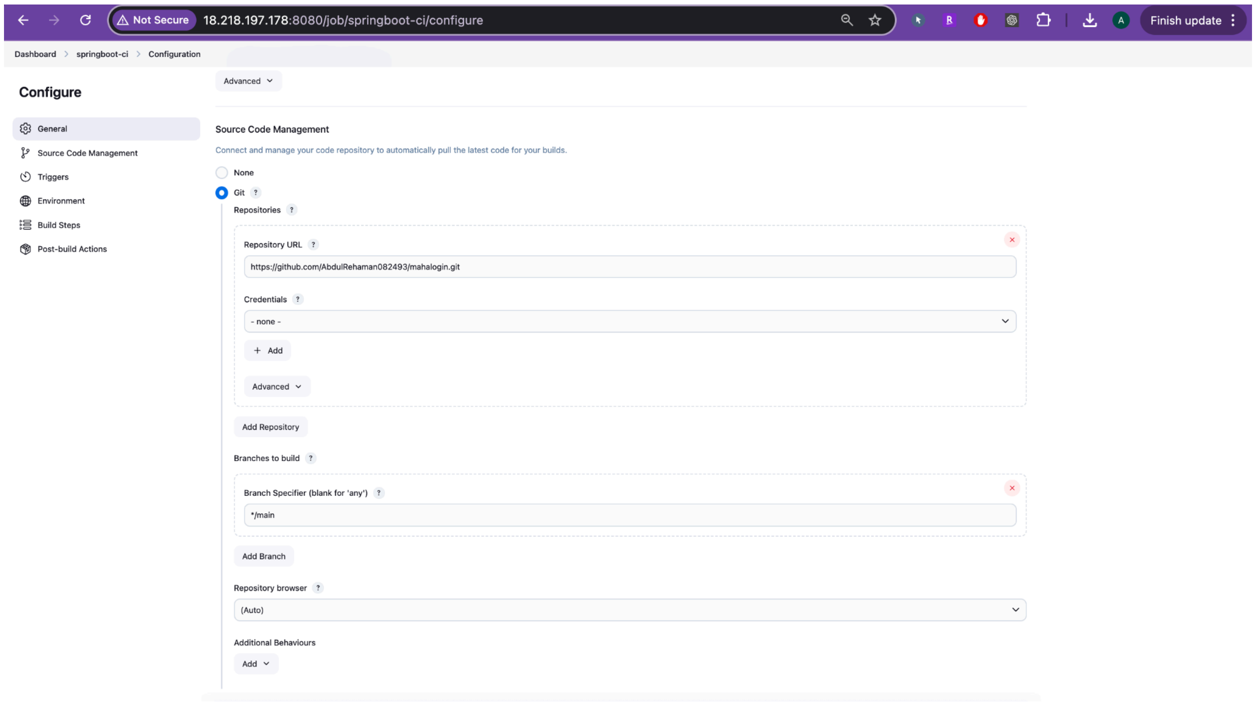
Branches to build:

```
*/main
```

-

✅ **Real-World Purpose**:

Jenkins pulls the latest code from GitHub `main` branch before every build.



Source Code Management

Connect and manage your code repository to automatically pull the latest code for your builds.

None [✓ Git]

Repositories

Repository URL: https://github.com/AbdulRehaman082493/mahalogin.git

Credentials: - none -

Branches to build

Branch Specifier (blank for 'any'): */main

Repository browser: (Auto)

Additional Behaviours

🔹 Step 5: Build Triggers (Screenshot 4)

Choose how the job should start:

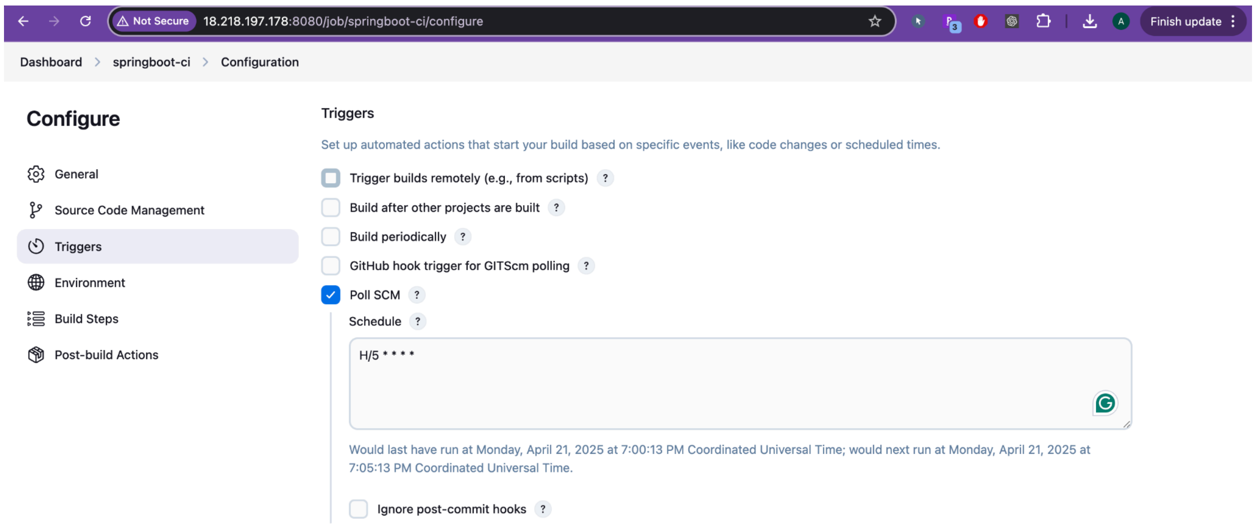
`Poll SCM`:

```
H/5 * * * *
```

- → Check for new commits every 5 minutes.
- `GitHub hook trigger for GITScm polling`:
  → Automatically trigger a build when someone pushes code to GitHub (needs webhook set in GitHub).

✅ **Real-World Use**:

Automatically run CI pipeline whenever developers push code.



Triggers

Set up automated actions that start your build based on specific events, like code changes or scheduled times.

- [ ] Trigger builds remotely (e.g., from scripts)
- [ ] Build after other projects are built
- [ ] Build periodically
- [ ] GitHub hook trigger for GITScm polling
- [✓] Poll SCM

Schedule

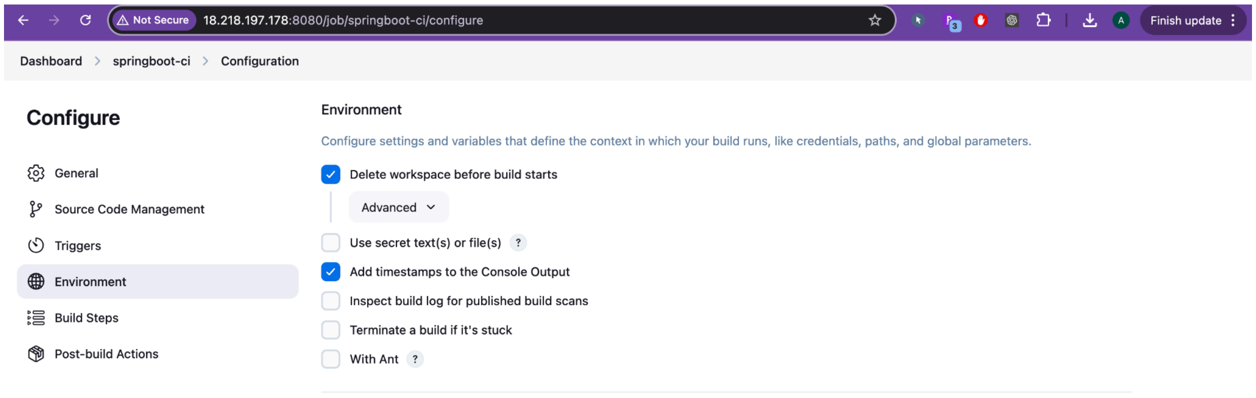
```
H/5 * * * *
```

Would last have run at Monday, April 21, 2025 at 7:00:13 PM Coordinated Universal Time; would next run at Monday, April 21, 2025 at 7:05:13 PM Coordinated Universal Time.

