Practice for Lesson 2: Build and Run Jobs on Jenkins

Practices for Lesson 2

Overview

In these practices, you will build and run a job on Jenkins. Further, schedule a job to execute periodically and also integrate GitHub repository source code with Jenkins job.

Practice 2-2: Scheduling Jobs on Jenkins Instance

Overview

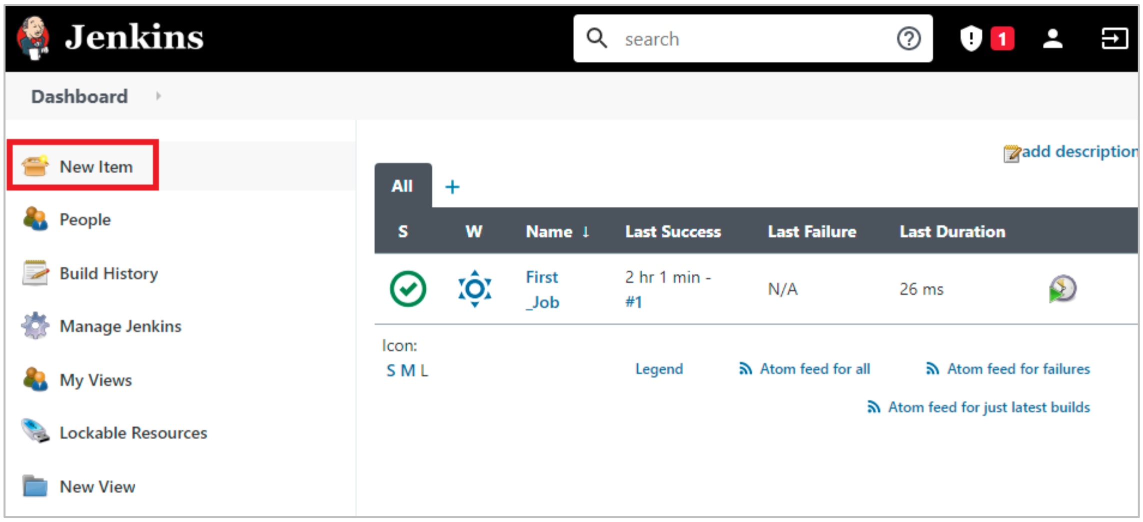
In this practice, you will learn how to periodically schedule a Job on Jenkins instance.

Assumptions

You should have completed the Practice of Lesson 2-1.

