



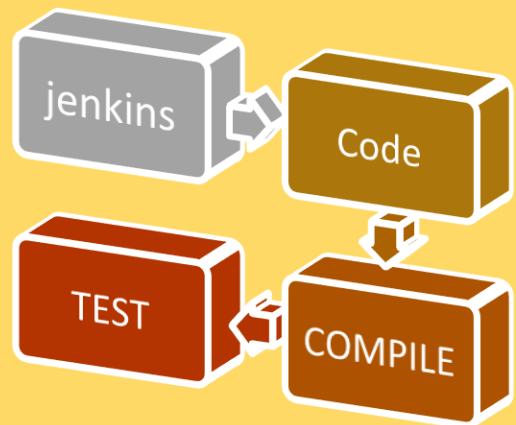
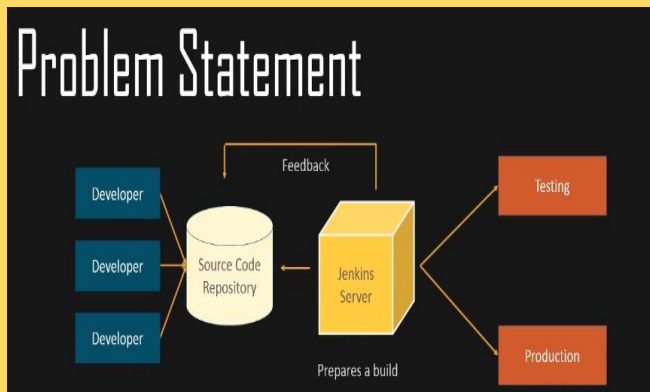
## Project: continuous integration pipeline using Jenkins

prepared by  
**Aya Rabi**

Linked 

 GitHub

# What is our plan?



## Step 1: install Jenkins

- ❖ We will start with Jenkins install
- ❖ You should have java version 11 or 16 in pc
- ❖ I will use windows by the way OS in URL <https://localhost:9090/>
- ❖ You will download Jenkins and start to install it and when you complete to install it will open you page to with address to get your password authentication

### Getting Started

## Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log ([not sure where to find it?](#)) and this file on the server:

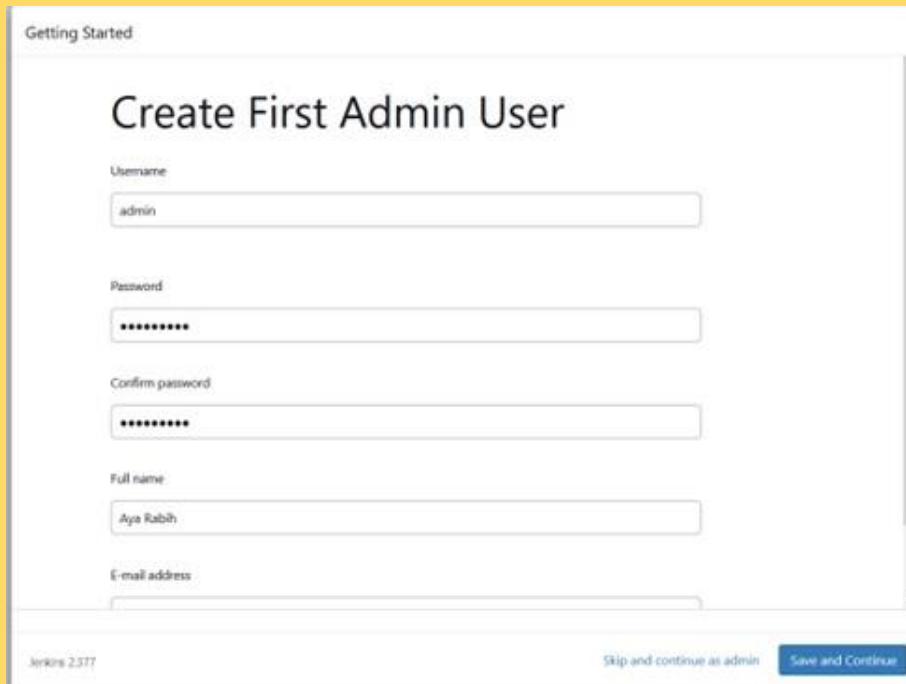
```
/home/ubuntu/.jenkins/secrets/initialAdminPassword
```

Please copy the password from either location and paste it below.

Administrator password

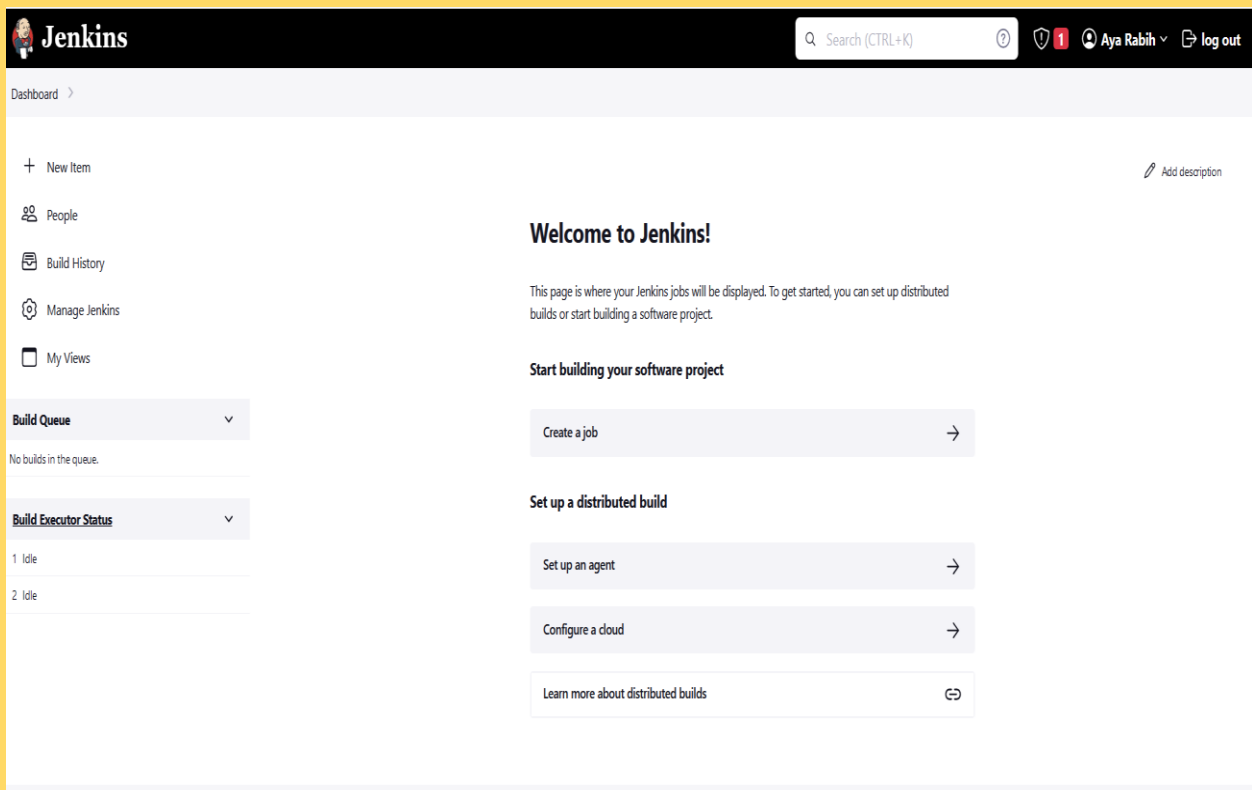
Continue

- ❖ Don't worry you will change password in this steps



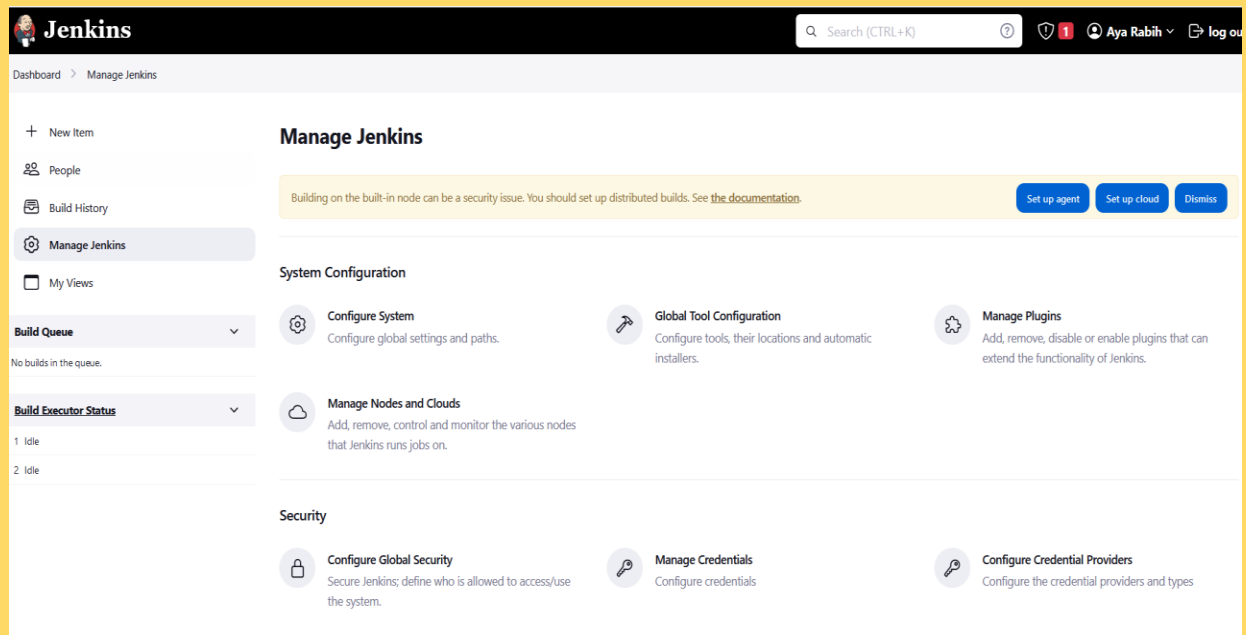
The image shows the 'Getting Started' page of Jenkins, specifically the 'Create First Admin User' form. The form has five input fields: 'Username' (containing 'admin'), 'Password' (containing seven asterisks), 'Confirm password' (containing seven asterisks), 'Full name' (containing 'Aya Rubih'), and 'E-mail address' (empty). At the bottom left, it says 'Jenkins 2.377'. At the bottom right, there are two buttons: 'Skip and continue as admin' and 'Save and Continue'.