- [ ] Ignore post-commit hooks

🔹 Step 6: Build Environment (Screenshot 5)

- Check:
    - `Delete workspace before build`: Clean start.
    - `Add timestamps`: Easier log reading.



18.218.197.178:8080/job/springboot-ci/configure

Dashboard > springboot-ci > Configuration

Configure

Environment

Configure settings and variables that define the context in which your build runs, like credentials, paths, and global parameters.

- [✓] Delete workspace before build starts
- [ ] Use secret text(s) or file(s)
- [✓] Add timestamps to the Console Output
- [ ] Inspect build log for published build scans
- [ ] Terminate a build if it's stuck
- [ ] With Ant

🔹 Step 7: Build Steps (Screenshot 5)

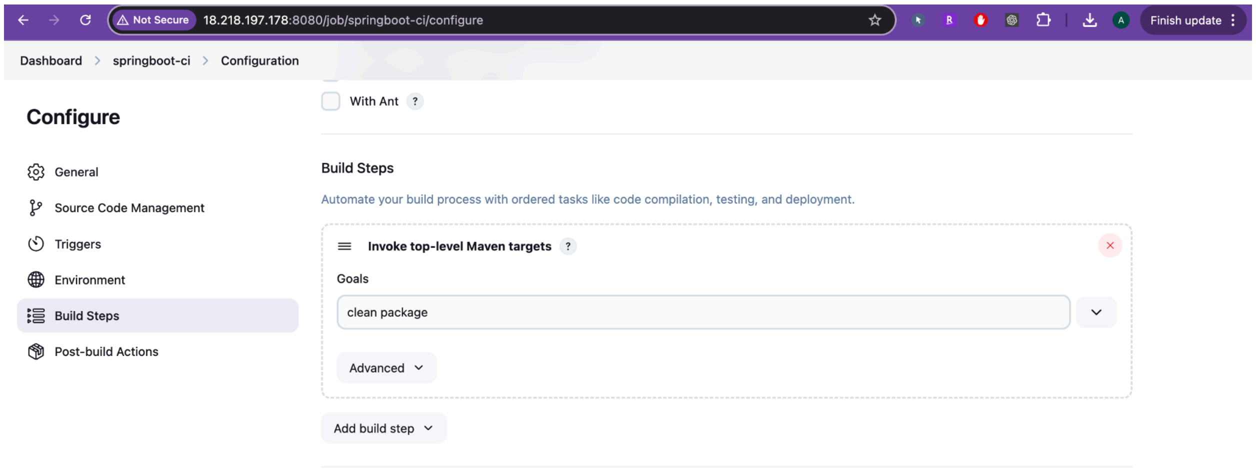
👉 Add Build Step > Invoke top-level Maven targets

Goals:
`clean package`

Root POM:
`pom.xml`

✅ What Happens:

> This will compile the code, run unit tests, and package the app into a JAR or WAR file using `mvn clean package`.


Dashboard > springboot-ci > Configuration

Configure

- General
- Source Code Management
- Triggers
- Environment
- Build Steps
- Post-build Actions

With Ant

Build Steps

Automate your build process with ordered tasks like code compilation, testing, and deployment.

Invoke top-level Maven targets

Goals: clean package

Advanced

Add build step

🔹 Step 8: Post-Build Actions (Screenshot 6)

1. 📦 Archive the Artifacts

Files to archive:

```
target/*.jar
```

✅ Why?