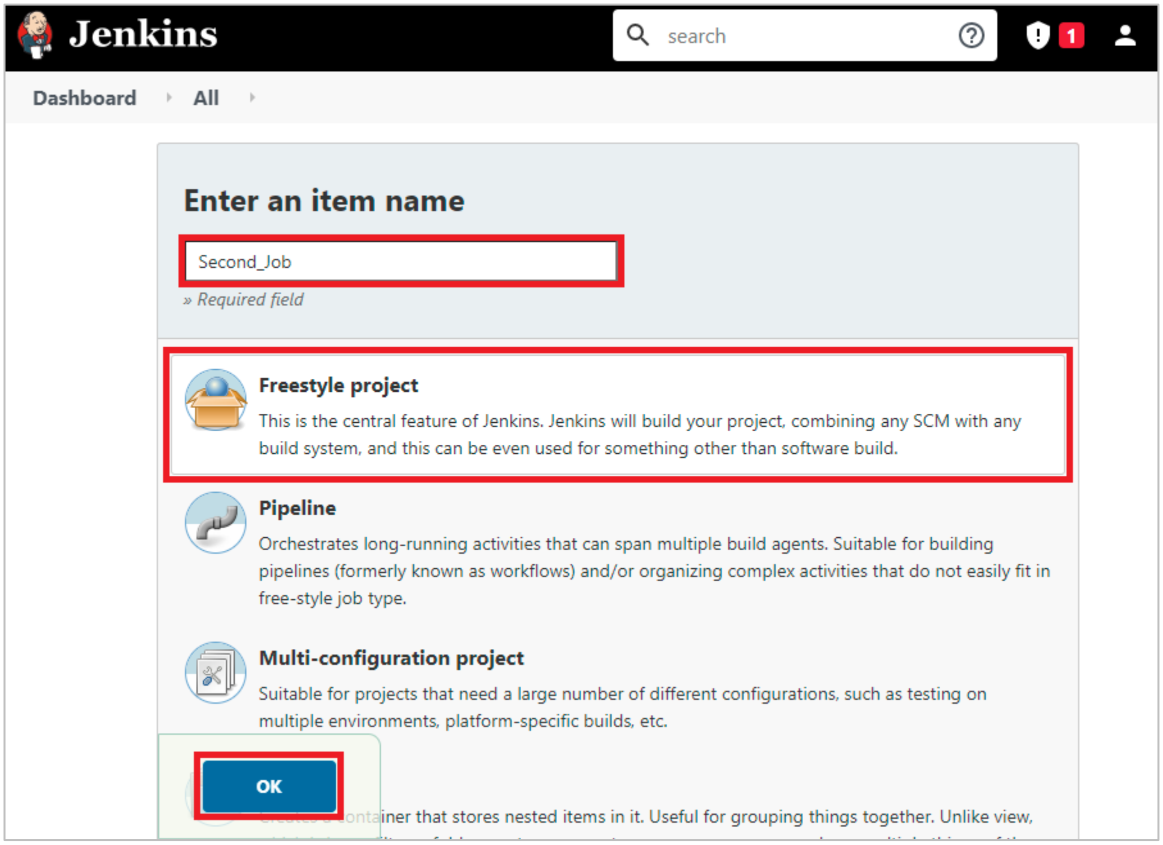
Tasks

1. To schedule a Jenkins job for every minute.
   a. In the Jenkins instance Dashboard navigate to the menu and select **New Item** to create a new job to schedule as shown below.



S	W	Name ↓	Last Success	Last Failure	Last Duration
✓	☀	First _Job	2 hr 1 min - #1	N/A	26 ms

   b. Provide the name for the Job in Jenkins, select **Freestyle project** and click **OK** as shown below.



Enter an item name

Second_Job

» *Required field*

Freestyle project

This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

Pipeline

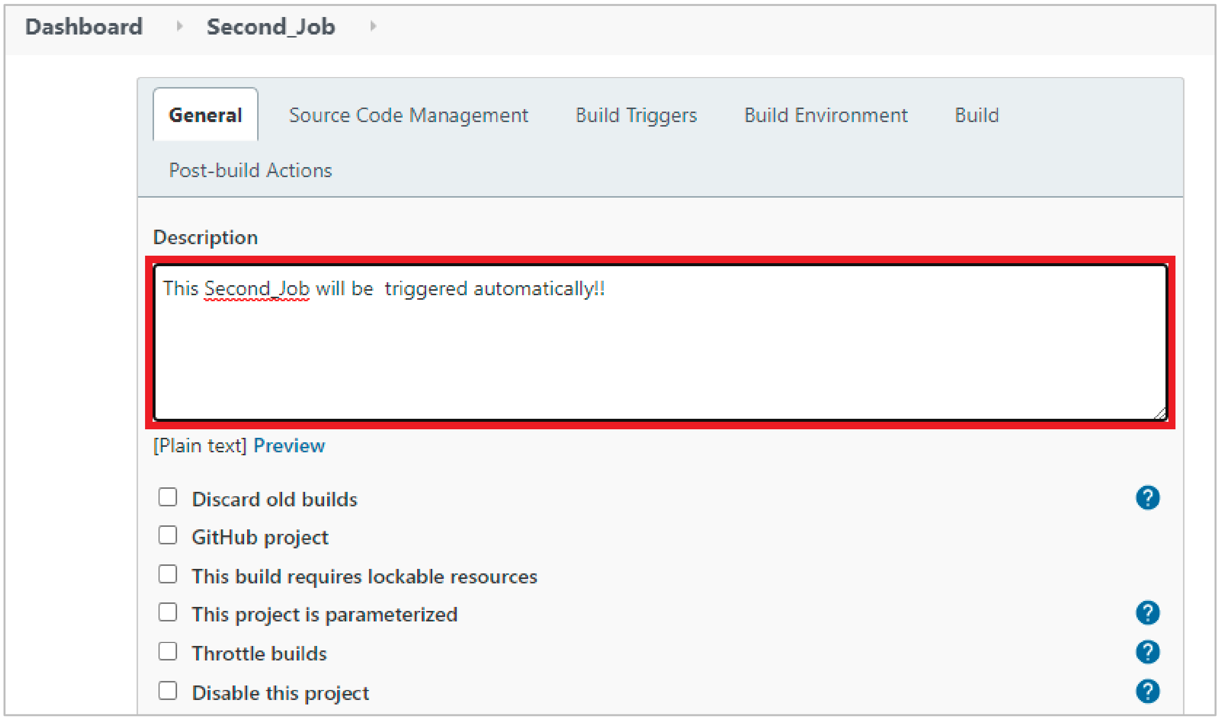
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project

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

ontainer that stores nested items in it. Useful for grouping things together. Unlike view,

c. Provide Description for the new job as shown below which is an optional.



Description

This Second_Job will be triggered automatically!!