- ❖ This is the first page you will see it in our project we need to install html plugins



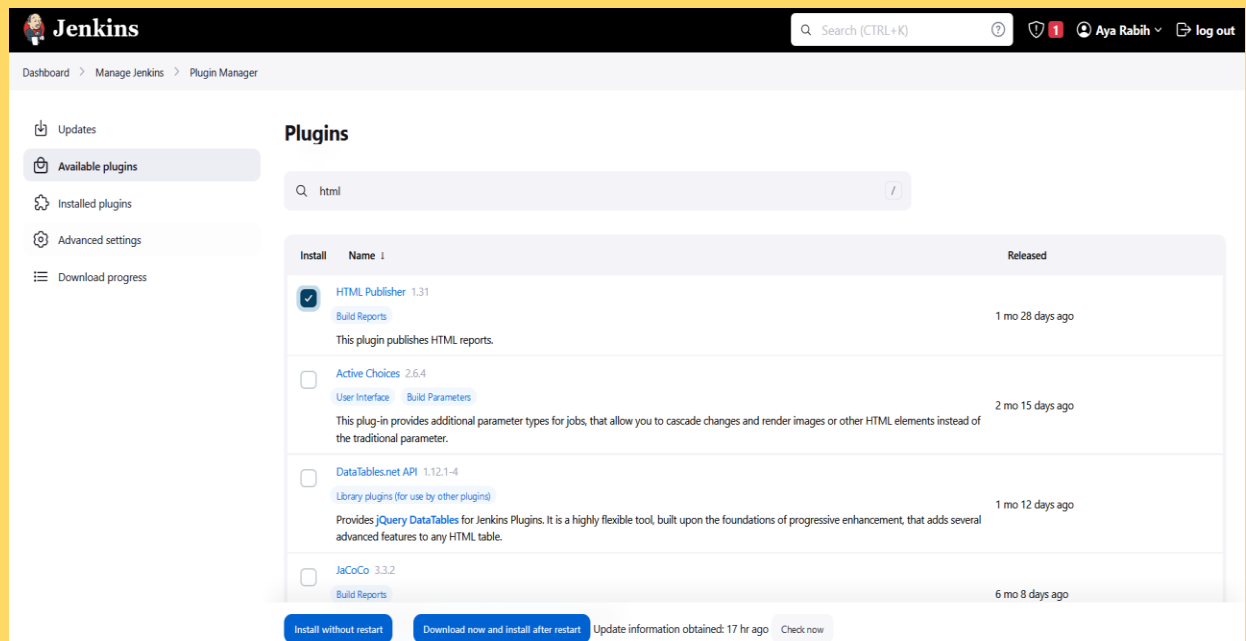
The image shows the Jenkins Dashboard. At the top, there's a black header with the Jenkins logo, a search bar (Search (CTRL+K)), and user information (Aya Rubih) with a 'log out' button. Below the header, the 'Dashboard' section is visible. On the left, there's a sidebar with links: '+ New Item', 'People', 'Build History', 'Manage Jenkins', and 'My Views'. Below these are two expandable sections: 'Build Queue' (showing 'No builds in the queue.') and 'Build Executor Status' (showing '1 Idle' and '2 Idle'). The main content area has a 'Welcome to Jenkins!' message, followed by a description of the page's purpose. Below this, there are three sections: 'Start building your software project' with a 'Create a job' button, 'Set up a distributed build' with 'Set up an agent' and 'Configure a cloud' buttons, and a link to 'Learn more about distributed builds'.

1. click to manage jenkins
2. click to manage plugins



The screenshot shows the Jenkins 'Manage Jenkins' dashboard. The left sidebar contains navigation links: New Item, People, Build History, Manage Jenkins (selected), and My Views. Below these are sections for 'Build Queue' (showing no builds) and 'Build Executor Status' (showing 2 idle executors). The main content area is titled 'Manage Jenkins' and includes a warning banner about distributed builds. It is organized into three sections: 'System Configuration' with links for Configure System, Global Tool Configuration, and Manage Plugins; 'Manage Nodes and Clouds'; and 'Security' with links for Configure Global Security, Manage Credentials, and Configure Credential Providers.

1. click to available plugins
2. write html
3. Click to install without restart

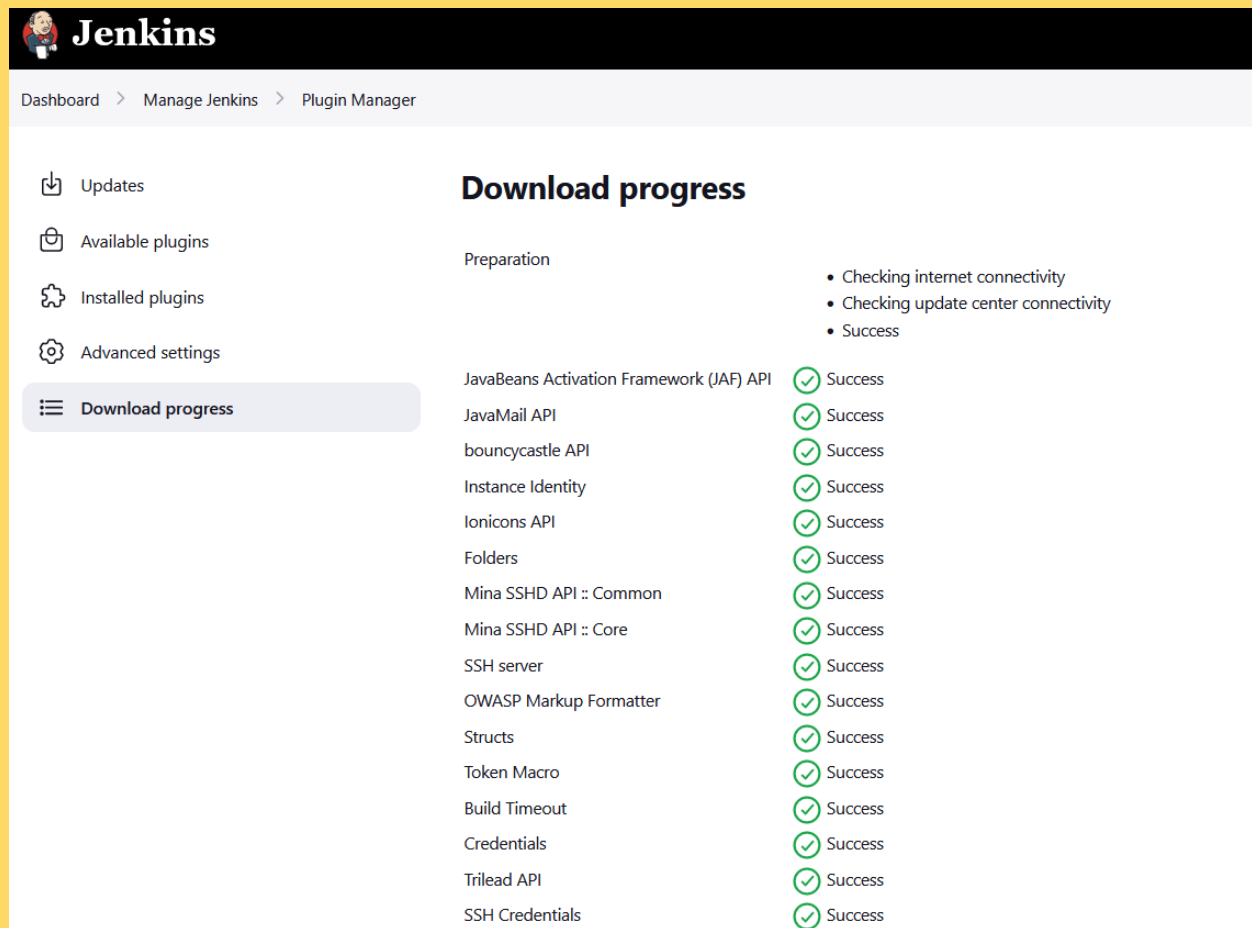


The screenshot shows the Jenkins 'Plugin Manager' interface. The left sidebar has links for Updates, Available plugins (selected), Installed plugins, Advanced settings, and Download progress. The main content area is titled 'Plugins' and features a search bar with 'html' entered. Below the search bar is a table of available plugins:

Install	Name	Released
<input checked="" type="checkbox"/>	<b>HTML Publisher</b> 1.31 Build Reports This plugin publishes HTML reports.	1 mo 28 days ago
<input type="checkbox"/>	<b>Active Choices</b> 2.6.4 User Interface Build Parameters This plug-in provides additional parameter types for jobs, that allow you to cascade changes and render images or other HTML elements instead of the traditional parameter.	2 mo 15 days ago
<input type="checkbox"/>	<b>DataTables.net API</b> 1.12.1-4 Library plugins (for use by other plugins) Provides <b>jQuery DataTables</b> for Jenkins Plugins. It is a highly flexible tool, built upon the foundations of progressive enhancement, that adds several advanced features to any HTML table.	1 mo 12 days ago
<input type="checkbox"/>	<b>JaCoCo</b> 3.3.2 Build Reports	6 mo 8 days ago

At the bottom, there are buttons for 'Install without restart' and 'Download now and install after restart', along with a status message: 'Update information obtained: 17 hr ago Check now'.

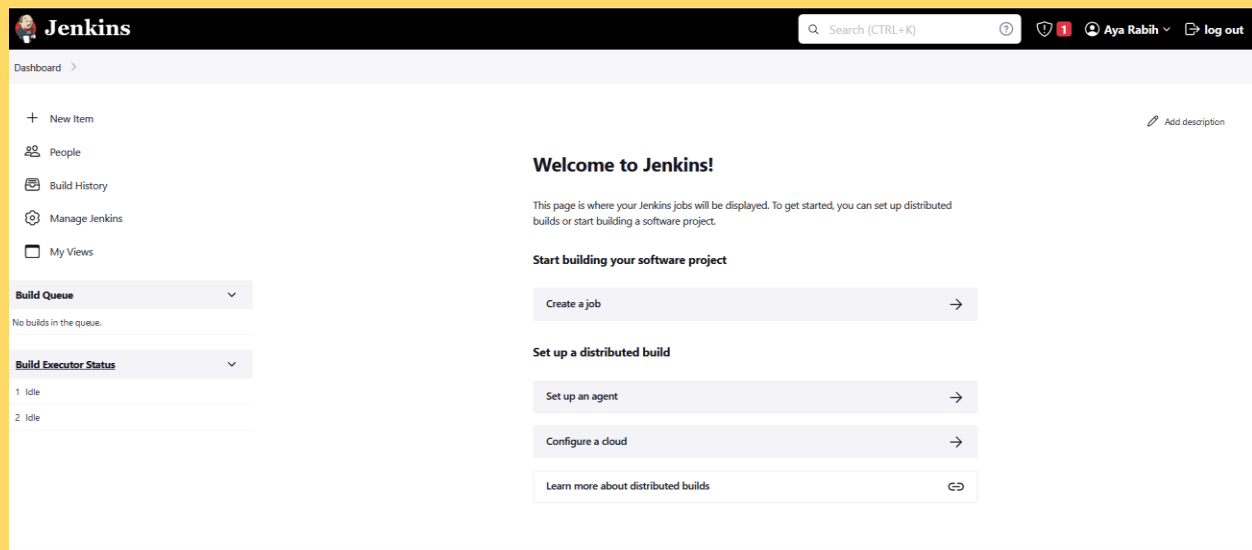
- ❖ Now it will install all html plugins



The screenshot shows the Jenkins 'Plugin Manager' interface. The left sidebar contains navigation links: Updates, Available plugins, Installed plugins, Advanced settings, and Download progress (which is highlighted). The main area is titled 'Download progress' and shows a list of plugins being installed, each with a green checkmark and the word 'Success'. The plugins listed are: JavaBeans Activation Framework (JAF) API, JavaMail API, bouncycastle API, Instance Identity, Ionicons API, Folders, Mina SSHD API :: Common, Mina SSHD API :: Core, SSH server, OWASP Markup Formatter, Struts, Token Macro, Build Timeout, Credentials, Trilead API, and SSH Credentials. Above the list, under the 'Preparation' section, there is a bulleted list: 'Checking internet connectivity', 'Checking update center connectivity', and 'Success'.