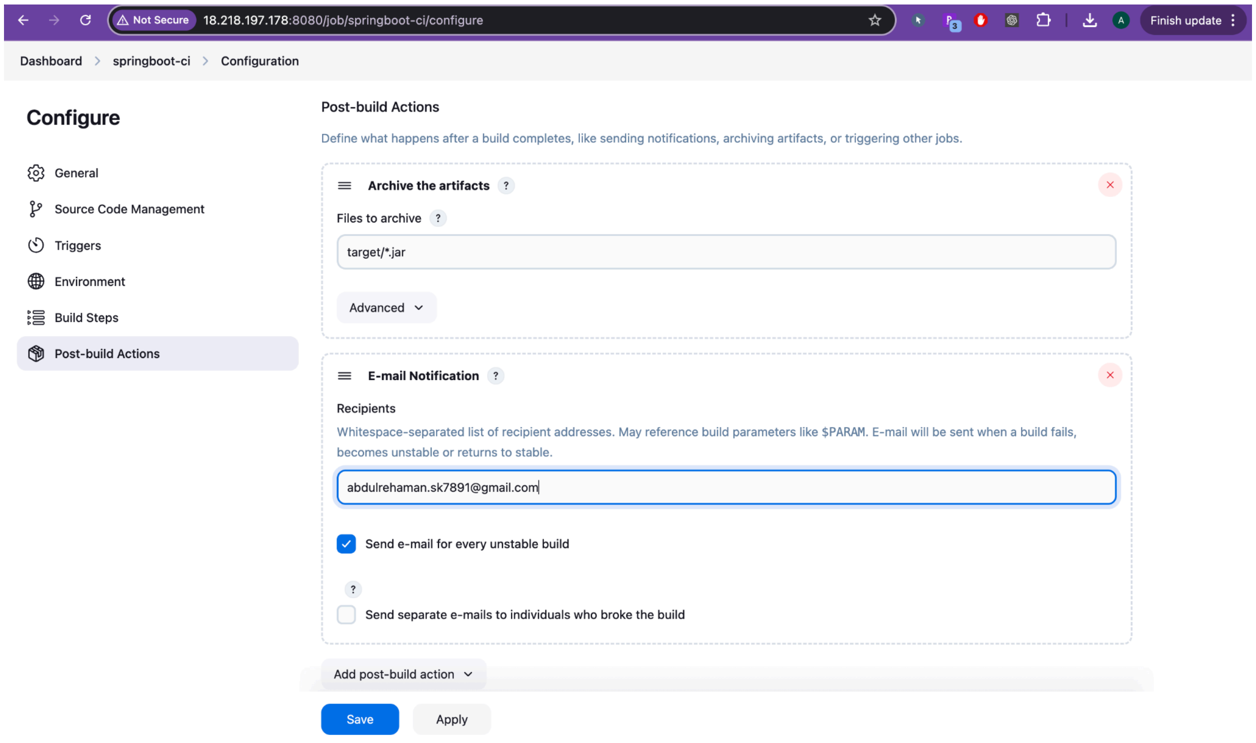
> This makes the built JAR downloadable from Jenkins UI and useful for deployment steps.

2. 📧 Email Notification

- Add "Email Notification":
  - Recipients: `dev-team@example.com`
  - Notify on failure/success

✅ Why?

> Team is immediately informed if build fails or succeeds.


18.218.197.178:8080/job/springboot-ci/configure
Dashboard > springboot-ci > Configuration
Configure
General
Source Code Management
Triggers
Environment
Build Steps
Post-build Actions

Post-build Actions

Define what happens after a build completes, like sending notifications, archiving artifacts, or triggering other jobs.