□ Discard old builds

□ GitHub project

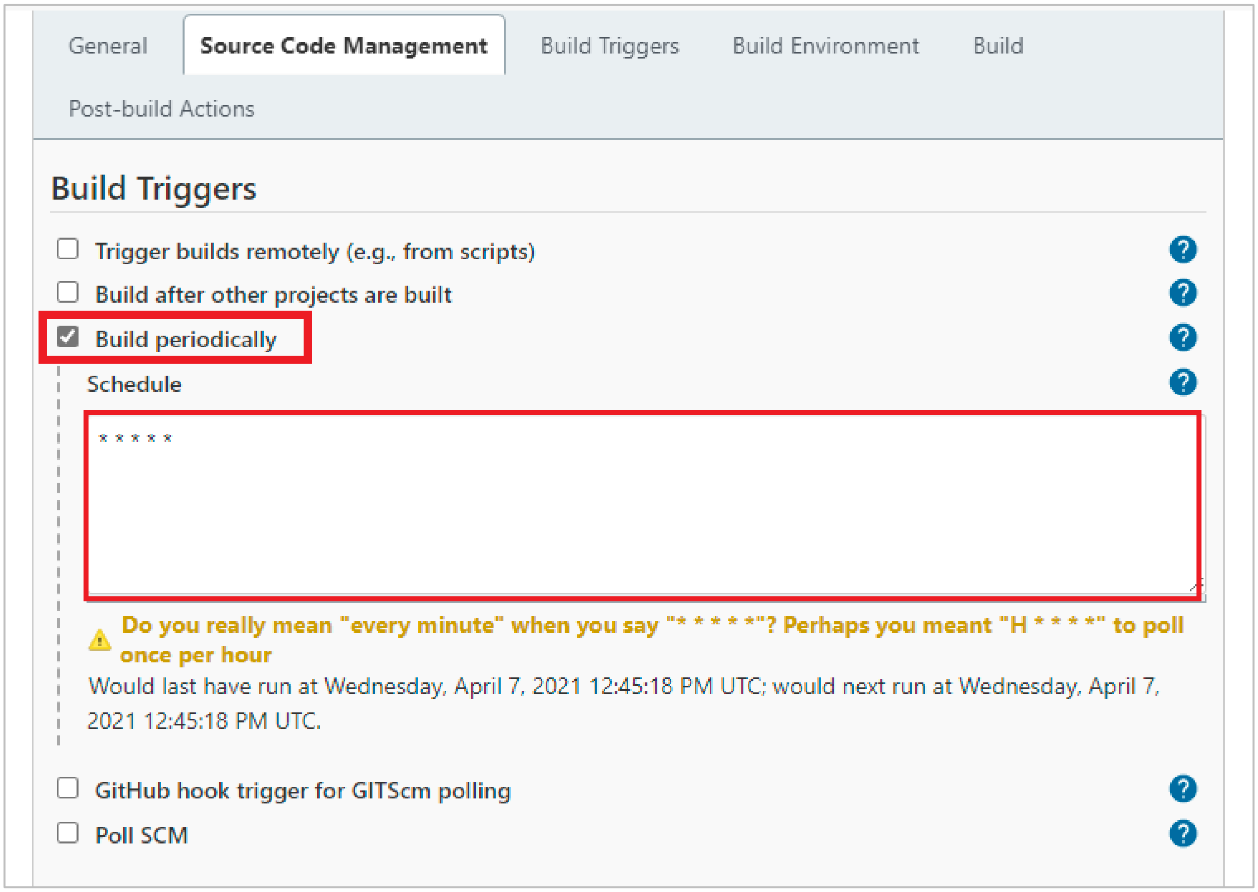
□ This build requires lockable resources

□ This project is parameterized

□ Throttle builds

□ Disable this project

d. Scroll down to **Build Triggers** and select the check box of **Build periodically.** In **Schedule** provide the pattern (syntax) for executing the job every minute as shown below.



Build Triggers

□ Trigger builds remotely (e.g., from scripts)

□ Build after other projects are built

☑ Build periodically

Schedule

```
* * * * *
```

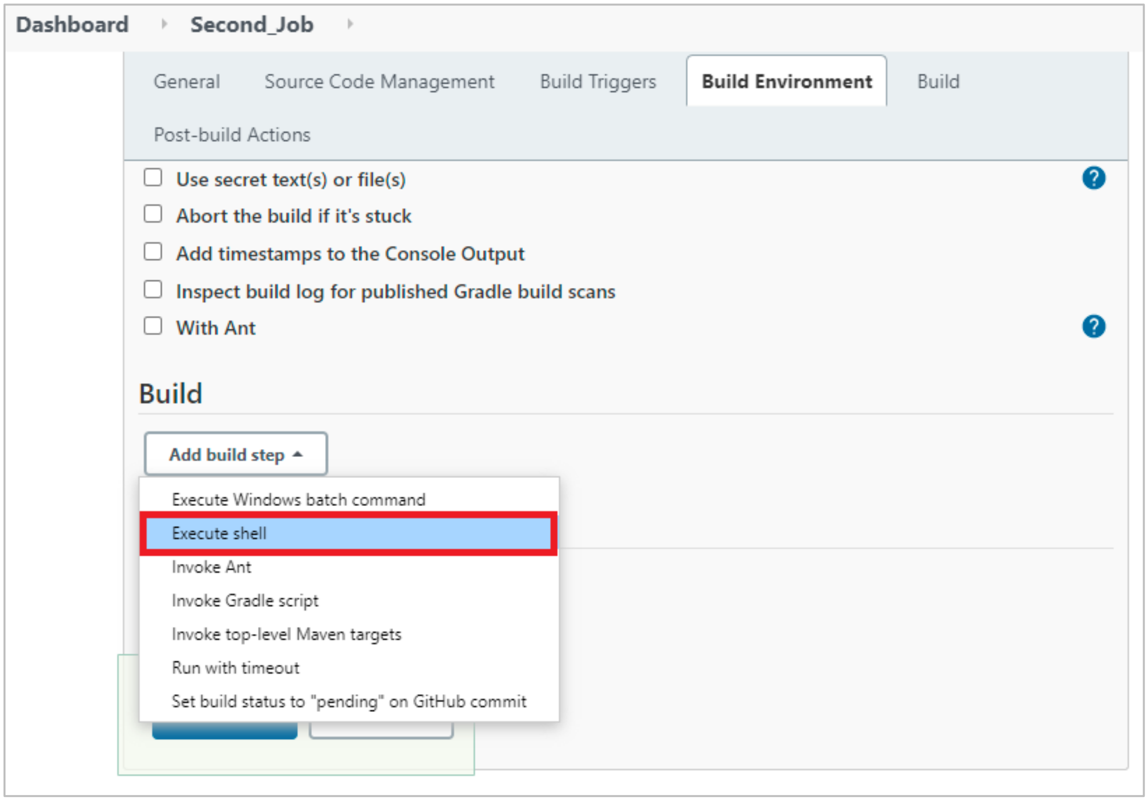
⚠ Do you really mean "every minute" when you say "* * * * *"? Perhaps you meant "H * * * *" to poll once per hour