Plugin	Status
JavaBeans Activation Framework (JAF) API	Success
JavaMail API	Success
bouncycastle API	Success
Instance Identity	Success
Ionicons API	Success
Folders	Success
Mina SSHD API :: Common	Success
Mina SSHD API :: Core	Success
SSH server	Success
OWASP Markup Formatter	Success
Struts	Success
Token Macro	Success
Build Timeout	Success
Credentials	Success
Trilead API	Success
SSH Credentials	Success

- ❖ We will create new item to start our project



The screenshot shows the Jenkins 'Dashboard'. The left sidebar has links for: New Item, People, Build History, Manage Jenkins, and My Views. The main area features a 'Welcome to Jenkins!' message with a sub-header 'Start building your software project'. Below this, there are three buttons: 'Create a job', 'Set up a distributed build', and 'Configure a cloud'. At the bottom, there is a link 'Learn more about distributed builds'. The 'Build Queue' section on the left shows 'No builds in the queue.' and the 'Build Executor Status' section shows two 'Idle' executors.

**Welcome to Jenkins!**

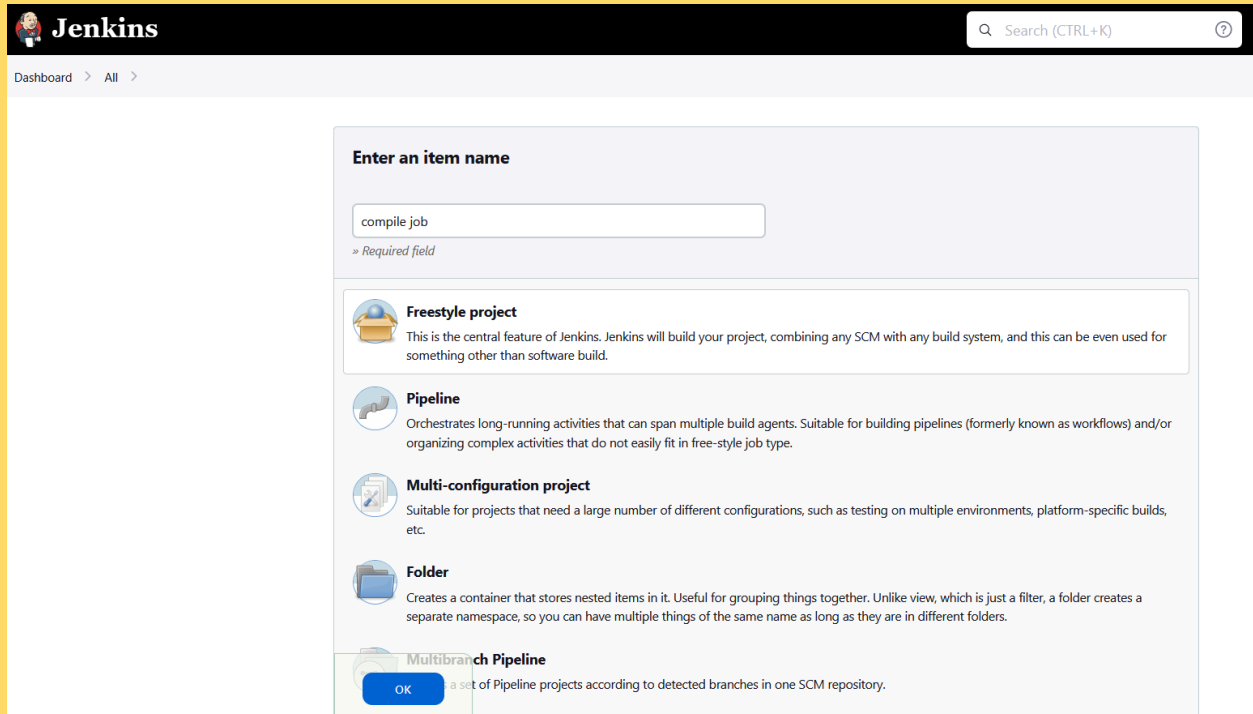
This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.

**Start building your software project**

- Create a job →
- Set up a distributed build →
- Configure a cloud →
- Learn more about distributed builds ↗

## Step 2: create compile job

1. We will name our project compile job or any thing you wanna
2. choice freestyle project
3. Ok

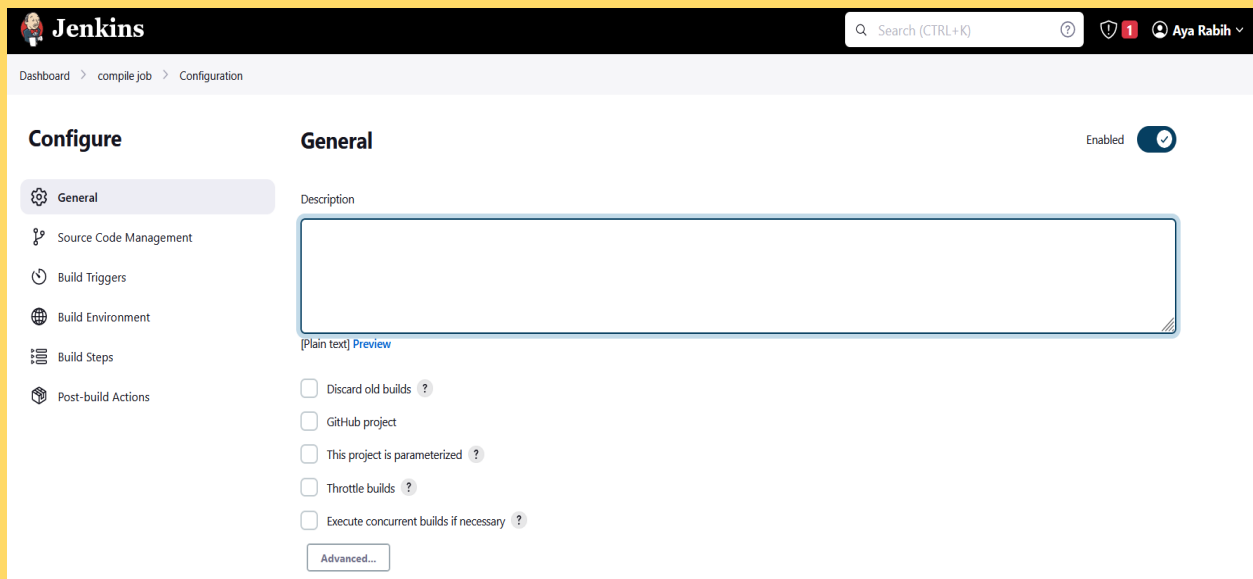


The screenshot shows the Jenkins 'New Item' page. At the top, there's a search bar and a 'Jenkins' logo. Below the header, the breadcrumb 'Dashboard > All >' is visible. The main section is titled 'Enter an item name' and contains a text input field with 'compile job' entered. Below the input field is a note '» Required field'. Underneath, there are four project type options, each with an icon and a description:

- 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.
- 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.

At the bottom, there's a 'Multibranch Pipeline' option with a description: 'Creates a set of Pipeline projects according to detected branches in one SCM repository.' and an 'OK' button.

❖ You will see this first page in configure page



The screenshot shows the Jenkins 'Configuration' page for the 'compile job'. The breadcrumb is 'Dashboard > compile job > Configuration'. The page is titled 'Configure' and has a 'General' tab selected. On the right, there's a status 'Enabled' with a checkmark icon. The 'General' tab has a 'Description' field with a large text area. Below the description field, there are several checkboxes with question marks for help:

- ☐ Discard old builds ?
- ☐ GitHub project
- ☐ This project is parameterized ?
- ☐ Throttle builds ?
- ☐ Execute concurrent builds if necessary ?

At the bottom, there's an 'Advanced...' button.

1. <https://github.com/upasanatestgit/demo>
2. we will use this repository to clone it in jenkins

Dashboard > compile job > Configuration

### Configure

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

#### Git

Repositories

Repository URL:

Credentials:

[+ Add](#) [Advanced...](#)

[Add Repository](#)

Branches to build

Branch Specifier (blank for 'any'):

[Add Branch](#)

[Save](#) [Apply](#)

1. Click to build
2. Select invoke top-level maven targets

Select

### Configure

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

#### Build Environment

- ☐ 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

#### Build Steps

[Add build step ^](#)

Filter

- 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

[Save](#) [Apply](#)