Archive the artifacts

Files to archive: target/*.jar

Advanced

E-mail Notification

Recipients

Whitespace-separated list of recipient addresses. May reference build parameters like $PARAM. E-mail will be sent when a build fails, becomes unstable or returns to stable.

abdulrehaman.sk7891@gmail.com

[✓] Send e-mail for every unstable build

[ ] Send separate e-mails to individuals who broke the build

Add post-build action

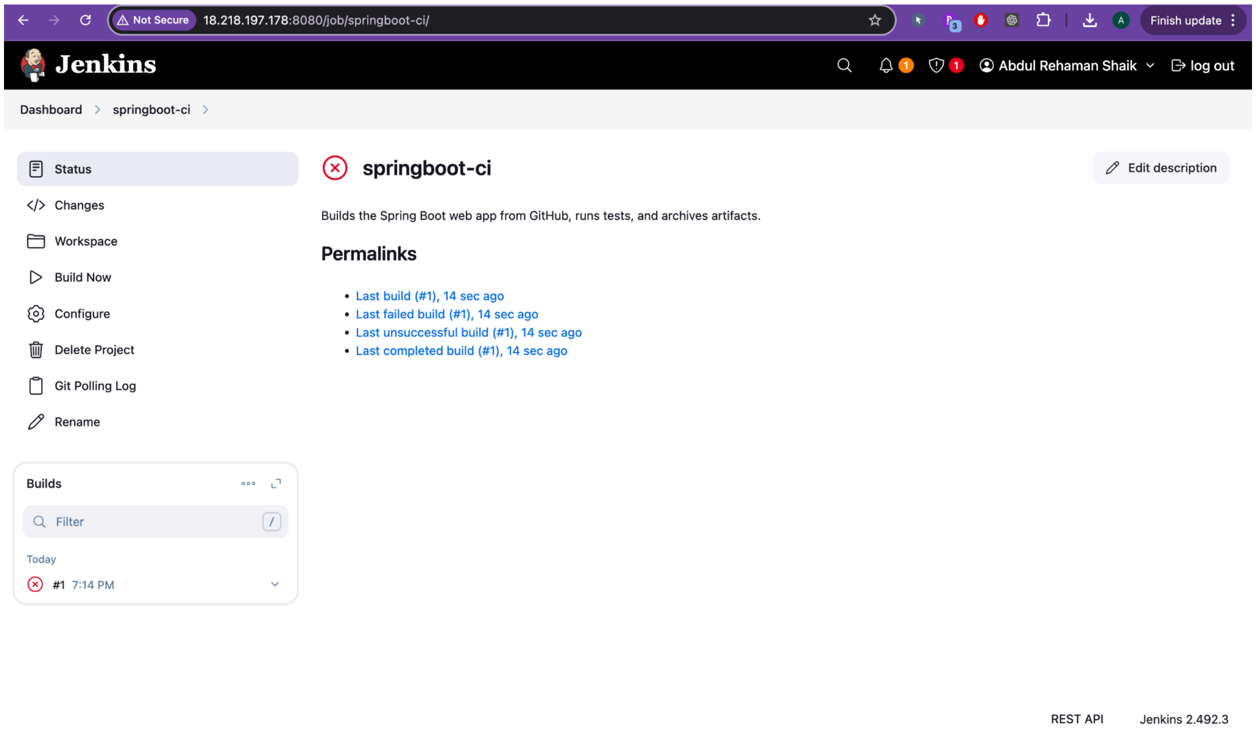
Save Apply

When someone pushes new code to GitHub:

1. Jenkins gets triggered via webhook or polling.
2. It pulls the latest code.
3. Runs `mvn clean package`.
4. Runs unit tests (fails the build if tests fail).
5. Archives the generated `.jar`.
6. Sends build status emails to the team.

Folder Structure of Project (GitHub)

```
springboot-webapp/
├── src/
│   └── main/
│       └── java/
│           └── com/example/
│               └── App.java
├── src/
│   └── test/
│       └── java/
│           └── com/example/
│               └── AppTest.java
├── pom.xml
```


18.218.197.178:8080/job/springboot-ci/

Dashboard > springboot-ci >

springboot-ci

Builds the Spring Boot web app from GitHub, runs tests, and archives artifacts.