Would last have run at Wednesday, April 7, 2021 12:45:18 PM UTC; would next run at Wednesday, April 7, 2021 12:45:18 PM UTC.

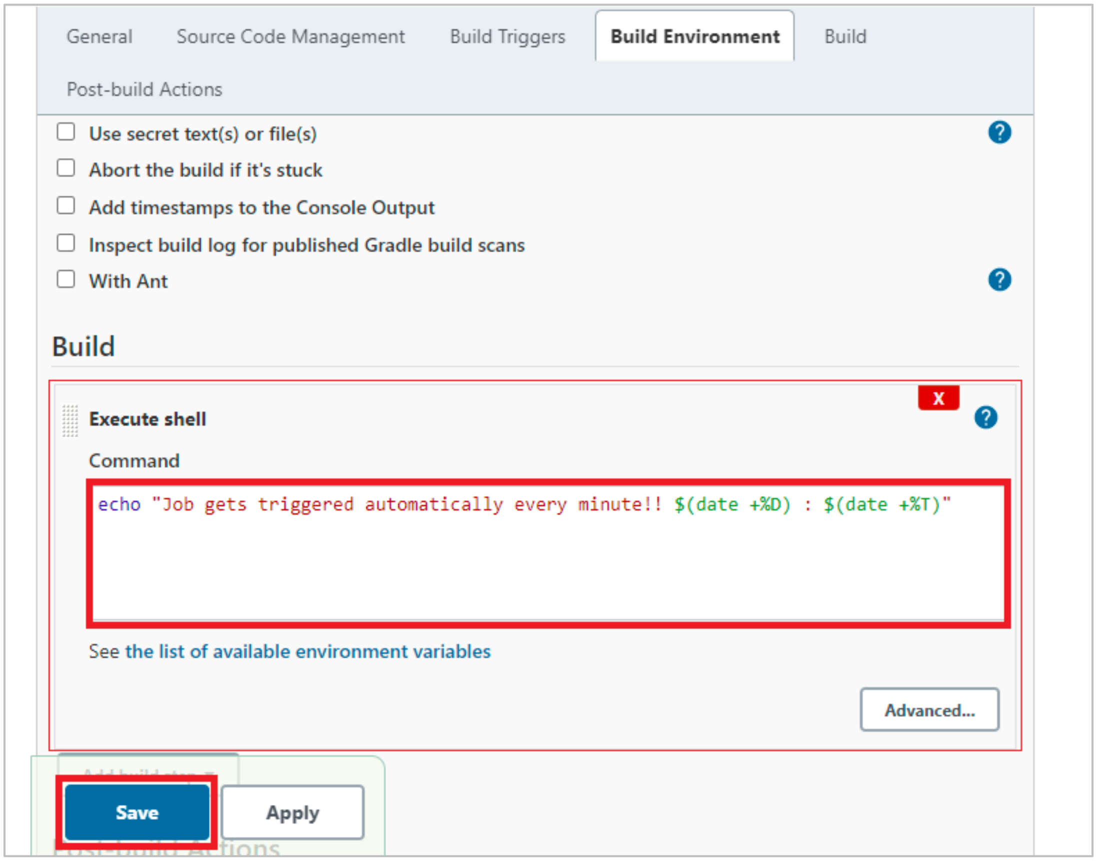
□ GitHub hook trigger for GITScm polling

□ Poll SCM

e. Scroll down to the **Build** section, click **Add build step** and select **Execute shell** as shown below.