1. We will write compile
2. Save

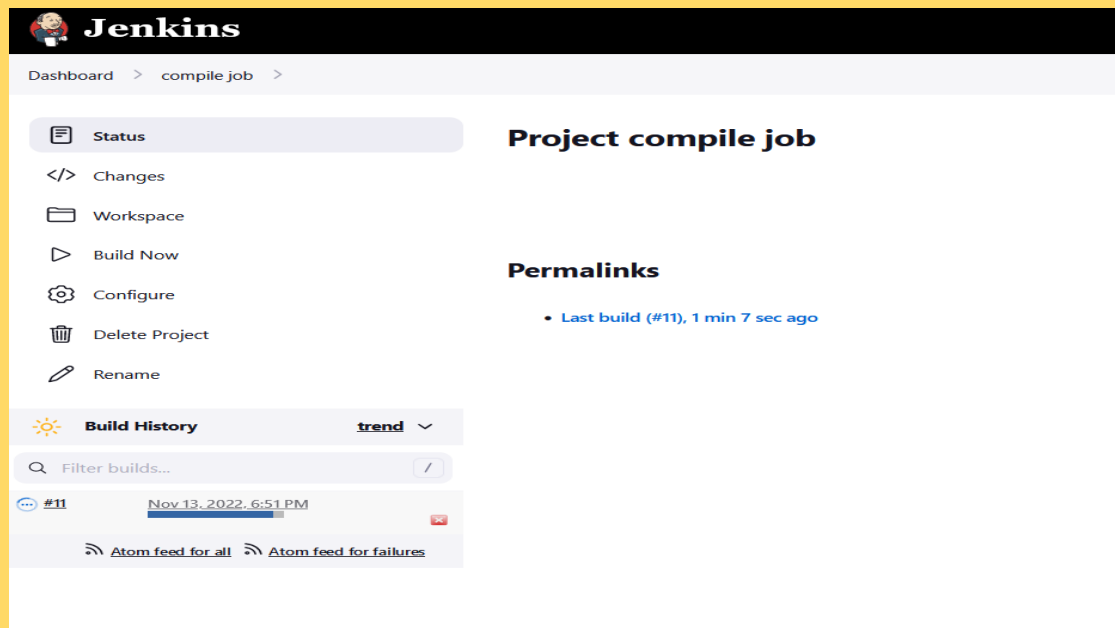
The screenshot shows the Jenkins 'Configure' page for a job named 'compile'. On the left is a sidebar with navigation links: General, Source Code Management, Build Triggers, Build Environment (selected), Build Steps, and Post-build Actions. The main content area is divided into sections. At the top, there are three unchecked checkboxes: 'Add timestamps to the console output', 'Inspect build log for published build scans', and 'Terminate a build if it's stuck'. Below these is a checkbox for 'With Ant' with a help icon. The 'Build Steps' section contains a single step named 'Invoke top-level Maven targets' with a help icon and a red close button. Inside this step's configuration, there is a 'Goals' field with 'compile' entered and a dropdown arrow, and an 'Advanced...' button. Below the step list is an 'Add build step' button. The 'Post-build Actions' section has an 'Add post-build action' button. At the bottom of the page are 'Save' and 'Apply' buttons.

❖ Now we will click to build now

The screenshot shows the Jenkins dashboard for a job named 'compile job'. The top navigation bar includes the Jenkins logo, a search bar, and user information for 'Aya Rabih' with a 'log out' link. The breadcrumb trail shows 'Dashboard > compile job >'. On the left sidebar, there are links for Status, Changes, Workspace, Build Now (highlighted), Configure, Delete Project, and Rename. The main content area is titled 'Project compile job' and includes an 'Add description' link and a 'Disable Project' button. Below this is a 'Permalinks' section. At the bottom, there is a 'Build History' section with a 'trend' dropdown, a 'Filter builds...' input, and a list of builds. The first build is dated 'Nov 13, 2022 5:02 PM' and has a status icon. Below the build list are links for 'Atom feed for all' and 'Atom feed for failures'.

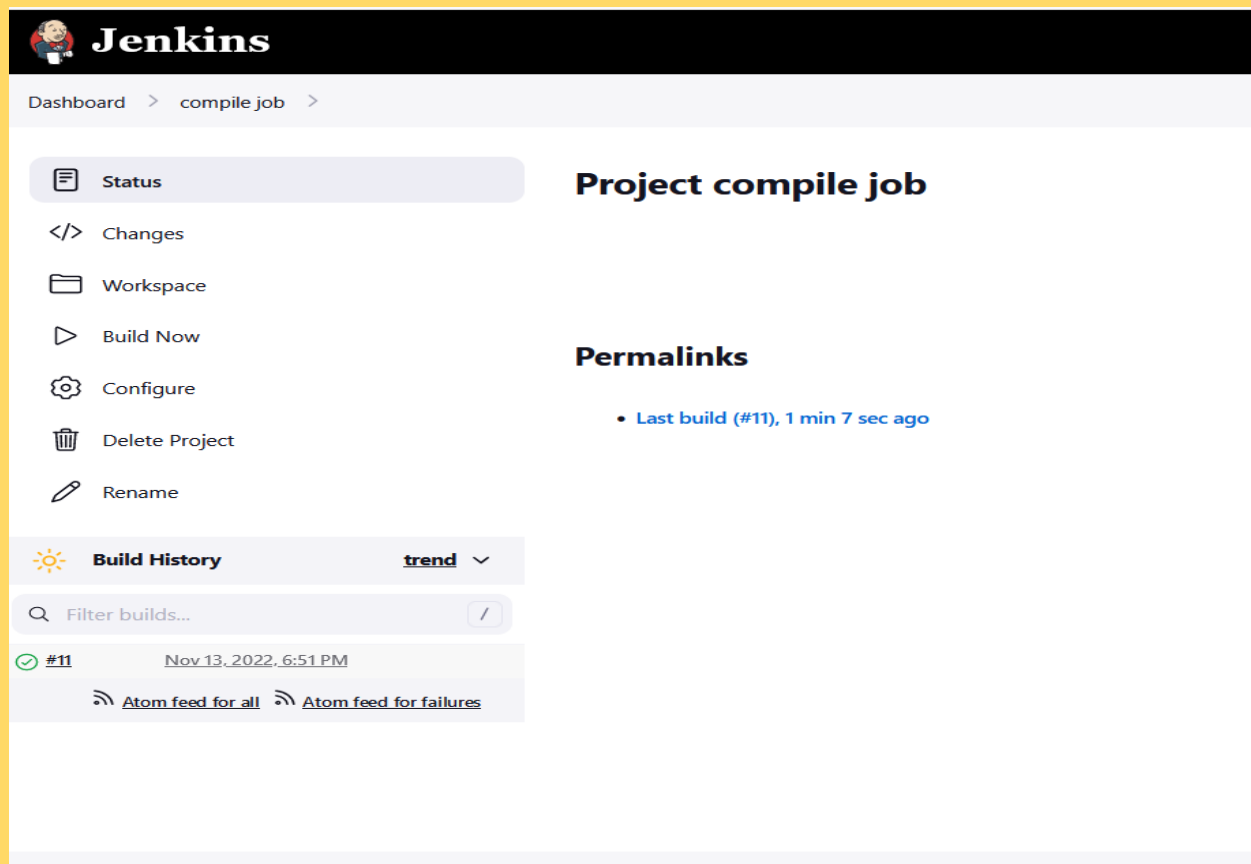


- ❖ These steps run compile job



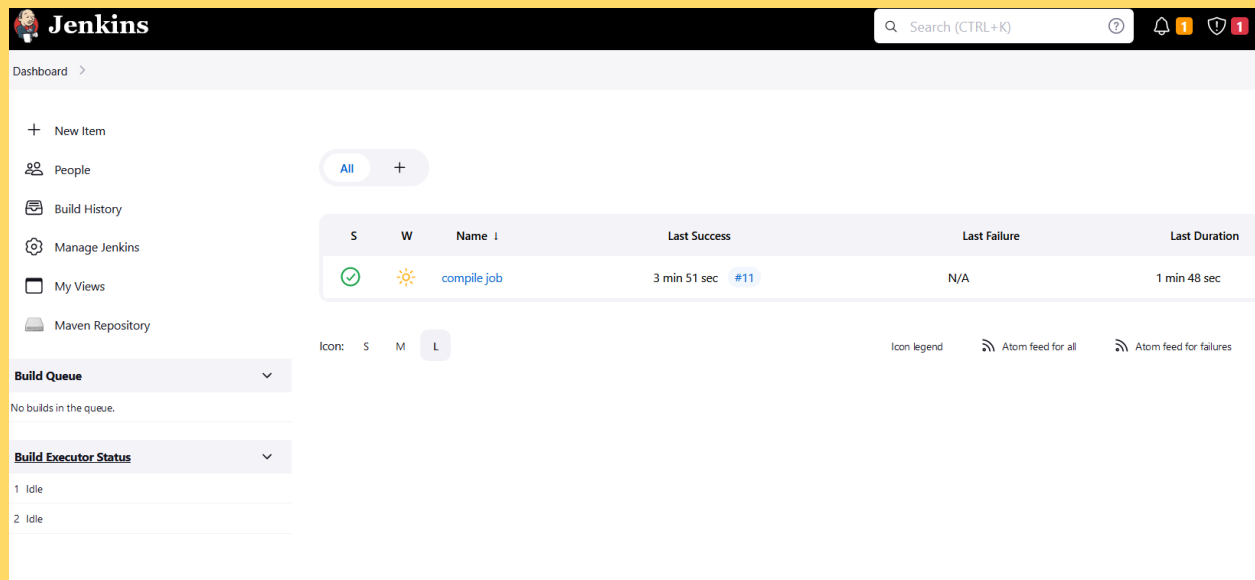
The screenshot shows the Jenkins web interface for a project named "Project compile job". The left sidebar contains navigation links: Status, Changes, Workspace, Build Now, Configure, Delete Project, and Rename. The main content area has a "Permalinks" section with a link for the last build (#11), which occurred 1 minute and 7 seconds ago. Below this, the "Build History" section shows a single build (#11) with a status icon of two blue circles, indicating it is pending. The build timestamp is "Nov 13, 2022, 6:51 PM".

- ❖ Now it is green done this step of compile the code



This screenshot shows the same Jenkins project page, but the build status has changed. The "Build History" section now shows build #11 with a green checkmark icon, indicating it is completed. The timestamp remains "Nov 13, 2022, 6:51 PM". The "Permalinks" section still shows the link for the last build (#11), 1 min 7 sec ago.