Permalinks

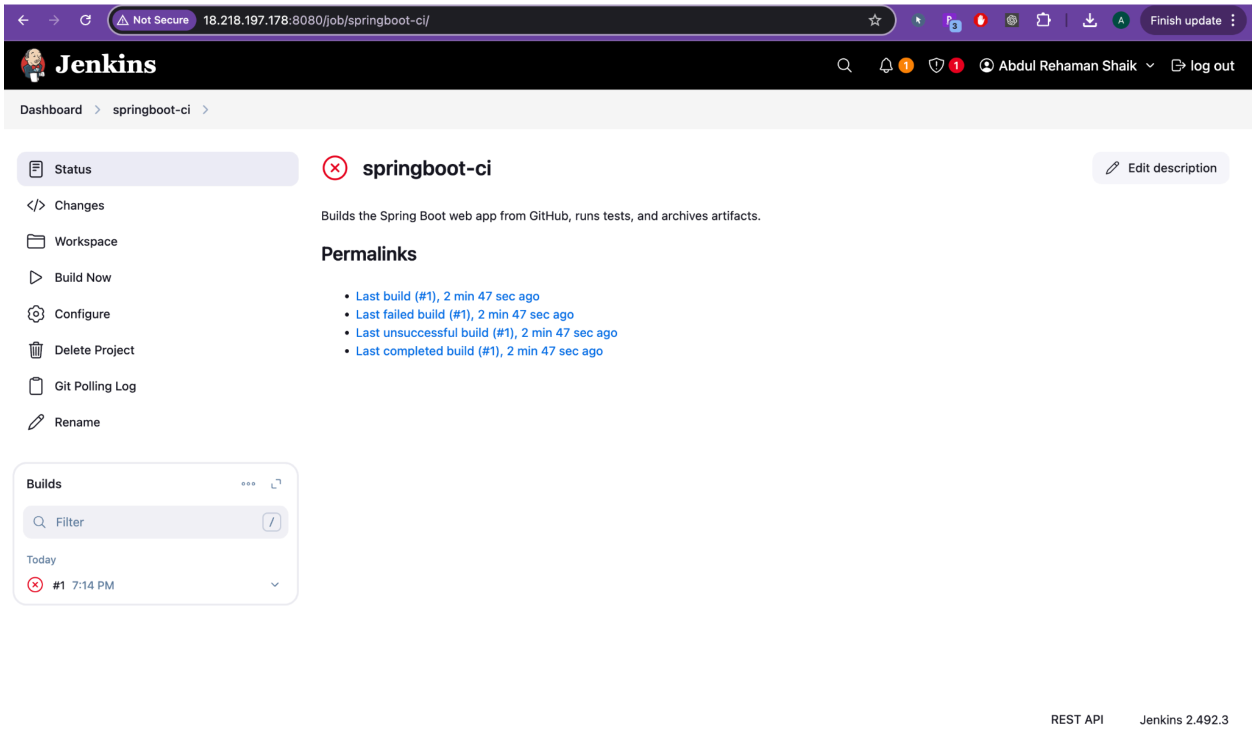
- Last build (#1), 14 sec ago
- Last failed build (#1), 14 sec ago
- Last unsuccessful build (#1), 14 sec ago
- Last completed build (#1), 14 sec ago

Builds

Today

#1 7:14 PM

REST API Jenkins 2.492.3


18.218.197.178:8080/job/springboot-ci/

Dashboard > springboot-ci >

springboot-ci

Builds the Spring Boot web app from GitHub, runs tests, and archives artifacts.

Permalinks

- Last build (#1), 2 min 47 sec ago
- Last failed build (#1), 2 min 47 sec ago
- Last unsuccessful build (#1), 2 min 47 sec ago
- Last completed build (#1), 2 min 47 sec ago

Builds

Today

#1 7:14 PM

REST API Jenkins 2.492.3