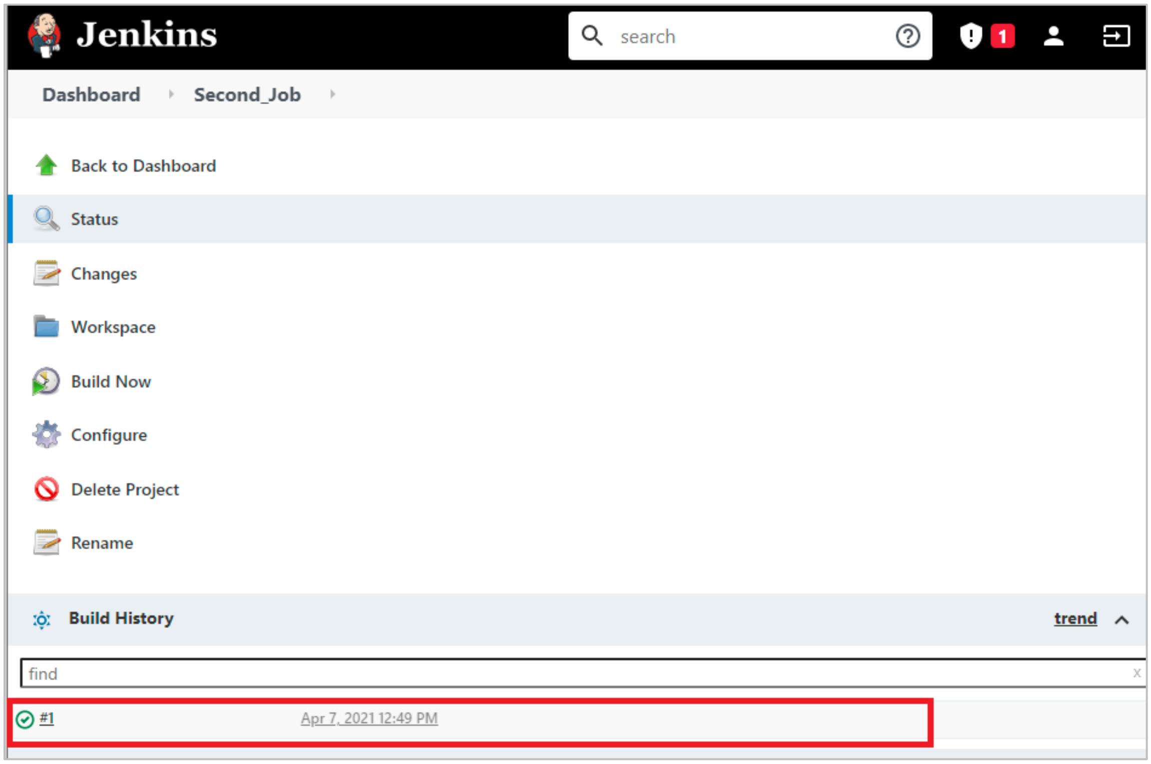

Dashboard
Second_Job
General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
□ Use secret text(s) or file(s)
□ Abort the build if it's stuck
□ Add timestamps to the Console Output
□ Inspect build log for published Gradle build scans
□ With Ant
Build
Add build step
Execute Windows batch command
Execute shell
Invoke Ant
Invoke Gradle script
Invoke top-level Maven targets
Run with timeout
Set build status to "pending" on GitHub commit

f. In **Execute shell** provide the command to execute periodically and click **Save** as shown below.


General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
□ Use secret text(s) or file(s)
□ Abort the build if it's stuck
□ Add timestamps to the Console Output
□ Inspect build log for published Gradle build scans
□ With Ant
Build
X
Execute shell
Command
See the list of available environment variables
Advanced...
Save
Apply