- ❖ These steps show us compile job done

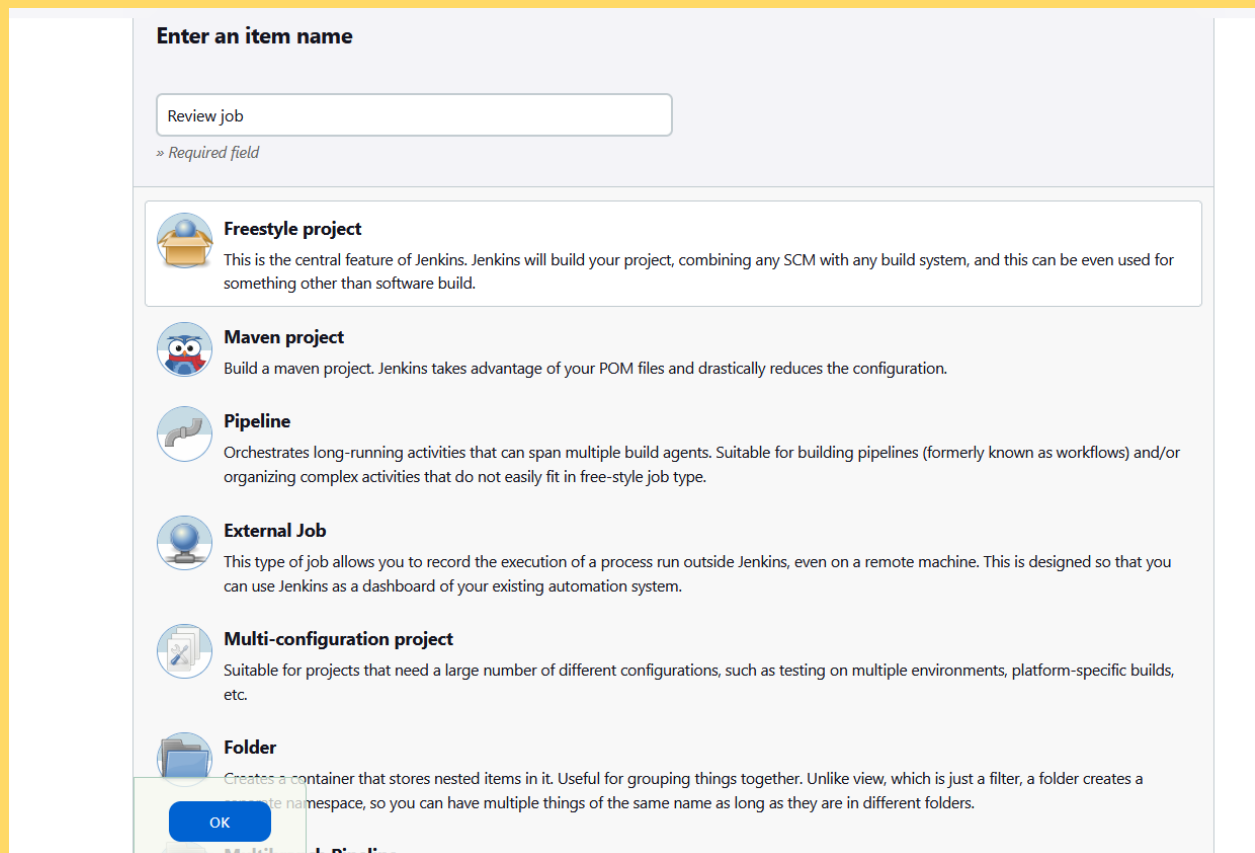


The screenshot shows the Jenkins Dashboard. On the left is a sidebar with navigation links: New Item, People, Build History, Manage Jenkins, My Views, and Maven Repository. Below these are sections for 'Build Queue' (showing 'No builds in the queue') and 'Build Executor Status' (showing two idle executors). The main area displays a table of jobs. A filter 'All' is selected. The table has columns: S (Status), W (Icon), Name, Last Success, Last Failure, and Last Duration. One job is listed: 'compile job' with a green checkmark status, a sun icon, a last success time of '3 min 51 sec' for build '#11', and a last duration of '1 min 48 sec'. Below the table are links for 'Icon legend', 'Atom feed for all', and 'Atom feed for failures'.

S	W	Name	Last Success	Last Failure	Last Duration
✓	☀	compile job	3 min 51 sec #11	N/A	1 min 48 sec

### Step 3: create Review job

- ❖ Now we need to testing so junkies will deploy the code onto a test server for testing
- ❖ We will make new item for testing name it Review job and click to freestyle project



The screenshot shows the 'Enter an item name' form in Jenkins. The 'Review job' name is entered in the text field. Below the field is a 'Required field' message. A list of job types is shown with icons and descriptions:

- 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.
- Maven project**: Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.
- 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.
- External Job**: This type of job allows you to record the execution of a process run outside Jenkins, even on a remote machine. This is designed so that you can use Jenkins as a dashboard of your existing automation system.
- 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.

An 'OK' button is visible at the bottom left of the form.

<https://github.com/upasanatestgit/Demo.git>

## Configure

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

### Source Code Management

☐ None

☒ Git ?

Repositories ?

Repository URL ? ✕

Credentials ?

▼

Branches to build ?

1. Invoke top-level Maven targets
2. Build Steps
3. -P matrix pmd:pmd

## Configure

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

### Build Steps

≡ **Invoke top-level Maven targets** ? ✕

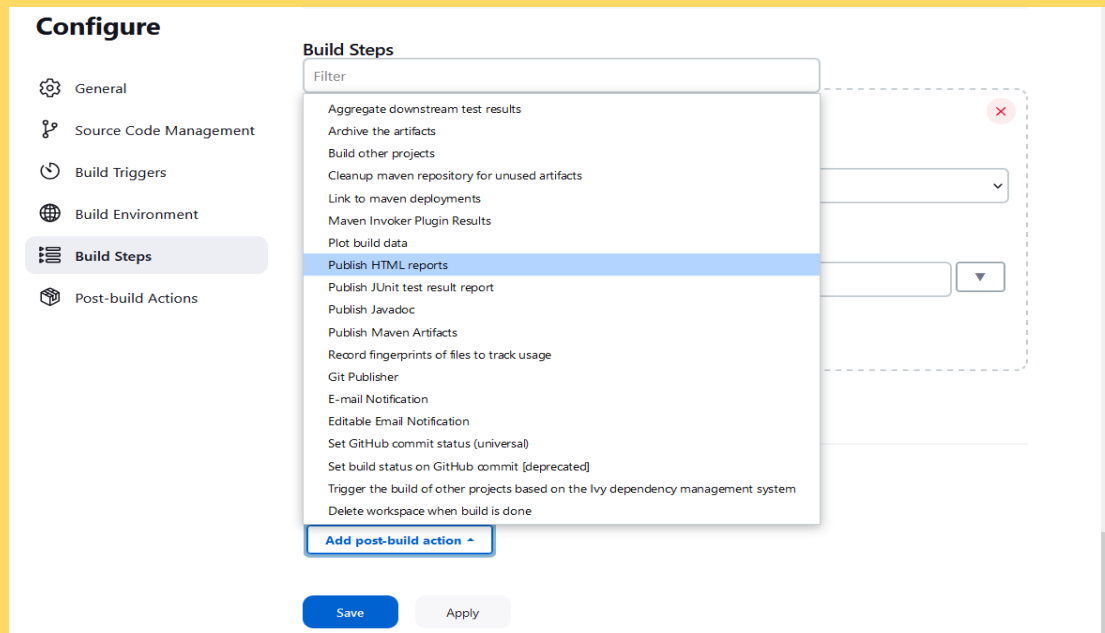
Maven Version

▼

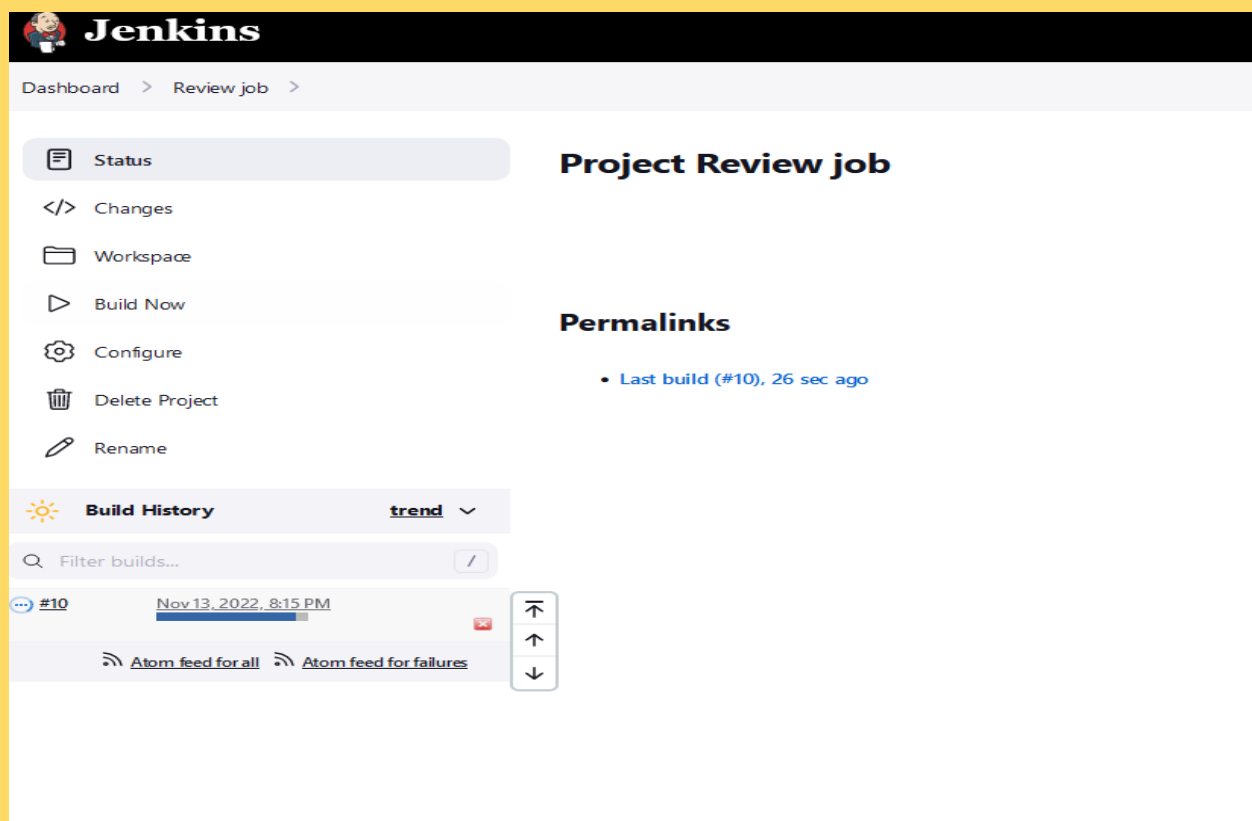
Goals

▼

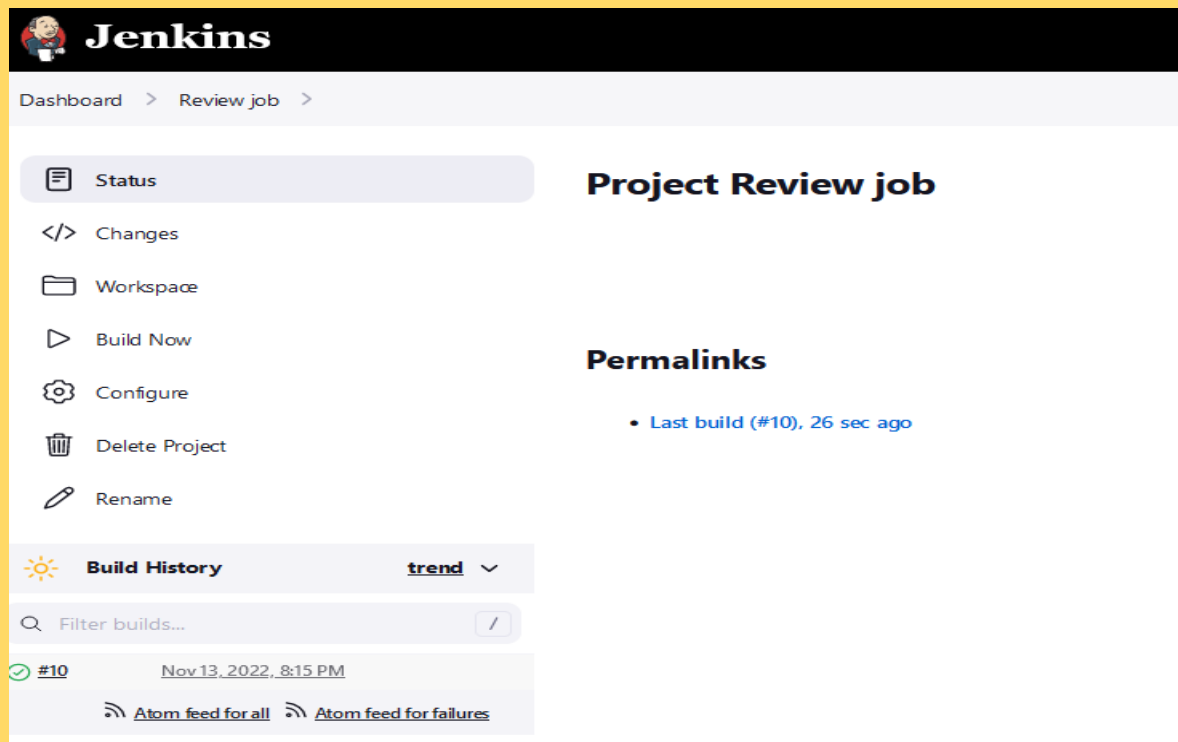
1. Post-build Actions
2. Publish HTML reports