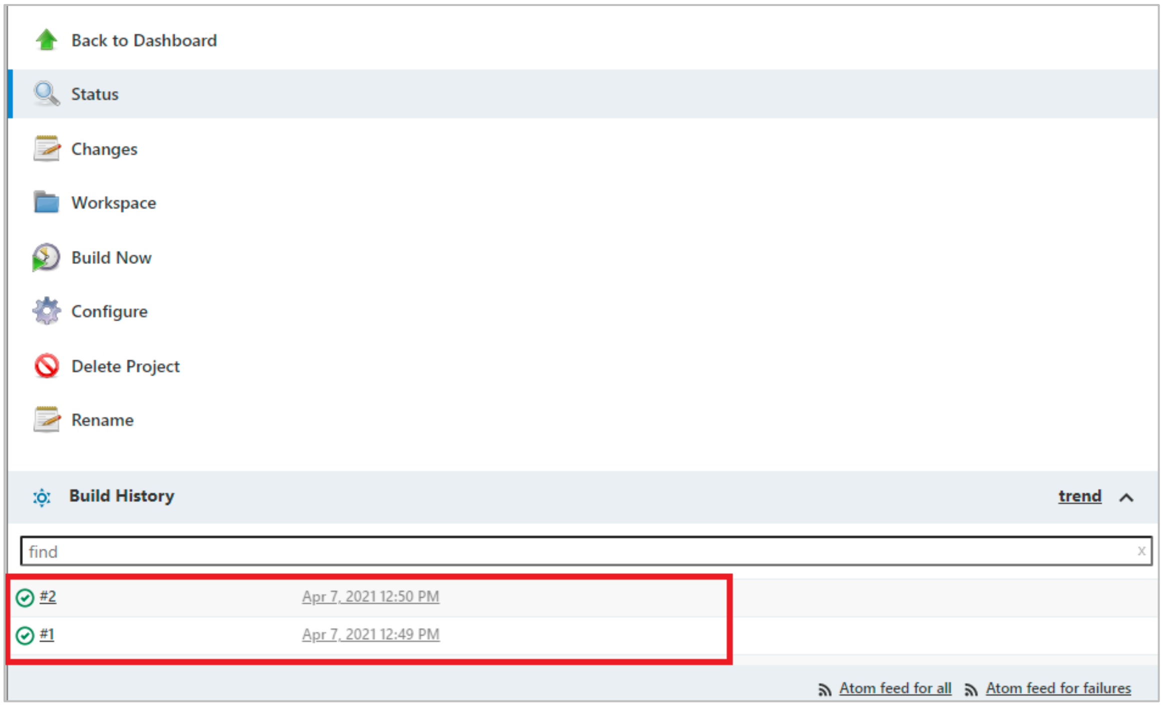
```
echo "Job gets triggered automatically every minute!! $(date +%D) : $(date +%T)"
```

g. After a minute of time the Job gets executed automatically. Verify it under **Build History** as shown below.



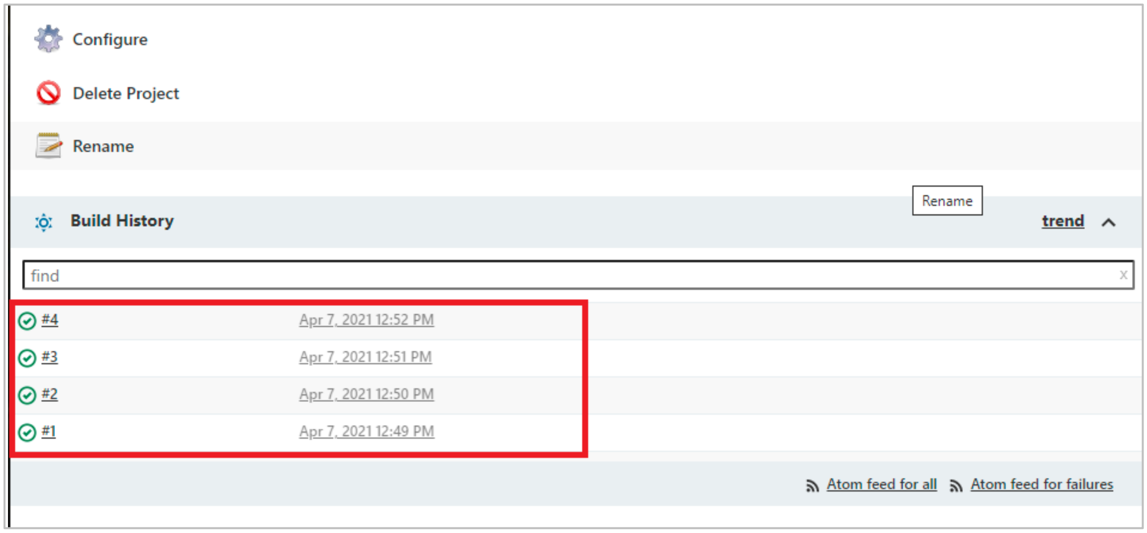
#1 Apr 7, 2021 12:49 PM

h. Exactly after a minute, job is executed (Build) for the second time as shown below.

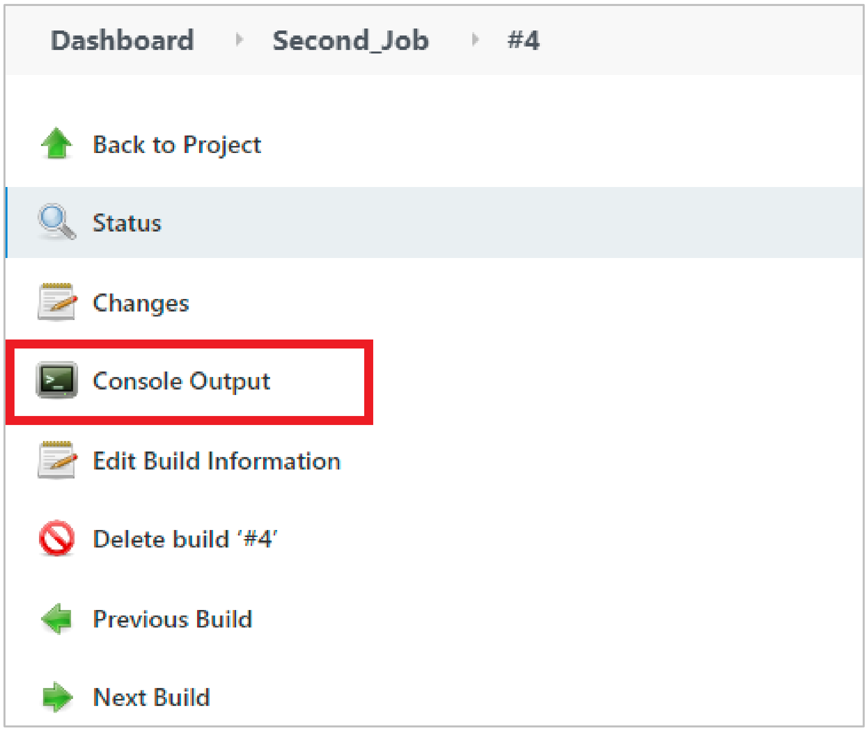


#2 Apr 7, 2021 12:50 PM
#1 Apr 7, 2021 12:49 PM

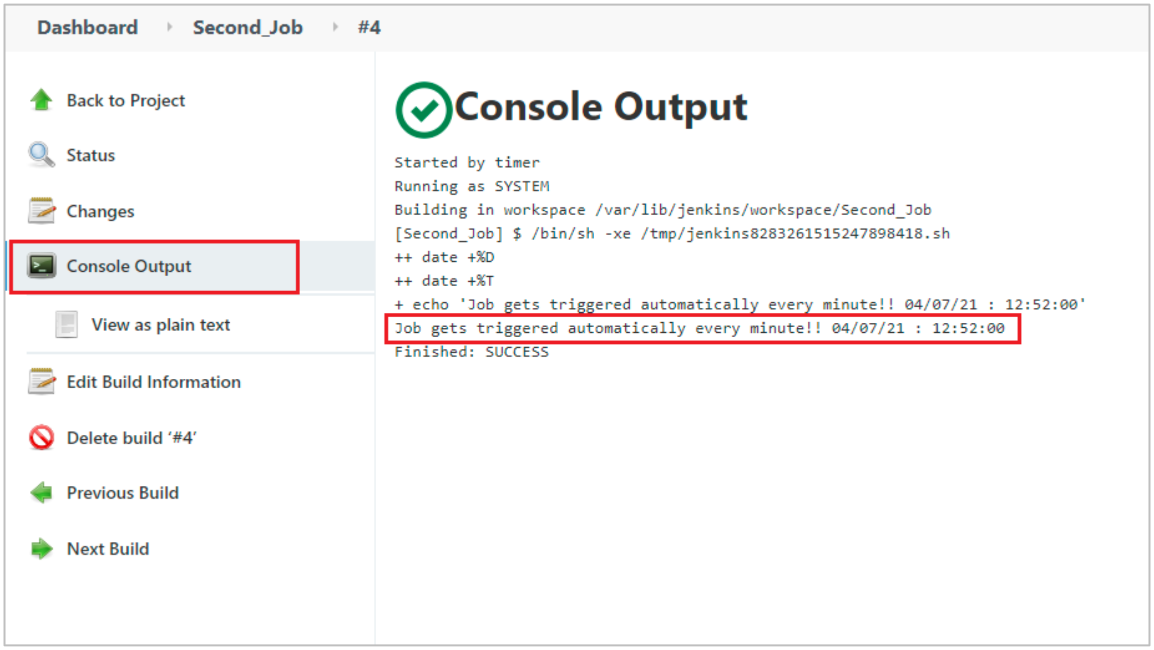
i. The job is built for every minute as shown below. Select any of the build to verify the output in the Jenkins Dashboard.



j. In the build page, select **Console Output** as shown below.



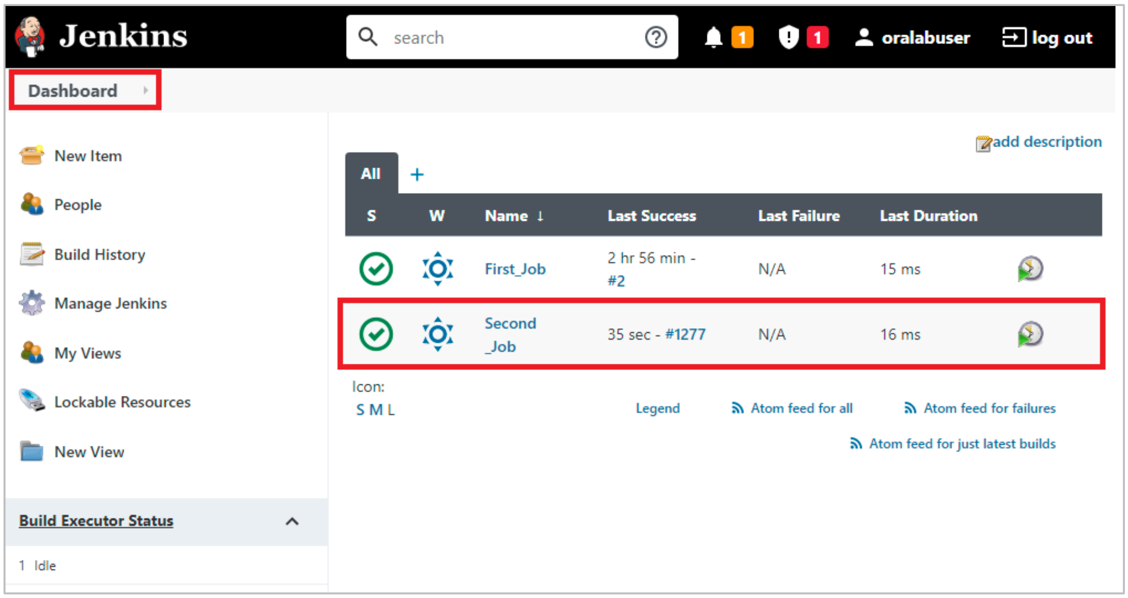
k. Verify the output displayed, which is executed by the shell command on scheduling the job in the Jenkins.