❖ Now it will start run review job



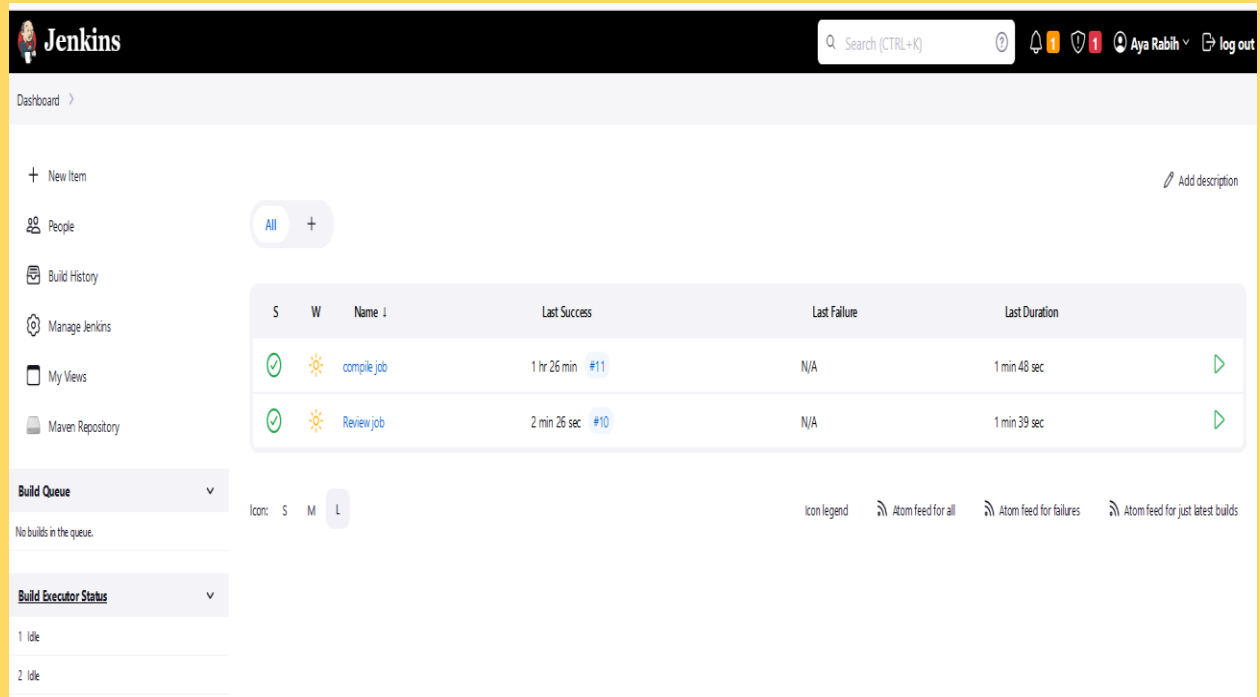
## ❖ Review job done



The screenshot shows the Jenkins 'Project Review job' page. The left sidebar contains a menu with options: Status (selected), Changes, Workspace, Build Now, Configure, Delete Project, and Rename. The main content area has a 'Permalinks' section with a link for 'Last build (#10), 26 sec ago'. Below this is the 'Build History' section, which includes a search bar and a table of build history. The table shows a single build, #10, completed on Nov 13, 2022, at 8:15 PM. The build status is 'Success' (green checkmark). The build duration is '26 sec'. The build is linked to the 'Review job'.

Build Number	Status	Timestamp
#10	Success	Nov 13, 2022, 8:15 PM

## ❖ Now we complete two job compile and review job

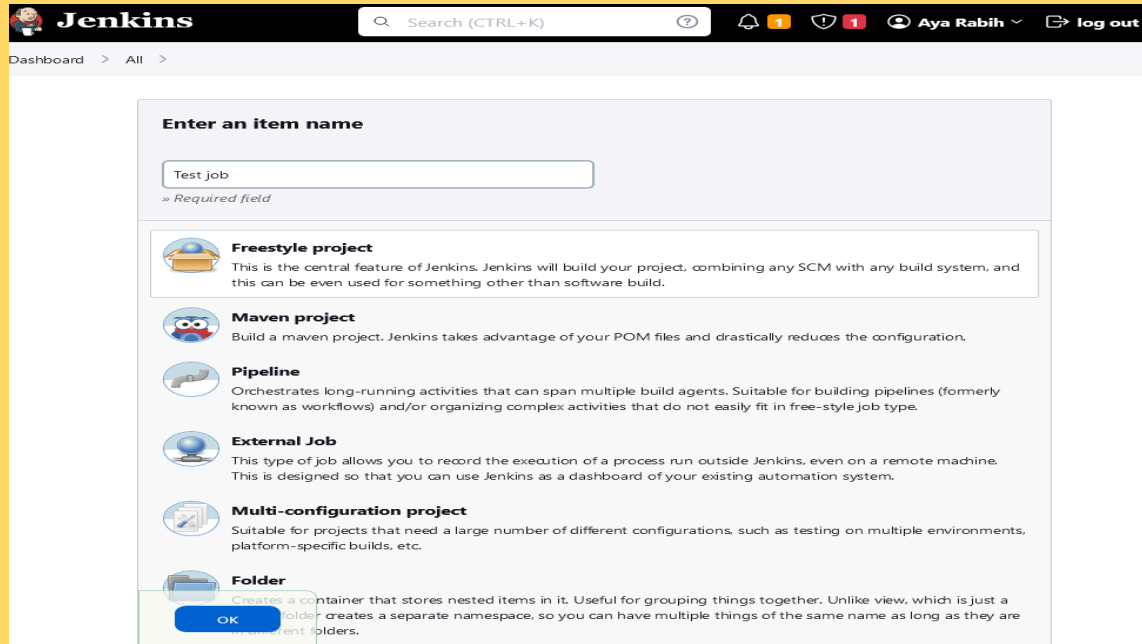


The screenshot shows the Jenkins Dashboard. The top navigation bar includes a search bar, a user profile (Aya Rabin), and a 'log out' button. The left sidebar contains a menu with options: New Item, People, Build History, Manage Jenkins, My Views, and Maven Repository. The main content area displays a table of build history. The table has columns for 'S' (Success), 'W' (Warning), 'Name', 'Last Success', 'Last Failure', and 'Last Duration'. The table shows two builds: 'compile job' and 'Review job'. The 'compile job' build is successful, with a duration of 1 min 48 sec. The 'Review job' build is also successful, with a duration of 1 min 39 sec. The 'Review job' build is linked to the 'Review job'.

S	W	Name	Last Success	Last Failure	Last Duration
✓	☀	compile job	1 hr 26 min #11	N/A	1 min 48 sec
✓	☀	Review job	2 min 26 sec #10	N/A	1 min 39 sec

## Step 4: create test job

1. Test job
2. Freestyle project
3. ok

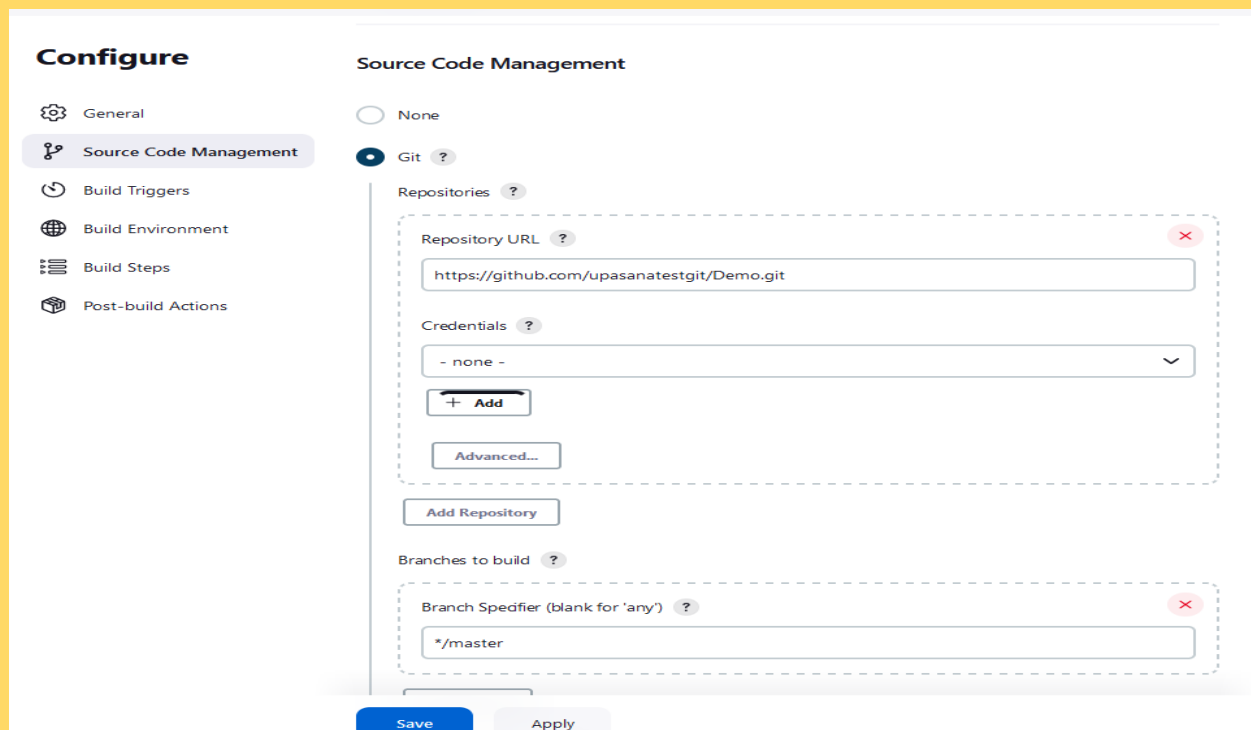


The image shows the Jenkins 'Enter an item name' dialog box. At the top, there's a search bar with the text 'Search (CTRL+K)'. Below it, the 'Enter an item name' section has a text input field containing 'Test job' and a note '» Required field'. Below this, there are several project type options, each with an icon and a description:

- 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.
- Maven project**: Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.
- 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.
- External Job**: This type of job allows you to record the execution of a process run outside Jenkins, even on a remote machine. This is designed so that you can use Jenkins as a dashboard of your existing automation system.
- 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 folder, creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

At the bottom left, there is a blue 'OK' button.