Console Output

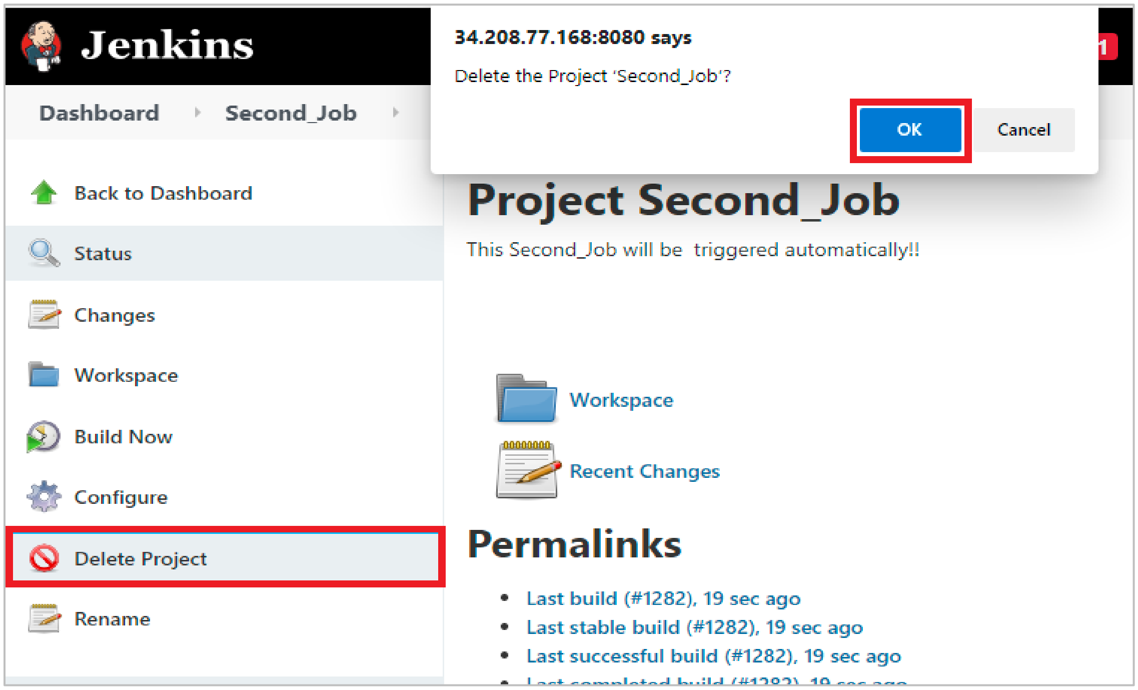
```
Started by timer
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/Second_Job
[Second_Job] $ /bin/sh -xe /tmp/jenkins8283261515247898418.sh
++ date +%D
++ date +%T
+ echo 'Job gets triggered automatically every minute!! 04/07/21 : 12:52:00'
Job gets triggered automatically every minute!! 04/07/21 : 12:52:00
Finished: SUCCESS
```

2. Delete the project created for the Job schedule in Jenkins instance.
    a. Navigate to the **Dashboard** in the top left corner as shown below, and select the **Second_Job** to delete.



S	W	Name ↓	Last Success	Last Failure	Last Duration	
✓	☀	First_Job	2 hr 56 min - #2	N/A	15 ms	
✓	☀	Second_Job	35 sec - #1277	N/A	16 ms	

    b. In the **Project** page navigate to menu, select **Delete Project** and click **OK** as shown below.



34.208.77.168:8080 says

Delete the Project 'Second_Job'?

Project Second_Job

This Second_Job will be triggered automatically!!

Permalinks

- Last build (#1282), 19 sec ago
- Last stable build (#1282), 19 sec ago
- Last successful build (#1282), 19 sec ago
- Last completed build (#1282), 19 sec ago

3. Keep the Jenkins Dashboard and the AWS Management Console open for the next practice.