❖ <https://github.com/upasanatestgit/Demo.git>



The image shows the Jenkins 'Configure' page for 'Source Code Management'. On the left, there's a sidebar with navigation links: General, Source Code Management (selected), Build Triggers, Build Environment, Build Steps, and Post-build Actions. The main content area is titled 'Source Code Management' and has two radio buttons: 'None' and 'Git' (selected). Below the 'Git' button, there's a 'Repositories' section with a 'Repository URL' input field containing 'https://github.com/upasanatestgit/Demo.git'. Below this is a 'Credentials' dropdown menu showing '- none -' with an 'Add' button and an 'Advanced...' link. Below the 'Add' button is an 'Add Repository' button. Below the 'Add Repository' button is a 'Branches to build' section with a 'Branch Specifier (blank for 'any')' input field containing '\*/master'. At the bottom, there are 'Save' and 'Apply' buttons.

1. Build Steps
2. Invoke top-level Maven targets ?
3. Test

### Configure

General

Source Code Management

Build Triggers

Build Environment

**Build Steps**

Post-build Actions

☐ With Ant ?

#### Build Steps

Invoke top-level Maven targets ?

Maven Version  
maven01

Goals  
test

Advanced...

Add build step


#### Post-build Actions

Add post-build action





Save

Apply

❖ Now test job running

**Jenkins**

Search (CTRL+K)

 1  1  Aya Rabih  log out

Dashboard > Testjob >

Status

Changes

Workspace

Build Now

Configure

Delete Project

Rename

Build History

trend

Filter builds...

#1 Nov 13, 2022 8:25 PM

Atom feed for all Atom feed for failures

## Project Test job

Add description

Disable Project

### Permalinks

❖ Now it is done for test job

Dashboard > Test job >

**Project Test job**

**Permalinks**

**Build History** trend

Filter builds...

Status	Build Number	Timestamp	Actions
✓ #1	Nov 13, 2022, 8:25 PM	⬆️ ⬆️ ⬇️	

Atom feed for all Atom feed for failures

So we have now all steps done we need now to make pipeline for them

Dashboard >

**Build Queue**

No builds in the queue.

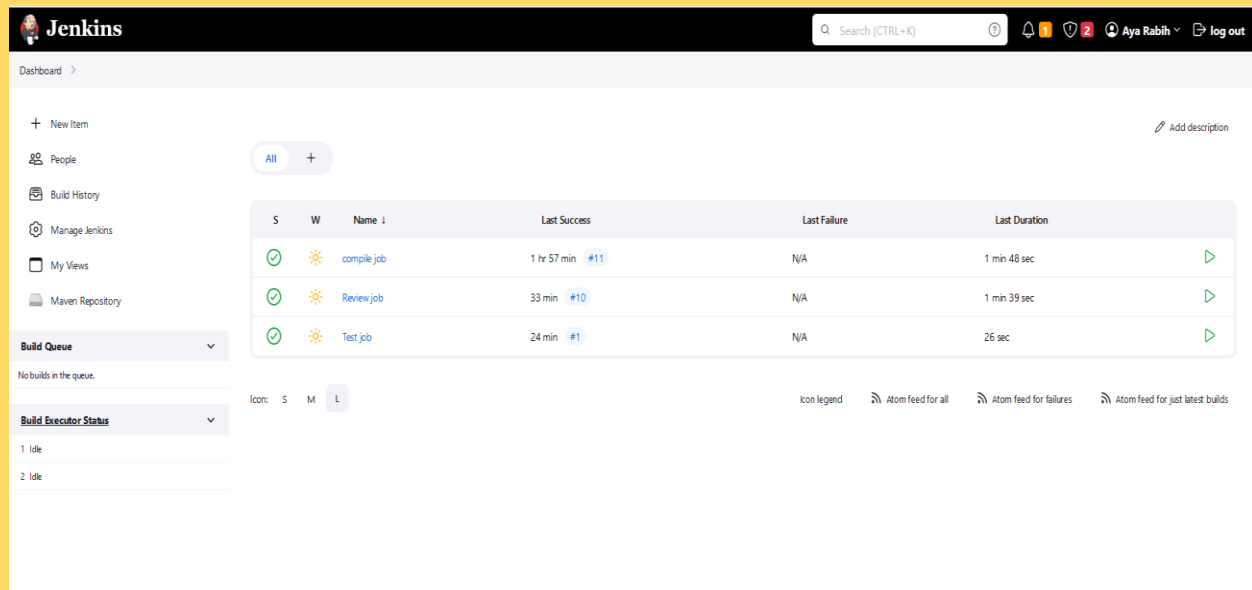
**Build Executor Status**

S	W	Name ↓	Last Success	Last Failure	Last Duration	Actions
✓	☀️	compile job	1 hr 34 min #11	N/A	1 min 48 sec	▶️
✓	☀️	Review job	10 min #10	N/A	1 min 39 sec	▶️
✓	☀️	Test job	50 sec #1	N/A	26 sec	▶️

Icon: S M L Icon legend Atom feed for all Atom feed for failures Atom feed for failures



- ❖ Now we will click to + to make pipeline

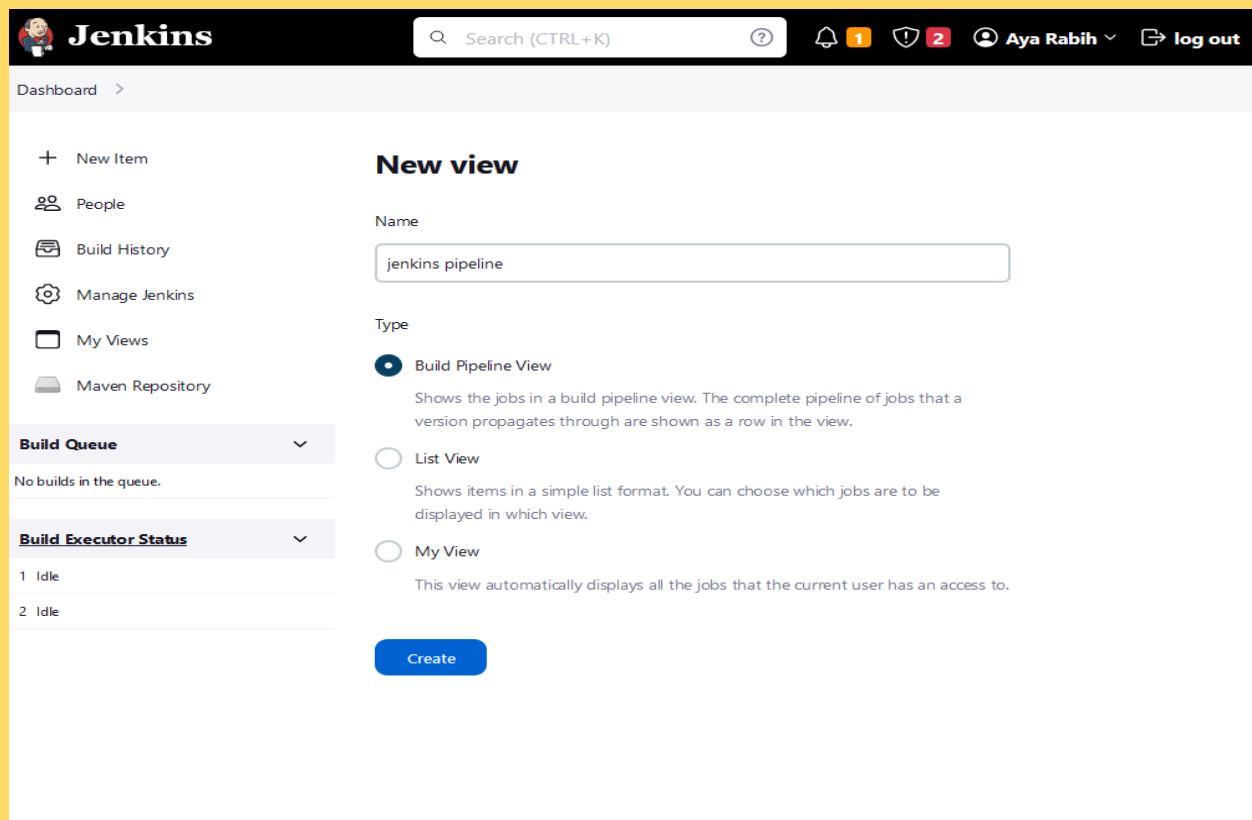


The screenshot shows the Jenkins Dashboard. On the left is a sidebar with navigation links: New Item, People, Build History, Manage Jenkins, My Views, and Maven Repository. Below these are sections for 'Build Queue' (showing 'No builds in the queue') and 'Build Executor Status' (showing two idle executors). The main area displays a table of build jobs with columns: S (Status), W (Webhook), Name, Last Success, Last Failure, and Last Duration. There are three jobs listed: 'compile job', 'Review job', and 'Test job', all with a success status and a duration. A '+ Add description' link is in the top right. At the bottom, there are links for 'Icon legend', 'Atom feed for all', 'Atom feed for failures', and 'Atom feed for just latest builds'.

S	W	Name	Last Success	Last Failure	Last Duration
✓	☀	compile job	1 hr 57 min #11	N/A	1 min 48 sec
✓	☀	Review job	33 min #10	N/A	1 min 39 sec
✓	☀	Test job	24 min #1	N/A	26 sec

## Step 5: create Jenkins pipeline

1. We will name this project Jenkins pipeline
2. Click to build pipeline view
3. create



The screenshot shows the 'New view' dialog in Jenkins. The 'Name' field is filled with 'jenkins pipeline'. The 'Type' section has three options: 'Build Pipeline View' (selected), 'List View', and 'My View'. Each option has a description. The 'Build Pipeline View' description says: 'Shows the jobs in a build pipeline view. The complete pipeline of jobs that a version propagates through are shown as a row in the view.' The 'List View' description says: 'Shows items in a simple list format. You can choose which jobs are to be displayed in which view.' The 'My View' description says: 'This view automatically displays all the jobs that the current user has an access to.' A blue 'Create' button is at the bottom.

**New view**

Name  
jenkins pipeline

Type

- ☒ Build Pipeline View  
Shows the jobs in a build pipeline view. The complete pipeline of jobs that a version propagates through are shown as a row in the view.
- ☐ List View  
Shows items in a simple list format. You can choose which jobs are to be displayed in which view.
- ☐ My View  
This view automatically displays all the jobs that the current user has an access to.

Create

- ❖ we will start with compile job

between jobs. This is the only out-of-the-box supported layout mode, but is open for extension.

---

### Upstream / downstream config

Select Initial Job ?

compile job

▼

- ❖ Select the No Of Displayed Builds from 1 to 5

---

### Display Options

No Of Displayed Builds ?

1

▼

- ❖ Like that

---


### Display Options

No Of Displayed Builds ?





5

▼

- ❖ Now click okay
- ❖ Now done for compile job in pipeline

**Jenkins**

Search (CTRL+K) ?

 1  2  Aya Rabih  log out

Dashboard > jenkins pipeline >

## Build Pipeline


Trigger a Pipeline Pipeline History Configure Add Step


Run History Configure Add Step Delete Manage


Pipeline

#11

#11 compile job




 Nov 13, 2022 6:51:57 PM

 1 min 48 sec


 admin



console re-run

❖ Now we will go to Review job


		<a href="#">Review job</a>	43 min #10	N/A	1 min 39 sec	
---	---	----------------------------	---------------	-----	--------------	---


❖ Click to configure


 **Jenkins**


Search (CTRL+K)   1


Dashboard > Review job >


 Status


 Changes

 Workspace

 Build Now

 Configure


 Delete Project

 Rename


## Project Review job



### Permalinks




- [Last build \(#10\), 45 min ago](#)
- [Last stable build \(#10\), 45 min ago](#)
- [Last successful build \(#10\), 45 min ago](#)
- [Last completed build \(#10\), 45 min ago](#)

 **Build History** trend v

Filter builds... /

 [#10](#) [Nov 13, 2022, 8:15 PM](#)

 [Atom feed for all](#)  [Atom feed for failures](#)



- ❖ Now we will click to

  1. Build Triggers
  2. Build after other projects are built
  3. Write compile job, this is the first job we make it
  4. Trigger even if the build fails
  5. Save

### Configure

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

### Build Triggers

☐ Trigger builds remotely (e.g., from scripts) ?

☒ Build after other projects are built ?




Projects to watch

compile job,

☐ Trigger only if build is stable  
☐ Trigger even if the build is unstable  
☒ Trigger even if the build fails  
☐ Always trigger, even if the build is aborted

☐ Build periodically ?  
☐ Enable Artifactory trigger  
☐ GitHub hook trigger for GITScm polling ?  
☐ Maven Dependency Update Trigger ?  
☐ Poll SCM ?

- ❖ Now we will go to test job



Test job
39 min
#1
N/A
26 sec


- ❖ Now test job done

Dashboard > Test job >

Status

Changes

Workspace

Build Now

Configure


Delete Project

Rename


## Project Test job

### Permalinks

- Last build (#1), 43 min ago
- Last stable build (#1), 43 min ago
- Last successful build (#1), 43 min ago
- Last completed build (#1), 43 min ago


**Build History**
trend v

#1 Nov 13, 2022, 8:25 PM

 [Atom feed for all](#)
 [Atom feed for failures](#)

1. Build Triggers
2. Build after other projects are built
3. Write your second job Review job,
4. Trigger even if the build fails
5. Save

### Configure

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

### Build Triggers

☐ Trigger builds remotely (e.g., from scripts) ?

☒ Build after other projects are built ?

Projects to watch

Review job,

☐ Trigger only if build is stable

☐ Trigger even if the build is unstable

☒ Trigger even if the build fails

☐ Always trigger, even if the build is aborted

☐ Build periodically ?

☐ Enable Artifactory trigger

☐ GitHub hook trigger for GITScm polling ?

☐ Maven Dependency Update Trigger ?

☐ Poll SCM ?

- ❖ Now let's see my pipeline from dashboard
- ❖ Click to Jenkins pipeline

# Jenkins

1
 2
 Aya Rabi
log out

Dashboard >

+ New Item

Add description

- People
- Build History
- Manage Jenkins
- My Views
- Maven Repository

Build Queue

No builds in the queue.

Build Executor Status

1 Idle
 2 Idle

All

jenkins pipeline

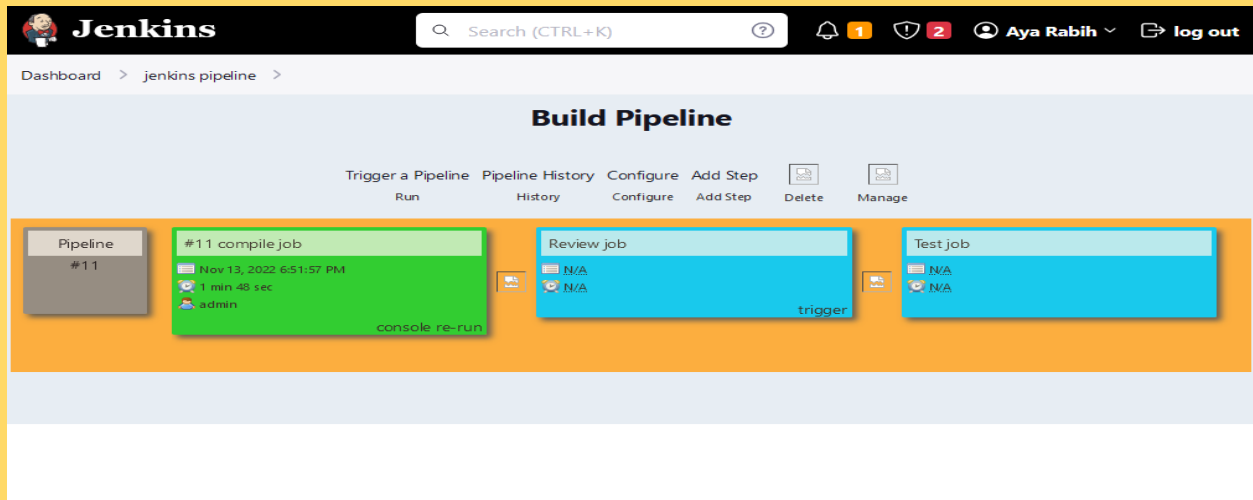
+

S	W	Name ↓	Last Success	Last Failure	Last Duration	
✓	☀	compile job	2 hr 20 min #11	N/A	1 min 48 sec	▶
✓	☀	Review job	56 min #10	N/A	1 min 39 sec	▶
✓	☀	Test job	47 min #1	N/A	26 sec	▶

Icon: S M L

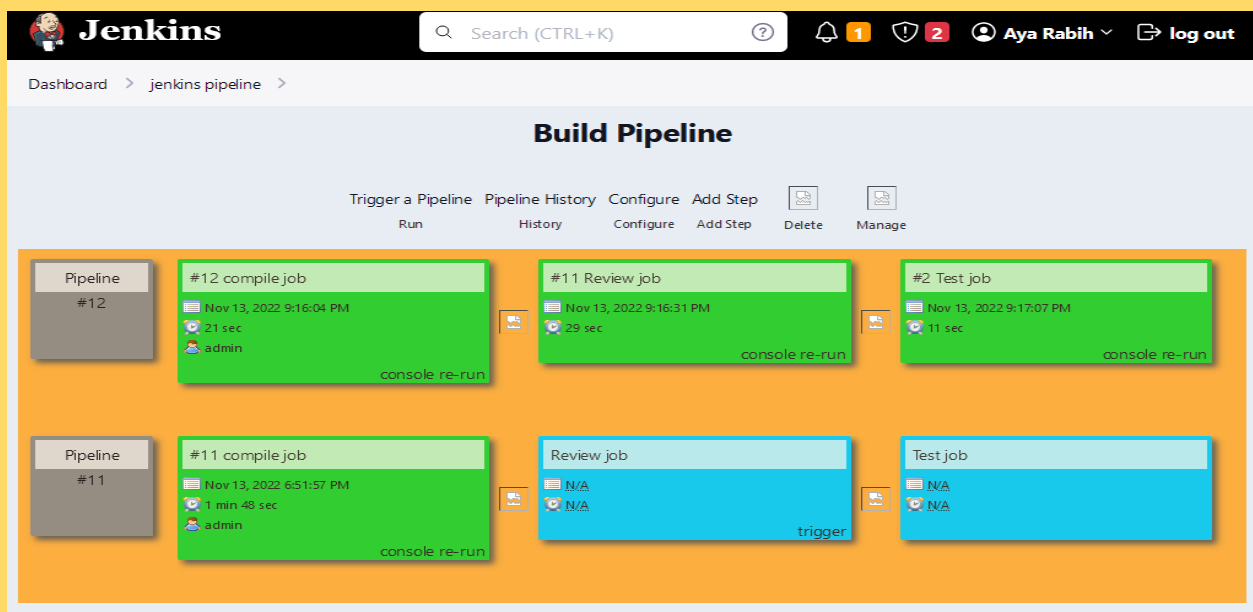
Icon legend
 Atom feed for all
 Atom feed for failures
 Atom feed for all failures

❖ Now we have all steps running



The screenshot shows the Jenkins 'Build Pipeline' view for pipeline #11. The top navigation bar includes the Jenkins logo, a search bar, and user information 'Aya Rabih'. The breadcrumb trail is 'Dashboard > jenkins pipeline >'. The main section is titled 'Build Pipeline' and contains a toolbar with 'Trigger a Pipeline', 'Pipeline History', 'Configure', 'Add Step', 'Delete', and 'Manage'. Below the toolbar, the pipeline is visualized as a sequence of four steps: 'Pipeline #11' (grey), '#11 compile job' (green, 'Nov 13, 2022 6:51:57 PM', '1 min 48 sec', 'admin', 'console re-run'), 'Review job' (blue, 'N/A', 'N/A', 'trigger'), and 'Test job' (blue, 'N/A', 'N/A'). All steps are shown as running.

❖ now all steps done



The screenshot shows the Jenkins 'Build Pipeline' view for pipeline #12. The top navigation bar is identical to the previous screenshot. The breadcrumb trail is 'Dashboard > jenkins pipeline >'. The main section is titled 'Build Pipeline' and contains the same toolbar. Below the toolbar, the pipeline is visualized as a sequence of four steps: 'Pipeline #12' (grey), '#12 compile job' (green, 'Nov 13, 2022 9:16:04 PM', '21 sec', 'admin', 'console re-run'), '#11 Review job' (green, 'Nov 13, 2022 9:16:31 PM', '29 sec', 'console re-run'), and '#2 Test job' (green, 'Nov 13, 2022 9:17:07 PM', '11 sec', 'console re-run'). All steps are shown as completed.

❖ all things perfect now